ВІСНИК
НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ
"Київський політехнічний інститут"

Інформатика, управління та обчислювальна техніка

54

2011

Міністерство освіти і науки України
Національний технічний університет України «КПІ»

ВІСНИК

НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТА УРКАЇНИ «КПІ»

Інформатика, управління
та обчислювальна техніка

Заснований у 1964 р.
Випуск 54

Київ “ВЕК+” 2011

УДК 004

Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Век+, – 2011. – № 54. – 239 с.

Рекомендований до друку Вченою радою факультету інформатики та обчислювальної техніки, протокол № 4 від 23.12.2011

Описано результати дослідження і створення компонентів обчислювальних й інформаційних систем і комплексів, пристроїв автоматики та передавання даних, систем автоматизації програмування, контролю й діагностики, штучного інтелекту тощо.

Для аспірантів, студентів, фахівців з обчислювальної техніки, систем керування, автоматизації програмування, штучного інтелекту та інших інформаційно-обчислювальних систем.

ISSN 0201-744X
ISSN 0135-1729

Свідоцтво про державну реєстрацію № 16949-5719 Р від 17.06.2010

Збірник наукових праць українською, англійською та російською мовами

Web-ресурс – http://it-visnyk.kpi.ua

Підп. до друку 23.12.2011. Формат 60×84 1/16. Гарнітура Times. Папір офсетний № 1. Наклад 150 прим.
Надруковано в ЗАТ "ВІПОЛ", 03151 м.Київ, вул.. Волинська, 60.

ЗМІСТ

Відомості про авторів

Баранов А.Ю.	аспирант института кибернетики им. В.М.Глушкова НАНУ
Богатырчук И.И.	аспирант кафедры ВТ НТУУ «КПИ»
Бойко Ю.В.	к.ф.-м.н., доцент, зав. каф. комп'ютерної інженерії РФФ КНУ
Бояринова Ю.Е.	к.т.н , с.н.с., ИПРИ НАНУ
Вітіщенко Н.С.	студентка кафедри АУТС, ФІОТ, НТУУ "КПІ"
Воробйов В.В.	студент кафедры ВТ НТУУ «КПИ»
Гаевская Е.А.	студент кафедры ВТ НТУУ «КПИ»
Гнатюк О.С.	студент кафедры ВТ НТУУ «КПИ»
Городько Н.А.	н.с. ИПРИ НАНУ
Гусев Е.И.	аспирант кафедры ВТ НТУУ «КПИ»
Дорогой Я.Ю.	ассистент кафедры АУТС НТУУ «КПИ»
Дьяконова С. В.	аспирантка НТУУ «КПИ»
Жабіна В.В.	к.т.н., старший викладач кафедри ПЗКС НТУУ «КПІ»
Зайченко Ю.П.	д.т.н., профессор кафедры ММСА УНК «ИПСА» НТУУ «КПИ»
Зволинський В.В.	студент кафедри ОТ НТУУ «КПІ»
Калиновский Я.А.	д.т.н., пр.н.с. ИПРИ НАНУ
Кириченко Л.О.	к.т.н., доцент кафедры прикладной математики Харьковского национального университета радиоэлектроники
Клименко И.А.	к.т.н., доцент кафедры компьютерных систем и сетей НАУ
Коган А.В.	аспирант кафедры ВТ НТУУ «КПИ»
Корначевський Я.І.	к.т.н. кафедри САПР НТУУ «КПІ»
Кузьмук А.В.	аспирант ИПМЭ им. Г.Е. Пухова
Кузьмук В.В.	д.т.н., проф., зав. отделом, ИПМЭ им. Г.Е. Пухова
Куссуль О.М.	аспирантка НТУУ "КПИ"
Кулаков А.Ю.	к.т.н., ст. преподаватель кафедры ВТ НТУУ «КПИ»
Кулаков Ю.А.	д.т.н., профессор кафедры ВТ НТУУ «КПИ»
Ладогубець В.В.	к.т.н. кафедри САПР НТУУ «КПІ»
Лисецкий Т.Н.	аспирант кафедры АСОИУ НТУУ «КПИ»
Малюков П.Н.	аспирант кафедры АУТС НТУУ «КПІ»
Марковський О.П.	к.т.н., доцент кафедры ВТ НТУУ «КПИ»
Минаев Ю.Н.	д.т.н., профессор кафедры компьютерных систем и сетей НАУ
Минаева Ю.И.	ассистент кафедры основ информатики КНУБА
Мурга М.О.	аспірант кафедри ММСА ННК «ІПСА» НТУУ «КПІ»
Надеран Е.	аспирант НТУУ «КПИ»
Нечипоренко О. М.	к.т.н., доцент кафедри приладів і систем керування літальними апаратами та комплексами, ФАКС НТУУ «КПІ»
Новиков А.Н.	д.т.н., профессор, директор Физико-технического института НТУУ "КПИ"
Новіков Ю.Л.	провідний інженер конструкторського бюро інформаційних систем НТУУ «КПІ»
Павлов А.А.	д.т.н., професор кафедры АСОИУ НТУУ «КПИ»
Павлов А.В.	аспирант МННЦ ІТС НАН та МОНМС України
Павлов В.А.	ст. викладач кафедри МКТМ ММІФ НТТУ «КПІ»
Пирогов А.А.	студент кафедры ВТ НТУУ «КПИ»
Пищаева Н.Н.	доц. кафедры АУТС НТУУ «КПІ»
Плакса О.С.	аспирант НТУУ «КПИ»
Полторак В.П.	к.т.н., доцент кафедры АУТС НТУУ «КПИ»
Ролик А.И.	к.т.н., ст.. науч. сотрудник, доцент кафедры АУТС НТУУ «КПИ»
Русанова О. В.	к.т.н., доцент кафедры вычислительной техники НТУУ "КПИ"
Савченко П.С.	студент НТУУ "КПИ"
Саидреза Махмали	аспирант кафедры ВТ НТУУ «КПИ»
Салапатов В.І.	к.т.н., доцент кафедры кибернетики, ЧНУ

Сальніков А.О.	аспірант кафедри комп'ютерної інженерії РФФ КНУ
Селиванов В.Л.	доцент кафедры ВТ НТУУ «КПИ», к.э.н.
Слюсар Є.А.	аспірант кафедри комп'ютерної інженерії РФФ КНУ
Стеценко І.В.	к.т.н., доцент кафедри комп’ютерних технологій Черкаського державного технологічного університету
Стиренко С.Г.	к.т.н., доцент каф. ВТ НТУУ «КПИ»
Супруненко О.А.	к.т.н., доцент кафедры программного обеспечения автоматизированных систем, ЧНУ имени Богдана Хмельницкого
Теленик С.Ф.	д.т.н., професор, зав. кафедры АУТС НТУУ «КПИ»
Турченко В.О.	к.т.н., доцент, с.н.с. НДІ ІКС ТНЕУ
Химич А.Н.	заведующий отделом Института кибернетики им. В.М.Глушкова НАН Украины, доктор физ.-мат. наук
Чистякова Т.В.	доктор. физ.-мат. наук, с.н.с. Института кибернетики им. В.М.Глушкова НАНУ
Федоречко О.І.	студентка кафедри ОТ НТУУ "КПІ"
Филимонова О.Ю.	к.т.н., доцент кафедры основ информатики КНУБА
Фоменко А.О.	асистент кафедри Інформаційних систем і технологій Бердянського університету менеджменту і бізнесу, аспірантка НТУУ "КПІ"
Шовгун Н.В.	аспирант ИПСА НТУУ «КПИ»
Ярох Ю. А.	магистр кафедры вычислительной техники НТУУ "КПИ"

УДК 681.513.8

ПАВЛОВ А.В.,
ПАВЛОВ В.А.

МЕТОДИКА ЭКСПЕРИМЕНТАЛЬНЫХ ИССЛЕДОВАНИЙ СХОДИМОСТИ ИТЕРАЦИОННЫХ АЛГОРИТМОВ МЕТОДА ГРУППОВОГО УЧЁТА АРГУМЕНТОВ

В работе предложена новая методика численного исследования сходимости итерационных алгоритмов. Основываясь на методике, экспериментально исследована скорость сходимости Обобщенного Релаксационного Итерационного Алгоритма (ОРИА).

The paper offers a new numerical investigation method for iterative algorithms convergence study. The method has been applied for convergence rate numerical investigation of generalized relaxational iterative algorithm of GMDH.

Введение

Данная работа посвящена описанию методики численного исследования сходимости. Экспериментально показана сходимость к решению и по структуре модели [1]. Поскольку любой алгоритм из ОРИА даёт один и тот же результат оценки параметров модели, проведем исследование для РИА, генерирующего Полное Дерево Структур (ПДС) [2].

Численное исследование скорости сходимости итерационных алгоритмов

Скорость сходимости итерационного алгоритма определяется количеством итераций r^*, необходимых для получения решения с заданной точностью ε. Однако, при проведении численных экспериментов, как правило, r^* существенно зависит от свойств (характеристик) данных, подаваемых на вход алгоритма. Такими характеристиками могут быть коррелированность столбцов или строк входной матрицы, число её обусловленности и др. Поэтому к ряду исследуемых параметров алгоритма (например, свобода выбора F на каждой итерации, количество и вид одночленов модели и т.д.) следует до бавить характеристику данных, влияющую на скорость его сходимости. Не ограничивая общ ности, пусть это будет одна характеристика Pr.

Для осуществления численных экспериментов используют Процедуру Генерации Матрицы (ПГМ), в основе которого лежит Генератор Псевдослучайных Чисел (ГПЧ). В экспериентах необходимо генерировать матрицы со свойствами, не влияющими на скорость сходимости алгоритма. Однако даже при настройке ПГМ на генерацию матрицы с определённым значением параметра Pr, этот параметр является случайной величиной. Поэтому необходимо настроить ПГМ так, чтоб математическое ожидание параметра Pr генерируемых матриц было близко к заданному значению. Для получения объективных результатов сходимости алгоритма при разных значениях его параметров и параметра Pr, предлагается следующая методика проведения экспериментальных исследований.

Методика проведения экспериментальных исследований

Идея состоит в осуществлении серии генераций входных матриц и получении гистограммы распределения значений параметра r^* при заданных значениях параметров алгоритма и параметра Pr. Для определения необходимого числа генераций матриц (количества реализаций) RN^* задаётся значение экспериментальной погрешности δ в процентах.

Методика численного исследования сходимости алгоритма моделирования состоит из трёх этапов:

Этап 1. Определить параметры ПГМ и их значения, позволяющие генерировать матрицы с математическим ожиданием параметра Pr, близким к заданному значению.

Этап 2. Исследовать скорость сходимости алгоритма (число итераций, необходимых для получения решения с заданной точностью) при изменении значения параметра Pr и неизменных параметрах алгоритма.

Этап 3. Для каждого из параметров алгоритма исследовать скорость сходимости при из менении его значения и неизменных значениях остальных параметрах (включая параметр Pr).

На каждом из этапов выполняется алгоритм *Explorer*, входным параметром которого является исследуемый параметр $par \in \{Pr, F, \ldots\}$.

Алгоритм *Explorer*

Шаг 1. Определить количество реализаций RN^*, удовлетворяющее заданному значению δ для гистограмм распределения параметра r^* при заданном значении параметра *par*.

1.1 Сгенерировать матрицу с помощью ПГМ.

1.2 Выполнить исследуемый алгоритм построения модели, подав на его вход сгенерированную матрицу.

1.3 Добавить полученное значение r^* в гистограмму.

1.4 Выполнить п.п. 1.1-1.3 заданное число раз *RN*.

1.5 Выполнить п. 1.4 для разных значений *RN*, увеличивая его до тех пор, пока мера отличия гистограмм для двух последовательных значений *RN* не станет удовлетворять заданной погрешности δ.

Шаг 2. Построить гистограммы распределения параметра r^* используя найденное количество реализаций RN^* для разных значений параметра *par*.

На первом этапе методики входным параметром алгоритма *Explorer* является параметр *Pr*. При этом в алгоритм *Explorer* добавляется третий шаг:

Шаг 3. Выполняется поиск значений параметров АГВМ, при которых математическое ожидание параметра *Pr* генерируемой матрицы близко к заданному значению.

Численное исследование скорости сходимости РИА ПДС

В экспериментах строятся линейные по аргументам модели. Будем исследовать сходимость алгоритма при условии, что матрица X_A содержит только истинные аргументы. Для описания ПГМ введём обозначения: W – выходная матрица ПГМ, $W = (X_A \vdots y)$, $\dim W = n \times (s_{\text{lin}} + 1)$; n – число наблюдений; s_{lin} – количество линейных истин-ных аргументов. Входными параметрами ПГМ являются: n, s_{lin}.

Процедура генерации матриц

1. Сгенерировать матрицу X_A, $\dim X_A = n \times s_{\text{lin}}$.

2. Для аргументов матрицы X_A сгенерировать вектор коэффициентов Θ со значениями в интервале $[a; b]$.

3. Рассчитать вектор выхода сгенерированной модели.

В ПГМ используется один из широко известных ГПЧ – Mersenne Twister MT19937 [3]. Период генератора равен 2^{19937}. Случайные числа генерируются по равномерному закону распределения на интервале [0; 1]. Применим методику для исследования скорости сходимости РИА ПДС.

Этап 1. Можно показать, что скорость сходимости РИА ПДС зависит от такой характеристики матрицы X_A, как мера ортогональности её вектор-столбцов. Сходимость РИА ПДС отслеживается по разности нормированной остаточной суммы квадратов $NRSS_A$ на двух соседних итерациях: $NRSS_{A,r} - NRSS_{A,r+1} = \Delta_{r+1}$. Останов алгоритма осуществляется при условии $\Delta_{r+1} < \varepsilon$. Формула для расчёта $NRSS_A$ имеет вид:

$$NRSS_A = \sum_{i=1}^{n} (\tilde{y}_{A,i} - \tilde{\hat{y}}_{A,i})^2 \Big/ \sum_{i=1}^{n} \tilde{y}_{A,i}^2 ,$$

где $\tilde{y}_A$, $\tilde{\hat{y}}_A$ – центрированные на выборке A значения векторов исходного и моделируемого выхода y и $\hat{y}$.

Несложно показать, что если вектор-столбцы матрицы X_A образуют ортонормированную систему, корреляционная матрица $\Sigma_X = X_A^T X_A$ является единичной с детерминантом равным 1. Поэтому в качестве меры ортогональности вектор-столбцов матрицы X_A выбрано значение детерминант d корреляционной матрицы Σ_X. Поскольку для меры ортогональности не важен знак парной корреляции между вектор-столбцами матрицы X_A, матрица Σ_X содержит модули значений. Следовательно, $d \in [0; 1]$, причём, если $d = 0$, матрица Σ_X – вырожденная, а при $d = 1$, она – единичная.

Определим параметры ПГМ, влияющие на значение детерминанта d матрицы корреляций. Как показывает анализ результатов генерации двух случайных вектор-столбцов в ПГМ, они обладают следующим свойством: чем больше число точек n в векторах, тем меньше значение их парной корреляции. Это объясняется тем, что чем больше случайных элементов в векторах, тем более они ортогональны между собой в n-мерном пространстве точек. Если это свойство обобщить на всю матрицу Σ_X, то число наблюдений можно использовать для генерации матрицы X_A с заданным значением детерминанта её корреляционной матрицы.

Подтвердим этот вывод экспериментально. Исследуем, как изменяется гистограмма рас-

пределения r^* и соответствующее значение усреднённого детерминанта d_{aver} при изменении параметра n. В соответствии с методикой вначале необходимо определить количество реализаций RN^* из некоторого множества значений.

Гистограммы строились для следующих значений $RN \in \psi$, $\psi = \{10^3, 10^4, 10^5, 10^6\}$. Частота (вероятность), указанная на рисунках ниже – это отношение количества матриц, при которых алгоритм находит модель с заданной точностью ε за количество итераций из соответствующего интервала r^* к общему количеству матриц, равному RN.

Обычно в качестве меры отличия гистограмм используют статистический крите-рий Пирсона χ^2 распределения параметра r^*. При этом минимальной мере отклонения соответствует оптимальное значение RN^*. Однако, учитывая известный произвол при выборе допустимой величины уровня значимости для χ^2-критерия, и необходимость разработки простой процедуры для автоматической многократной проверки гистограмм предлагается следующая мера M.

Процедура вычисления меры отличия гистограмм

Пусть количество интервалов параметра r^* гистограмм равно $IntNum$. Мера M_i, (i = 1, 2, 3) пары соседних значений из множества ψ рассчитывается следующим образом:

1. Для каждого интервала значений r^* гистограмм меньшее значение частоты делится на большее. Получаем $IntNum$ значений точности $\gamma_j \in [0; 1]$, $j=\overline{1, IntNum}$.

2. $M_i = \frac{1}{IntNum}\sum_j \gamma_j$.

Выполнив данную процедуру для всех пар, получаем множество значений мер отличия гистограмм: {82.1%, 91.5%, 98%}, которые можно интерпретировать как точность моделирования.

Гистограммы для последней пары показаны на рис. 5. Как видно из рисунка, если $RN = 10^6$ взять за «эталон» полученная погрешность $\tilde{\delta}$ = =2% (M_3 = 98%) удовлетворяет значению заданной погрешности δ = 5%, то значение RN^*= 10^5. Именно это значение используется в дальнейших экспериментах, поскольку, как было установлено в численных экспериментах для ряда исследуемых значений параметров алгоритма моделирования, на первом шаге выполнения алгоритма *Explorer* были получены аналогичные гистограммы.

В данном эксперименте строилась модель:

$$\breve{y} = 6.29447 + 8.11584\cdot x_1 - 7.46026\cdot x_2 - 7.29046\cdot x_3 + 6.70017\cdot x_4 + 9.37736\cdot x_5. \quad (5)$$

Свобода выбора алгоритма F = 5, ошибка моделирования $\varepsilon = 10^{-12}$. Гистограммы r^* при изменении количества наблюдений n показаны на рис. 6. Над столбиками гистограмм представлены тысячные доли значений усреднённого детерминанта d_{aver}.

Усредненное значение детерминанта рассчитывается следующим образом. Вычисляются значения детерминантов всех матриц, для которых алгоритм сошёлся к решению за r^* итераций, попадающих в соответствующий интервал для r^*. Значение d_{aver} есть среднее от рассчитанных значений.

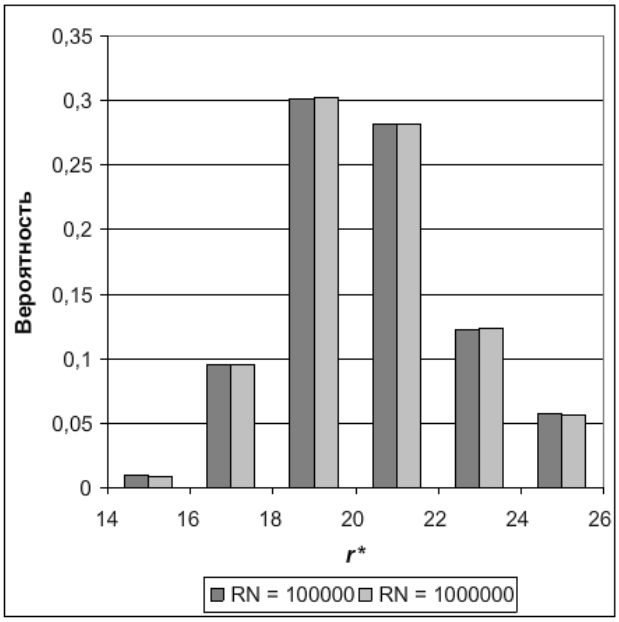


Рис. 5. Гистограмма распределения параметра r^* для последней пары значений из множества RN

Анализируя рис. 6, можно сделать выводы:

1. Параметр n прямо пропорционально влияет на значение детерминанта d, а значит, и на меру ортогональности векторстолбцов матрицы X_A. Следовательно, количество наблюдений можно использовать для получения матрицы корреляций с заданным значением детерминанта.

2. Чем больше значение n, тем меньше дисперсия параметра d генерируемых матриц. Этим объясняется увеличение вероятности генерации матриц с заданным значением d

3. Скорость сходимости алгоритма возрастает пропорционально увеличению значения параметра *d*.

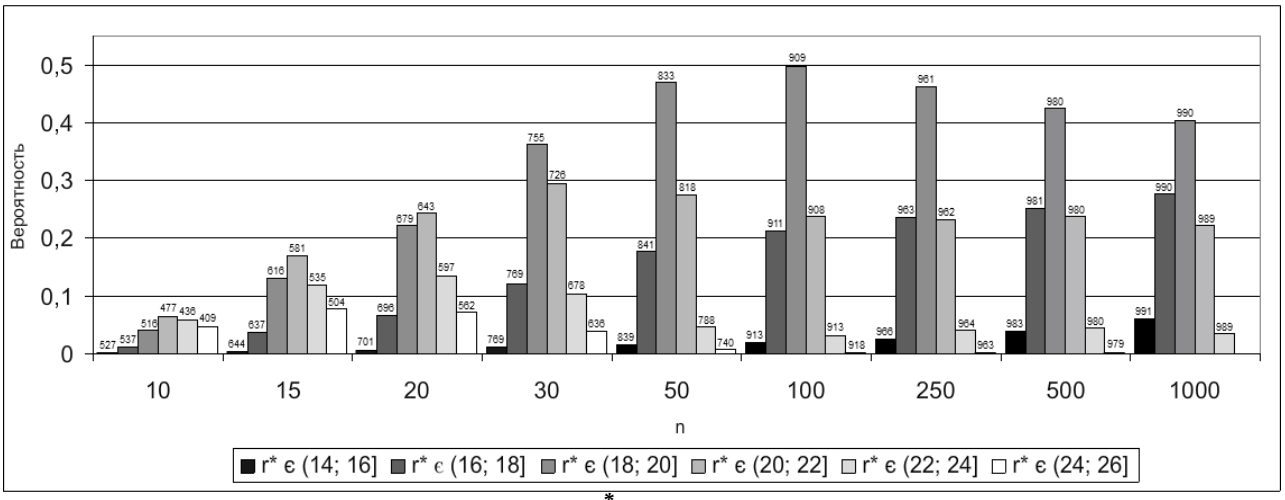


Рис. 6. Гистограммы распределения r^* при изменении количества наблюдений n

Для модели (5) были определены значения *n*, позволяющие генерировать матрицы X_A с заданным значением детерминанта $d \in D$, $D = \{0.5, 0.6, 0.7, 0.8, 0.9, 0.99\}$, см. табл. 1.

Табл. 1. Количество точек и соответствующее значение детерминанта

d	0.5	0.6	0.7	0.8	0.9	0.99
n	12	16	25	41	94	875

Этап 2. Исследуем, как скорость сходимости алгоритма зависит от параметра *d*.

В эксперименте определялась модель (5). Гистограммы распределения параметра r^* для $d \in D$ показаны на рис. 7. Как видно на рисунке, между значением детерминанта и скоростью сходимости алгоритма имеется прямая пропорциональная зависимость.

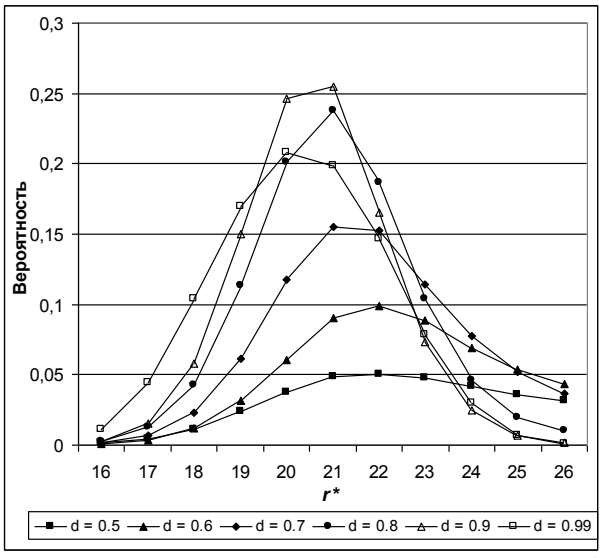


Рис. 7. Гистограммы распределения r^* при разном значении детерминанта

Этап 3. Исследования осуществим для модели (5), варьируя параметром свободы выбора *F* при $d = 0.7$, из табл. 1 соответствующее $n = 25$. Результаты представлены на рис. 8. Из анализа рисунка заключаем, что свобода выбора прямо пропорционально влияет на скорость сходимости алгоритма.

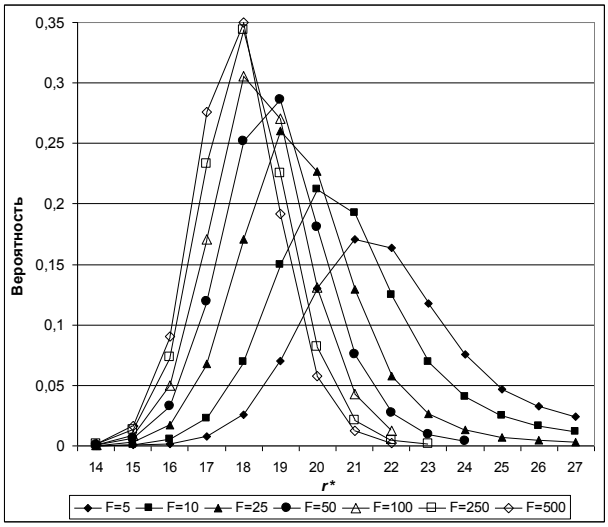


Рис. 8. Гистограммы распределения r^* при изменении параметра F

Исследуем, как изменяется значение критерия $NRSS_A$ на каждой итерации с изменением параметра *F*. В данном эксперименте усреднялись модели, отвечающие пикам гистограмм рисунка 8. Пиковая частота соответствует максимальной частоте на гистограмме. Усреднение осуществлялось по коэффициентам и критерию моделей. Значение r^*, соответствующее пику обозначим через r_{prob}. Строились семейства усреднённых лучших моделей на итерациях с номерами 5, 7, 9, 11, 13, 15, 17 для каждой пары (F, r_{prob}). Результаты в логарифмической шкале представлены на рис. 9

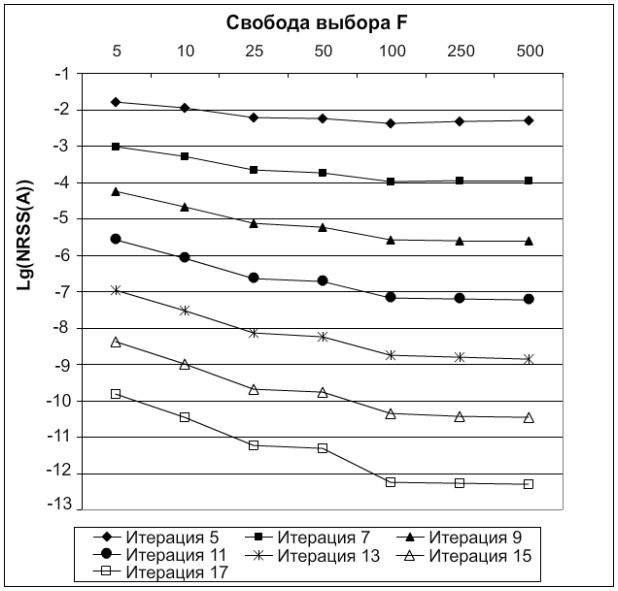


Рис. 9. Изменения минимума критерия $NRSS_A$ при изменении параметра F для выбранных номеров итераций

Анализ результатов, представленных на рис. 9, приводит к следующим выводам:

1. Свобода выбора существенно влияет на изменение минимума критерия $NRSS_A$, причём, чем больше номер итерации, тем это влияние сильнее.

2. Существует значение F, при превышении которого, влияние этого параметра значительно снижается. В данном примере это значение равно 100.

В табл. 3 проиллюстрирована сходимость по структуре и параметрам лучшей усреднённой модели для варианта $F = 50$. В ячейках таблицы содержатся коэффициенты соответствующих аргументов, $\hat{\theta}_0$ – оценка свободного члена модели.

Как видим из таблицы 3 алгоритм обладает достаточно быстрой сходимостью: для получения решения с погрешностью $\varepsilon = 10^{-12}$ ему необходимо осуществить $3s_{lin}$ итераций.

Выводы

С целью исследования скорости сходимости ОРИА предложена новая методика численного исследования итерационных алгоритмов. Основываясь на методике, экспериментально исследована скорость сходимость РИА ПДС. При этом выявлены следующие особенности:

– количество наблюдений можно использовать для генерации входной матрицы с заданым значением детерминанта её корреляционной матрицы;

– значение детерминанта (характеристика исходных данных) и свобода выбора (характеристика алгоритма) связаны со скоростью сходимости к точному решению прямо пропорционально;

– алгоритм обладает быстрой сходимостью при построении линейных моделей.

Табл. 3. Результаты сходимости РИА ПДС к решению

№ итерации	$NRSS_A$	$\hat{\theta}_0$	x_3	x_1	x_4	x_2	x_5
5	0,005473	6.35397	-7.16523	7.99056	6.57297	-7.46026	9.37736
7	0,000184	6.29582	-7.28527	8.11217	6.69561	-7.33251	9.25849
9	$5{,}82\cdot10^{-6}$	6.29464	-7.2905	8.11572	6.69998	-7.45551	9.37293
11	$1{,}8\cdot10^{-7}$	6.2945	-7.29044	8.11584	6.70012	-7.46026	9.37737
13	$5{,}61\cdot10^{-9}$	6.29448	-7.29046	8.11583	6.70017	-7.46028	9.37735
15	$1{,}69\cdot10^{-10}$	6.29447	-7.29046	8.11584	6.70017	-7.46026	9.37736
Истинная модель:		6.29447	-7.29046	8.11584	6.70017	-7.46026	9.37736

Список литературы

1. Юрачковский Ю.П. Сходимость многорядных алгоритмов МГУА // Автоматика, 1981. – № 3. – С. 36-43.
2. Павлов А.В. Обобщённый релаксационный итерационный алгоритм МГУА // Індуктивне моделювання складних систем. Збірник наук. праць. – К.: МННЦІТС, 2011. – С. 130-143.
3. Доступно на сайте http://www.boost.org/.

УДК 004.93(045)

ТЕЛЕНИК С.Ф.,
МАЛЮКОВ П.Н.,
ПИЩАЕВА Н.Н.

НЕЙРОСЕТЕВОЙ МЕТОД КЛАССИФИКАЦИИ АНОМАЛИЙ В ТРАФИКЕ ПРОВАЙДЕРА МОБИЛЬНОЙ СВЯЗИ

Рассмотрены популярные нейронные сети для классификации фродов в трафике провайдера. Проанализированы алгоритмы для определения лучшего варианта. При помощи разработанных программ проведены исследования для сравнения быстродействия и качества классификации.

Popular neural networks for classification of fraud in the traffic of the operators are considered. Algorithms for definition of the best variant are analyzed. By means of the developed programs researches for comparison of speed and quality of classification are carried out.

Введение

По мере увеличения масштабов и сложности современных сетей связи растет, и число попыток использовать широко разветвленную телекоммуникационную инфраструктуру в противоправных целях. Несанкционированные, сомнительные и откровенно мошеннические действия приводят не только к ощутимым потерям в доходах операторов, но и к снижению объемов и качества предоставляемых услуг.

Цепочка формирования доходов провайдера связи начинается с сетевого оборудования, обеспечивающего собственно предоставление услуги связи и фиксирующего это событие. Затем эта информация проходит через разнообразные промежуточные программные и аппаратные комплексы, агрегирующие и трансформирующие данные. Завершается все биллинговой системой, которая тоже состоит из нескольких модулей, ответственных за тарификацию услуг, прием платежей, выставление счетов и т.д. К этому следует добавить систему обслуживания клиентов, отчетности, управление услугами и т.п., то видно, насколько сложна дорога от предоставления телекоммуникационных услуг до получения за нее денег оператором. На каждом участке этого пути информацию поджидают сбои, конфликты оборудования, ошибки в настройках систем и, конечно же, воровство или фрод (мошенничество в телекоммуникационной сфере).

Задача противодействия мошенничеству и гарантирования доходов актуальна для телекоммуникационных компаний и сегодня эта тема выходит на новый уровень значимости по нескольким причинам:

– мировой фінансово-экономический кризис, выдвигающий особенно жесткие требования к эффективности ведения бизнеса;

– насыщения рынка телекоммуникационных услуг и обострение конкуренции;

– новая волна государственных регуляторов к тому, как компании осуществляют мониторинг и отчетность своей выручки;

– стремительное развитие технологий и маркетинговых инноваций, за которыми развитие методов и инструментов контроля просто не успевают.

Анализ проблемы

Одной из актуальных задач, стоящих перед компаниями – операторами связи является минимизация финансовых потерь, которые могут бать следствием мошеннических действий абонентов. Под воздействием фрода падает эффективность использования сетевых ресурсов, страдает имидж компании, уходят ценные клиенты, замедляется продвижение новых телекоммуникационных услуг. Основные услуги сотовой связи включают передачу голосового трафика, передачу факсимильных, коротких и мультимедийных сообщений, передачу данных, предоставление всевозможных контентных сервисов, доступ к сети интернет, мобильное телевидение и.т.п. К дополнительным услугам можно отнести переадресацию вызова, сохранение вызова и ожидание вызова, услугу конференц – связи позволяющей вести разговор по телефону одновременно несколькими абонен-

тами, услугу запрета (или ограничения) автоматического определения вызывающего абонента, голосовую почту, роуминг и т. д.[1]

Основные типы потерь от мошенничества в телекоммуникационных сетях таковы:

– отсутствие корректной полной информации о предоставленных услугах телекоммуникаций;

– неверная обработка такой информации;

– использование технических и логических ошибок в работе билинговой системы;

– несобранная выручка (задолжности абонентов);

– недополученная прибыль (оператор телекоммуникаций получил меньше, чем мог бы). Оператору необходима классификация видов мошенничества, которая помогла бы ему упорядочить деятельность по борьбе с фродом в своей сети. Классификация фрода в телекоммуникациях чрезвычайно затруднена, поскольку мошенники постоянно совершенствуют свои навыки и способы отъема денег у операторов и клиентов связи. Одним из критериев дифференциации видов фрода можно использовать направленность действий злоумышленников: в одних случаях пострадавшей стороной являются абоненты, в других – сами операторы. На сегодня Ассоциациями CFCA определено более 200 видов мошенничества в телекоммуникациях.

Виды мошенничества в сетях мобильной связи:

– клонирование SIM – карт мобильных телефонов (для организации междугородних переговорных пунктов, одновременного использования неограниченных тарифных планов рядом лиц, противоправных действий под видом истинного владельца SIM карты, если ее клонировали без ведома владельца);

– использование реквизитов подставных лиц при регистрации в сети (для заключения контракта на корпоративное обслуживание группы абонентов с кредитной формой оплаты) ;

– незаконное завершение соединения;

– создание «чужих» SIM – центров (в результате «свой» SIM – центр не получает информацию об отправленном сообщении, и оно не тарифицируется);

– роуминговый фрод (генерация трафика на дорогостоящие PRS направления из расчета на задержку своевременной тарификации роумера в билинге домашней сети);

– Bypass (нарушение порядка маршрутизации международного трафика в результате чего дорогостоящие роуминговые звонки тарифицируются оператором как его условно бесплатные внутресетевые);

– мошенничество со службами заказа билетов, оплаты мелких товаров ,услуг, перевод денег с банковских счетов при помощи SMS – транзакций и т.д.

Виды внутреннего мошенничества – мошенничество сотрудников самого оператора телекоммуникаций:

– умышленная активация неоплаченных сервисов;

– мошенничество при использовании предоплаченных услуг;

– ошибочная тарификация.

Для активного противодействия фроду необходимы регулярные сбор и анализ информации о действии злоумышленников, выявление тенденций, всесторонний анализ тарифных планов и маркетинговых ходов, прогнозирование новых схем мошенничества и создание активных способов защиты. Для борьбы с проявлениями мошенничества в телекоммуникациях применяются специализированные системы класса Fraud Management System (FMS), которые способны обнаруживать проявления мошенничества.

По мере развития систем сотовой мобильной связи развиваются методы и приемы фрода. С одной стороны, постоянно возникают новые услуги, новые технологии и, как следствие, новые виды фрода. С другой стороны, телекоммуникационное мошенничество, в отличие от традиционных угроз информационной безопасности, весьма «операторо специфично». Оно слишком сильно завязано на конкретные реализации тех или иных услуг определенного оператора, на его системы, его процессы и т.д. Помимо общих проблем фрода, у каждой телекоммуникационной компании будет свой специфический набор фродовых схем, присущих только ей. Поэтому при планировании своей деятельности, операторам связи предлагается воспользоваться топологизацией фродовых схем на основе методов их выявления. Такая классификация представляет собой законченный, ограниченный набор классов фрода. Каждая вновь возникающая, в том числе уникальная для данного оператора, фрод схема может быть отнесена к одному из этих классов. [2]

Цель работы

Целью данной работы является исследование и реализация нейросетевой технологии классификации аномалий трафика провайдера, для повышения достоверности решения задачи обнаружения фрода и обеспечения ее работы в реальном времени.

Постановка задачи

Обычно защита основана на анализе телефонии контрольных выборок абонентов и обнаружении разного рода закономерностей не характерных профилю эталонного абонента подобного тарифного плана, анализе подозрительных событий, например, сильный рост трафика абонента и т.д. Таким образом, речь идет о сканировании большого объема информации и выделении в нем определенного паттерна, с большой вероятностью соответствующему фроду. Это типичная задача классификации, для решения которой могут использоваться как классические методы [3], так и нейронные сети. Внедрение существующих систем противодействия мошенничеству – дело довольно сложное и дорогое. Требуется серьезная интеграция в инфраструктуру оператора и обработка больших объемов разнообразных данных из разных систем, которые у каждого оператора свои. Настройка этого процесса требует привлечения специалистов высокой квалификации. Может оказаться и так, что внедрение дорогостоящей системы будет просто не нужно, так как наиболее приоритетный риск удастся закрыть другими способами с меньшими затратами. [4]

Проблема классификации (выделения отдельных групп) рассматривается в статистике как задача группировки исходных данных. Идея структурности реализована в кластерном анализе, основная цель которого – выделение в многомерных данных такие однородные подмножества, чтобы объекты внутри групп были близки (похожи) в известном смысле друг от друга, а объекты из разных групп – не похожи.

Метод решения задачи классификации

Нейронные сети характеризуются нечетким представлением и переработкой информации, высоким параллелизмом обработки информации и распределенным хранением информации. В нашем исследовании мы ограничились однослойной нейронной сетью (персептроном**)** и многослойной нейронной сетью с прямыми и обратными связями.

В качестве функции активации F , преобразующей взвешенную сумму входных сигналов S в выходной сигнал $y = F(S)$, использовались

– сигмоидная

$$y = \frac{1}{1 + e^{-cS}},$$

где $c>0$ – коэффициент ширины сигмоиды по оси абсцисс;

– гиперболический тангенс

$$y = th(cS) = \frac{e^{cS} - e^{-cS}}{e^{cS} + e^{-cS}}.$$

Взвешенная функция входных сигналов имеет вид

$$S = \sum_{j=0}^{n} w_j x_j ,$$

где $x_1..x_n$ – значения, поступающие на входы (синапсы) нейрона, $w_1..w_n$ – *веса* синапсов. [5]

В качестве среды проектирования был выбран продукт компании Sybase и язык высокого уровня C#. Отказ от использования указателей приводит к незначительному снижению производительности, но автоматический сбор мусора и расширенный функционал механизмов работы с коллекциями упрощают программную реализацию, делают программный код проще для поддержки и восприятия.

Программная часть состоит из библиотеки ClassificationLibrary.dll, реализующей методы классификации и графического интерфейса Classification.exe.

Основной класс Network для работы с односвязной и многослойной сетью прямого распространения (персептрон) содержит в себе методы, выполняющие следующие функции:

- добавления слоя с заданным количеством входов и выходов, задания функции активации с ее параметрами в качестве аргументов;
- вычисления выхода сети для объекта;
- возвращает номер нейрона с максимальным выходом;
- обучения многослойной сети методом обратного распространения. Функция принимает в качестве параметров список элементов, целевые значения выходов, количество шагов обучения и скорость обучения нейронной сети;
- создания и обучения однослойной нейронной сети. Функция использует дельта-правило.

С учетом выбранной структуры сети было принято решение отойти от принципов ООП и отказаться от ввода объекта для отдельного нейрона.

Работая с представлением слоя сети матрицей синаптических весов, мы получаем заметный прирост производительности за счет повышения скорости обращения к элементам матрицы в сравнении с поэтапным переходом к элементу в векторе снаптических весов.

Переход от общепринятой в нейронных сетях поэлементной функции активации к векторной позволяет реализовать слои с выходом типа «победитель забирает все», а также поднять производительность за счет уменьшения количества вызовов функции.

В процессе обучения сети необходимо вычислять производную от функции активации нейрона. Поскольку функция может принимать произвольную форму и вычисление ее производной по аналитической формуле может быть затруднительным, вместо производной используется первая конечная разность.

На основании этих решений был построен класс NetLayer, включающий методы: вычисления выхода слоя сети и вычисления первой конечной разности для последнего выхода сети. Свойства класса NetLayer содержат матрицу синаптических весов, копии последнего входного и выходного векторов. Сеть представляется списком объектов NetLayer, реализующих слой сети.

В процессе обучения нейронных сетей задавались обучающие выборки с различным количеством объектов, признаков и классов. Повторное обучение показало лучшие результаты, что свидетельствует о склонности алгоритма обучения «застревать» в локальных экстремумах целевой функции. Однослойная искусственная нейронная сеть показала себя достаточно хорошо на наборах с небольшим количеством классов.

Двухслойная сеть, обученная методом градиентного спуска, показывает схожие результаты. Среди отличий следует отметить значительно большее время обучения. Тестирование двухслойной сети показало ее высокую чувствительность к количеству нейронов скрытого слоя. Процент ошибки на однослойной сети не превысил 10% , для многослойной сети – 5%.

Выводы

Сети показали свою высокую чувствительность к входным наборам данных, т.е. время обучения нейронной сети напрямую зависит от количества признаков, характеризующий трафик. Таким образом, для повышения качества классификации в дальнейшем необходимо рассмотреть вопрос сжатия пространства признаков.

Список литературы

1. Попов В.И. Основы сотовой связи стандарта GSM. – М:Зко-Трендз,2005. – 296 с.
2. Шонин Д. Прагматическая классификация телекоммуникационного фрода, Jet Info, 11,2010 – С.10 -14
3. Малюков П.Н.Сравнительный анализ методов классификации аномалий в трафике провайдера мобильной связи /П.Н. Малюков, Н. Н. Пищаева / Електроніка та системи управління: сб. наук. праць – К.:НАУ, 2010. – Вип.2 (24). – С. 148 – 152
4. Шонин Д. Гарантирование доходов и противодействие мошенничеству в телекоммуникационных компаниях, Jet Info, 2,2010 – С.6 -11
5. Нейроинформатика / А.Н.Горбань, В.Л.Дунин-Барковский, А.Н.Кирдин и др. - Новосибирск: Наука. Сибирское предприятие РАН, 1998

УДК 004.932.75'1

ЗАЙЧЕНКО Ю. П.,
НАДЕРАН Е.

СТРУКТУРНЫЙ АНАЛИЗ РУКОПИСНЫХ МАТЕМАТИЧЕСКИХ ВЫРАЖЕНИЙ

В настоящей статье рассматривается разработанный метод структурного анализа для онлайн распознавания рукописных математических выражений. Рассматриваются следующие алгоритмы: размещения, чередования, группировки.

This article discusses the developed algorithm of structural analysis for online recognition of handwritten mathematical formulas. Introduce the following algorithms: placement, interleaving, grouping.

Введение

В настоящее время, в связи с развитием планшетных персональных компьютеров, появляется необходимость ввода данных без использования клавиатуры. Математические выражения составляют основную часть в большинстве научных и технических дисциплин. Ввод математических выражений в компьютеры является более сложным, чем ввод обычного текста, так как математические выражения, как правило, состоят из специальных символов и греческих букв в дополнение к английским символам и цифрам. Задачу распознавания рукописных математических выражений можно разделить на два основных этапа: распознавание символов и структурный анализ. Такое разделение на два этапа имеет ряд преимуществ, в частности подзадачи могут быть решены независимо друг от друга, что позволяет проанализировать результаты и сделать улучшения на каждом из этапов, при этом сохранится целостность всей системы. Целью данной статьи является описание разработанного метода структурного анализа, который позволяет определить пространственные отношения между символами и полностью отобразить структуру, написанного пользователем математического выражения.

Одна из самых ранних работ по распознаванию математических выражений была написана Андерсоном [1]. Для определения структуры математического выражения, Андерсон использовал нисходящий подход. Алгоритм начинается с одной окончательной синтаксической цели и попыток разделить данную цель на подцели до тех пор, пока все подцели не будут выполнены или все попытки потерпят неудачу. Однако эксперименты показали, что данный алгоритм структурного анализа не всегда эффективен при работе со сложными математическими выражениями, поскольку разделения до n-1 могут быть произведены множеством из n символов, в свою очередь каждое из этих разделений может дополнительно генерировать другие разделения, что ведет к неконтролируемому увеличению подцелей. В [2] в структурно-аналитическом этапе был использован нисходящий парсинг для разделения выражения на подвыражения, затем применялся восходящий парсинг, для того чтобы объединить подструктуры в общую структуру. Однако эксперименты были выполнены только на нескольких простых математических выражениях. В [3] был предложен редактор ввода математических выражений. Система позволяет пользователю вводить цифры и символы в любом порядке. Для того чтобы распознать символы, используется алгоритм соответствия образцу, что накладывает определенные ограничения на работу с программой. В [4] используется подход, основанный на скрытых Марковских моделях. Предполагается, что пользователь всегда пишет выражение в определенном порядке, например, при написании дроби, сначала должен быть написан числитель, затем дробная линия и знаменатель. Такое требование может быть легко нарушено на практике из-за высокой изменчивости стиля написания у различных пользователей.

Предлагаемый в данной статье метод структурного анализа, включающий в себя алгоритмы размещения, чередования и группировки, позволяет анализировать математические выражения любой сложности. Рассматриваемый метод анализирует написанное пользователем математическое выражение после каждого внесенного изменения, что позволяет записывать

составляющие математического выражения в любом порядке, а также вносить изменения в уже написанное выражение.

Обзор процесса распознавания

Дадим определение термина математическое выражение – это символическая запись законченного логического суждения, которая может состоять из цифр, букв и различного вида символов, при этом все цифры, буквы и символы формируют внутреннюю иерархическую структуру. Для того чтобы правильно распознать математическое выражение необходимо точно знать значение каждого из символов, которые в него вошли, и определить расположение этих символов относительно друг друга.

В данной статье будет подробно рассмотрен метод структурного анализа, позволяющий определить расположение символов относительно друг друга. Но сначала кратко рассмотрим процесс сегментации и непосредственно распознавания каждого отдельно сегментированного символа, и после перейдем к описанию самого метода структурного анализа. Такой подход позволит создать полную картину процесса онлайн распознавания рукописных математических выражений.

В предлагаемой системе, используется набор из 77 символов, который позволяет писать тригонометрические, логарифмические функции, интегралы, греческие и латинские буквы.

На первом этапе происходит сегментация символов. В рукописном математическом выражении обычно символы написаны отдельно друг от друга. Будем считать, что каждый математический символ состоит из одного или нескольких отрезков. Отрезок состоит из точки касания пера, точки отрыва пера и всех точек лежащих между ними. При добавлении новых отрезков или удалении уже существующих, может значительно измениться значение символа, поэтому происходит динамическая интерпретация символа после каждого внесенного пользователем изменения.

Для решения задачи распознавания изолированных символов был выбран нечеткий классификатор NEFClass. Система NEFClass прекрасно справляется с необходимой задачей классификации символов, поскольку сочетает в себе нейронные сети и нечеткие системы логического вывода, при этом обладает высокой скоростью работы алгоритмов обучения. Для задачи распознавания рукописных символов

0	7	e	l	s	z	{	.	Ω	ν	Δ
1	8	f	m	t	+	}	,	π	!	∇
2	9	g	n	u	-	[	∑	α	≈	λ
3	a	h	o	v	/	]	∞	β	γ	σ
4	b	i	p	w	*	<	≥	μ	θ	∫
5	c	j	q	x	(	>	≤	φ	ε	∂
6	d	k	r	y	)	%	=	δ	∈	∋

Рис. 1. Рассматриваемые символы в системе онлайн распознавания математических выражений

был построен нечеткий классификатор с 12 входами и 77 выходами.

Входам системы соответствуют признаки, которые были подобраны специально для математических символов, цифр, латинских и греческих букв, а выходам соответствует набор из 77 символов, которые пользователь может использовать при написании математических выражений. В качестве алгоритма обучения был выбран алгоритм сопряженных градиентов, который обеспечивает сходимость к оптимальному решению за конечное число шагов. В результате этапа распознавания, на выходе нейронной сети будет получен один из символов.

Структурный аналіз

Основной задачей этапа структурного анализа является определение расположения символов в математическом выражении относительно друг друга.

Рассмотрим предлагаемый метод более подробно. Все символы были разделены на группы, в зависимости от допустимых позиций размещения других символов относительно них. На Рисунке 2 схематически изображены все восемь возможных позиций.

Рис. 2. Схематическое изображение допустимых позиций расположения символов относительно текущего рассматриваемого символа

Были выделены следующие группы символов:

1. Цифры. Принимают значения от 0 до 9, π, ∞. Символы могут быть размещены относительно данной группы в следующих позициях – 1, 2.
2. Математические знаки. К этой группе были отнесены такие символы: *, +, -, /, =, <, ≤, >, ≥, %, ≈ , ∈ , э , ∆ , ∇ . Допустимо размещение символов во 2 позиции.
3. Скобки открывающие. В эту группу вошли следующие символы: (, {, [. Относительно этой группы символы могут располагаться во 2 позиции.
4. Скобки закрывающие. К этой группе относятся такие символы:), },]. Допустимое размещение символов относительно рассматриваемого символа: 1, 2 позиции.
5. Знаки препинания. К знакам препинания были отнесены: «.», «,», «!». Символы могут располагаться в 1, 2 позиции.
6. Интеграл ∫. Допустимое размещение символов во 2, 4, 8 позиции.
7. Извлечение корня √. Допустимое размещение: 1, 2, 6 позиции.
8. Сумма ∑. Допустимое расположение символов: 2, 4, 8 позиции.
9. Греческие и латинские буквы, рассмотренные на Рис.1. Допустимое размещение символов: 1, 2, 3 позиции.
10. Тригонометрические функции. Рассматриваются следующие тригонометрические функции sin, cos, tan/tg, ctg/ctn/cot. Допустимое размещение символов: 1, 2 позиции.
11. Логарифмические функции ln, lg, log. Допустимое размещение символов: 1, 2, 3 позиции.

Под допустимой позицией расположения символов понимается возможная, но не обязательная к заполнению символом позиция. Под термином «обязательная позиция» имеется в виду такая позиция, которая должна быть обязательно заполнена одним из следующих символов, такая позиция получит приоритет при добавлении нового символа. Например, знак ∫ или √ предполагает обязательное наличие зависимого символа в позиции 2.

Разработанный метод структурного анализа включает в себя следующие алгоритмы: размещения, чередования, группировки. Ниже в статье рассмотрим каждый из этих алгоритмов подробно.

Алгоритм размещения нового символа относительно других символов работает по следующему принципу.

1. Выражение перезаписывается слева направо каждый раз при добавлении нового символа.
2. Выбирается позиция для нового символа. А по следующему правилу:
   - Проверяется наличие пустой обязательной позиции, в случае ее наличия данная позиция получает приоритет и в нее помещается символ А.
   - Если пустая обязательная позиция отсутствует, то выбирается ближайшая допустимая позиция, в которую помещается новый символ А.
3. В случае, если позиция, в которую был помещен новый символ А, была занята символом Б, то выбирается новая позиция для символа Б по правилам описанным в пункте 2.

Стоит отдельно рассмотреть знаки «.» и «-». Так точка в выражении может быть десятичной запятой или оператором умножения в зависимости от положения точки и ее соседних символов. Горизонтальная линия может быть дробной чертой или знаком минус в зависимости от длины линии и наличия символов выше и ниже линии. В предлагаемом методе для решения такого рода неопределенности были использованы следующие подходы:

- Знак «-» будет рассматриваться как дробная черта, в том случае, если имеет символы в 4 и/или 8 позициях.
- Знак «.» рассматривается как оператор умножения, если центр масс его сегмента находится ближе к 2 позиции соседнего слева символа.

Рассмотрим работу алгоритма чередования. Несколько букв написанных вместе могут формировать единое значение. Это могут быть тригонометрические функции sin, cos, tan/tg, ctg/ctn/cot или логарифмические функции ln, lg,

log. Рассматривая группу букв написанных подряд алгоритм чередования проверяет, не образуют ли они название функции. В этом случае происходит чередование нескольких букв на одну из известных математических функций. Кроме того, алгоритм проверяет различные варианты написания, так, если на этапе распознавания одна из букв была распознана неправильно, на данном этапе ошибка будет исправлена, как показано на Рис. 3. На Рис.3 видно, что каждый раз при добавлении нового символа математическое выражение начинает анализироваться сначала, в данном случае буква «о» на этапе распознавания была ошибочно распознана как цифра «0». После добавления следующего символа алгоритм чередования просматривает все возможные комбинации, имеющиеся в базе, и, найдя наиболее подходящую, исправит ошибку.

Рассмотрим работу алгоритма группировки. Специальные символы, к ним были отнесены различного вида скобки, дробная черта и знаки препинания: точка, запятая, позволяют сгруппировать несколько отдельных значений в одну группу. В результате работы алгоритма группировки соответствующее оригиналу математическое выражение будет отображено наиболее верно.

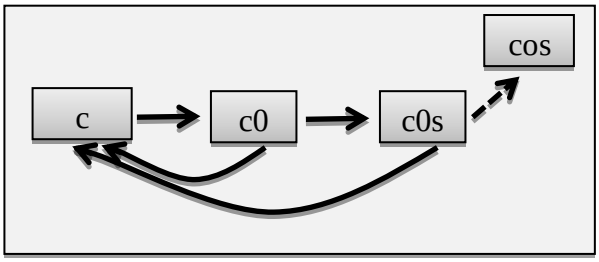


Рис. 3. Пример работы алгоритма чередования

Использование описанного метода позволяет определить пространственные отношения между символами и полностью отобразить структуру, написанного пользователем математического выражения. Кроме того полный анализ структуры математического выражения после добавления каждого нового символа делает систему устойчивой к внесению изменений.

Экспериментальные исследования

Для проверки эффективности работы рассматриваемого метода структурного анализа были проведены тесты в разработанном редакторе ввода рукописных математических выражений Рис. 4.

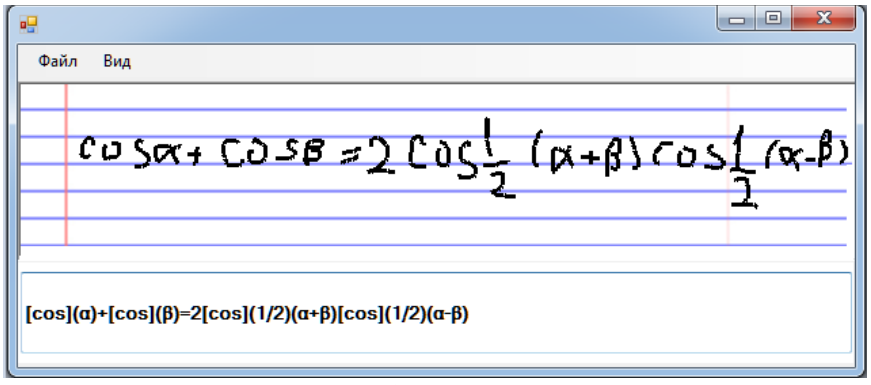


Рис. 4. Пример работы редактора ввода рукописных математических выражений

Для тестирования были отобраны 50 различных математических выражений, включающих в себя тригонометрические, логарифмические функции, операции возведения в степень, извлечения корня, дробного деления, индексы.

На Рис. 4 видно, что редактор хорошо структурирует достаточно сложное математическое выражение, при этом различает последовательность написанных символов в качестве названия тригонометрической функции, которую показывает в квадратных скобках.

В Таблице 1 приводятся результаты работы метода по количеству неправильно структурированных символов. Можно сказать, что такие символы как буквы и знаки операций имеют меньший процент ошибки, в отличие от логарифмических функций и знака суммы.

Табл. 1. Результаты работы метода структурного анализа

	Кол-во символов, содержащихся в протестированных математических выражениях	Кол-во неправильно структурированных символов	% неправильно структурированных символов
Цифры	371	14	6,46
Знаки операций	194	10	5,15
Скобки	92	8	8,69
Знаки препинания	17	3	17,6
Интеграл	25	3	12
Извлечение корня	23	2	8,69
Сумма	15	2	13,3
Греческие и/или латинские буквы	160	7	4,37
Тригонометрические функции	58	4	6,89
Логарифмические функции	15	2	13,3

На Рис. 5 приводится пример неправильно определенной структуры математического выражения. Здесь метод структурного анализа неправильно определил пространственные отношения между символами. Это ошибка могла произойти по причине слишком близкого расположения значения «2» к обязательному полю тригонометрической функции sin.

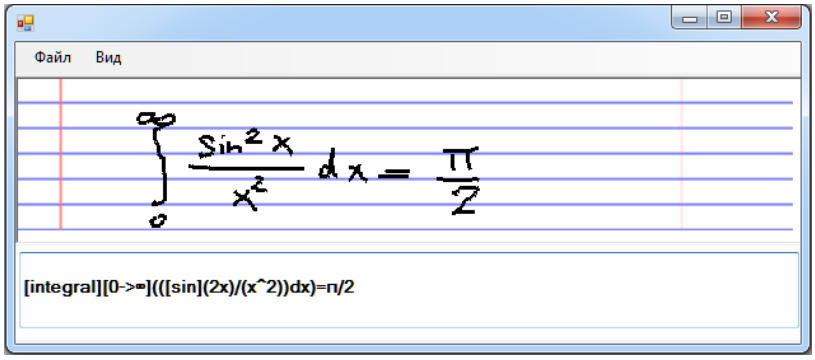


Рис. 5. Пример работы редактора ввода рукописных математических выражений

Табл. 2. Результаты тестирования

Общее кол-во символов, содержащихся в протестированных математических выражениях	Общее кол-во неправильно структурированных символов	% ошибочно структурированных символов
970	55	5,67

Как показано в Таблице 2, в результате тестирования было получено общее значение ошибочно структурированных символов равно 5,67% из проверочного набора данных. В других подходах значение ошибочно структурированных символов лежало в пределах 12%-20% [7]. Полученное значение свидетельствует о достаточно высокой эффективности использования предложенного в работе метода структурного анализа к работе с математическими выражениями любой сложности.

Заключение

В результате проведенного исследования был разработан метод структурного анализа, который позволяет достаточно точно определить пространственные отношения между символами и как результат полностью отобразить структуру, написанного пользователем математического выражения. Общее значение ошибочно классифицированных символов равное 5,67% свидетельствует об эффективности применения разработанного метода к поставленной задаче. Как показали исследования такие символы как логарифмические функции, знак суммы, знаки препинания имеют самый большой процент неправильно структурированных символов. Это можно объяснить наличием нескольких обязательных позиций у данных символов. Необходимо провести дальнейшие исследования, которые позволят улучшить рас-

смотренный метод структурного анализа. Улучшения будут достигнуты путем доработки алгоритма размещения. На данном этапе модель метода структурного анализа разработана с учетом правила, что пользователь может ввести только одно математическое выражение в редакторе. Для того чтобы пользователь мог ввести несколько математических выражений одновременно, метод должен быть усовершенствован введением дополнительных символов, которые будут разделять математические выражения. Кроме того необходимо разработать систему подсказок, которая позволит пользователю выбирать одну из предлагаемых в списке структур. Предлагаемые улучшения позволят снизить процент ошибочно классифицированных символов.

Список литературы

1. Anderson R. Syntax-directed recognition of hand-printed two-dimensional mathematics. - New York: Academic Press, 1968. – pp. 436-442.
2. Belaid A. Haton J.-P. A syntactic approach for handwritten mathematical formula recognition. - IEEE Transactions on Pattern Analysis and Machine Intelligence, 1984. - pp. 105-107.
3. Nakayama Y. A prototype pen-input mathematical formula editor. – EDMEDIA, 1993. - pp. 400-407.
4. Kosmala A. Rigoll G. On-line handwritten formula recognition using statistical methods. - International Conference on Pattern Recognition 1998 pp. 1306-1308.
5. Зайченко Ю.П. Нечёткие модели и методы в интеллектуальных системах. Учебное пособие для студентов высших учебных заведений. – К.: «Издательский Дом «Слово»», 2008. – с. 344
6. Зайченко Ю.П. Основы проектирования интеллектуальных систем. Учебное пособие. – К.: «Издательский Дом «Слово»», 2004. – с. 352
7. Mouchère H., Viard-Gaudin C., Kim D. H., Kim J. H., Garain U. CROHME2011: Competition on Recognition of Online Handwritten Mathematical Expressions. – IEEE, 2011. – pp. 1497-1500

УДК 519.6:512.972

МИНАЕВ Ю.Н,
КЛИМЕНКО И.А.,
ФИЛИМОНОВА О.Ю.,
МИНАЕВА Ю.И.

«МЯГКИЕ» ВЫЧИСЛЕНИЯ НА ОСНОВАНИИ МОДЕЛЕЙ КРОНЕКЕРОВОЙ (ТЕНЗОРНОЙ) АЛГЕБРЫ

Рассматриваются вопросы представления объекта в условиях неопределенности, в том числе представления нечеткой переменной в виде нечеткого множества в форме тензорной модели двух видов: Кронекерово (тензорное) произведение значения и функции принадлежности и многомерный массив. Показана рациональность применения тензорного подхода (базиса) к моделированию неопределенности, дающему возможность исключить при анализе функцию принадлежности. Приведены примеры реализации операций нечеткой математики в тензорном базисе как процедуры «мягких» вычислений.

We consider the questions of presentation of object in conditions of uncertainty, including fuzzy variable, present in the manner of the fuzzy set, in the form a tensor's models of two types: Kronecker (tensor) product a value by membership and multivariate array. We are Shown rationality of using an tensor approach (base) to modeling of uncertainty, enable exclude at the analysis a membership function. Examples to realization of operations fuzzy mathematicians as procedures "soft" calculations are cited.

Введение

Управление в условиях неопределенности представляет ключевую проблему современной теории и практики управления. Предложенная Л. Заде парадигма – теория нечетких множеств (ТНМ) – позволила решить ряд прикладных задач [1]. Обобщенное понятие неопределенности впервые рассмотрено в работах М. Блэка [2]. Функции принадлежности (ФП) позволили *объективно* представить *субъективно* определенные объекты и тем самым расширить возможности применения (принцип нечеткого расширения (ПНР)) стандартных методов и моделей к условиям неопределенности. Недостаточность ФП с точки зрения представления неопределенности была показана практически одновременно с созданием теории НМ.

В работах [3, 4] предложены т.н. *интуиционистские* НМ (*IFS*), определяемые на универсуме *U*: *IFS* над *U* – множество упорядоченных троек: элемент универсума, степень членства *M*, степень не-членства *N* так что $M + N \le 1$ и $M, N \in [0, 1]$. Когда M + N = 1 получаем НМ, и если M + N < 1 есть неопределенность [4], которая равна $I = 1 - M - N$. На основании упорядоченных троек *IFS* дополнительно предложены несколько обобщений НМ. В последнее время применение ТНМ расширилось настолько, что возникли сомнения в правомочности такого подхода. В работе [5] отмечено, что теория НМ « ... *наиболее успешно используется там и тогда, где и когда нечеткость порождается присутствием человека и его разума»*.

Предложенная Л. Заде ФП требует безусловного определения методов ее получения и операции над ними. В целом ряде случаев ФП получить или невозможно или ее эвристическая формулировка не отражает объективных условий явления или процесса. «Следствием этой недосказанности служит «зоопарк» операций над нечеткими множествами, за последние годы разросшийся в рамках этой теории до неприличных размеров…» [5]. В работах Г. Крона [6] показано, что представление объекта исследования (измерения) в виде тензора есть более адекватным, чем в виде величины. Тензорная модель позволяет анализировать объект в разных системах координат, понимая под системами не только декартовы (прямоугольные или косоугольные, прямолинейные или непрямолинейные) координаты, но и точки зрения. Является естественным и закономерным рассмотреть неопределенность, наложив минимальные ограничения на ее сущность, не прибегая к дополнительным эвристикам типа ФП или ПНР, на уровне тензорных моделей.

Современное состояние исследований

ТНМ как способ описания неопределенности имеет прямую связь с известной идеей Галилея о *координатизации.* Любые объекты, являющиеся предметом математического исследования: кривые, поверхности, отображения, величины и

другие могут быть «координатизированы» или «измерены». Но для такой координатизации «обычных» чисел, обычной стандартной (архимедовой) метрики, как показывает практика, в целом ряде случаев далеко не достаточно. Встречаясь с новым типом объектов, необходимо рассматривать их «в условиях неопределенности», для которой может потребоваться конструировать и новые типы «величин» (объектов), что их координатизируют, и новые метрики. Напомним, что обобщением понятия числа является *матрица*, которая в общем случае рассматривается как проекция тензора.

В работе [7] раскрыты особенности НМ как формы моделирования неопределенности. Отмечено, что переход от обычной характеристической функции множества к ФП есть совершенно естественным, однако, идея НМ как ФП не является единственной. Развитие ТНМ в связке с ФП не обязано идти в направлении безусловного использования ФП, так как в ряде ситуаций ФП построить невозможно или она такова, что не отражает действительной информации, заложенной в эмпирическом утверждении. Нечеткую математику в [7] рекомендуется рассматривать в виде системы образующих (возможно даже бесконечной), на которой заданы правила перехода от одних образующих к другим. Современные исследования неопределенности, моделируемой в форме НМ, показывают, что создание новых операций на основе аналогов классических – результат пересечения или объединения – не есть рациональным. Операции нечеткой математики должны определяться экспериментально. Завершая обзор работы [7], укажем, что современный поход к моделированию неопределенности состоит не в создании новых способов построения ФП, а в создании механизмов извлечения нечеткости, что, в свою очередь, приводит к новой концепции получения нечетких знаний.

В этой связи укажем, что тензор есть объектом, который может моделировать множество отдельных значений, в том числе нечеткий (неоднозначный) объект, не требуя при этом обязательного назначения ФП. При необходимости параметр, аналогичный ФП, всегда может быть приближенно вычислен.

Постановка основных задач

Проведенный анализ позволил определить обстоятельства, которые обусловили специфику постановки задач:

- неопределенность представляет собой сложный объект, предполагать возможность решения задач, опираясь исключительно на теорию НМ, что имеет место сегодня, абсолютно неправильно;
- теория НМ в современном виде игнорирует получение новых знаний, в т.ч. нечетких, на основании представленной информации.

Особенности постановок основных задач состоят в следующем:

- условия неопределенности ограничены следующими случаями: нечеткая переменная (НП) как элемент НМ- подмножества упорядоченных пар, интервал, числовой ряд; тензорное (матричное) представление объекта в условиях неопределенности (неоднозначности) для всех рассматриваемых случаев, что должно дать возможность получения новых знаний из рассматриваемых условий неопределенности (неоднозначности);
- выполнение арифметических операций при тензорном представлении объектов должно исключать возможность субъективных оценок или выводов;
- необходимо обеспечить инвариантность результатов по отношению к ФП или в общем случае отказ от использования ФП;
- исключить или по возможности ограничить применение эвристик при выполнении арифметических и логических операций.

Представление НП системой тензорных моделей

Наделение объекта (переменной) субъективной ФП создает новый (более сложный) объект – НП, чем тот, который задан первоначально, в этом достоинство и недостаток ТНМ. Тензорная модель создается на первоначальном этапе и не наделяет объект дополнительными свойствами. Моделирование объекта в тензорном базисе естественно требует соответствующего моделирования арифметических (алгебраических) операций над ними и должно включать этапы:

- визуализация тензор-переменных (ТП), определение рациональной формы модели (целесообразность и возможность использования тензоров низких рангов);
- визуализация алгебраических операций над тензор-переменными и их инвариантами;
- определение связи между результатами, полученными в тензорном базисе, и результа-

тами, полученными при помощи ПНР на стандартных НМ.

Моделирование операций нечеткой математики должно включать определенные предостережения, если они выполняются в определенной модельной среде, в частности, учитывать особенности системы математического моделирования *Matlab*. Это касается, прежде всего, главных определений и способов реализации операции. В системе математического моделирования *Matlab* не проводится различия между НЧ и НП, хотя, как показано в [8], между ними есть существенное различие, но использование системы заставляет игнорировать это обстоятельство. Кроме того, нечеткая арифметика построена на эвристическом принципе нечеткого расширения [4], который не определяет величину, обратную НП или НЧ.

ПНР позволил перенести все операции четкой арифметики на нечеткую. Его можно рассматривать как принцип *инвариантности* арифметических операций относительно замены типа переменной (четкая ⇔ нечеткая). Но некоторые понятия в условиях неопределенности кроме математического содержания должны получать, и получают на практике дополнительные ударения, это касается, в частности, понятия расстояния.

Нечеткая математика, как показано в работах [9, 10], эффективно реализуется в тензорном базисе, если ввести аналог НП – тензор-переменную, определяемую как *Kronecker*'ово (тензорное) произведение пар «значение – ФП» и вычисляемую на универсальном множестве $E\{x\}$, где $x \in E$, или только на «усеченной» совокупности упорядоченных пар $\{(x, \mu_{\tilde{A}}(x))\}$, $\forall x \in E$, x∈E. Где $\mu_{\tilde{A}}(x)$ – ФП, принимающая свои значения на множестве *M* значений ФП и определяющая степень принадлежности *x* в $\tilde{A}$, то $\tilde{A}$ – нечеткое подмножество множества *E*, $\mu_{\tilde{A}} \to [0,1]$. Например, для ФП типа $trimf\ (x,[a\ b\ c])$ можно определить ТП на УМ или только на множестве значений $[a\ b\ c] \in E$.

Будем рассматривать неопределенность как ситуацию, в которой присутствует неоднозначность, в частности, ограничимся тем случаем, когда измерение (представление) состояния объекта исследования или его отдельных характеристик возможно в виде НМ, включая нечеткий интервал и высказывания естественного языка типа «больше», «меньше», «близко к ...», «приблизительно одинаково ... » и др.

Предложенные случаи необходимо представить в унифицированном виде, позволяющем применять стандартную математику, минимально требующую дополнительных расширений, основанных на эвристических «правдоподобных» предположениях [11]. Учитывая неизбежную «мягкость» вычислений, желательно предусмотреть минимальный контроль точности.

Уточним рассматриваемые объекты в нотации ТНМ:

НП с произвольной ФП –

$$\tilde{x}_{\text{var}} = \{x_i / \mu_i^x\},\ i = 1, n;$$
$$\tilde{x}_{\text{var}} = \{x_1 / \mu_1^x, x_2 / \mu_2^x, ..., x_n / \mu_n^x\}; \quad (1)$$

НП с треугольной ФП –

$$\tilde{x}_{trimf} = \{x_i / \mu_i^x\} = \{x_1 / 0, x_2 / 1, x_3 / 0\};$$

интервал –

$$^x I = [\underline{x}, \overline{x}] = [x^{\min}, x^{\max}] = [x_1, x_2];$$

нечеткий интервал –

$$^x \tilde{I} = [x_i / \mu_i^x],\ \text{i} = 1, 4;$$
$$^x \tilde{I} = [x_1 / 0, x_2 / 1, x_3 / 1, x_4 / 0].$$

В соответствии с работой [12] тензор это многомерный массив. Более формально, *N*-мерный или *N*-порядковый тензор – элемент тензорного произведения *N* векторных пространств, каждое из которых имеет *собственную систему координат*. Это понятие не противоречит понятию тензора в физике (тензорное поле) и в инженерном проектировании (тензор мощности, например, в смысле Г. Крона [6]). Тензор третьего порядка имеет три индекса, тензор второго порядка – два индекса (матрица), однопорядковый тензор – один индекс (вектор), тензоры порядка три и выше названы высокопорядковыми тензорами.

Представим НП с произвольной ФП (1) в виде двух массивов одномерных векторов $x = \{x_1, x_2, ..., x_n\}$ и $\mu_x = \{\mu_1^x, \mu_2^x, ..., \mu_n^x\}$. ТП определена как $T_x = x \otimes \mu_x^T$, где индекс ^{T} – символ транспонирования. ТП с матрицами 9 × 9 и 3 × 3, соответствующие НП *примерно 5* с треугольной ФП, приведены ниже: *TDt* – ТП, определенная на УМ, *TDts* – усеченная ТП, *TDts*0 – свертка.

ТП *TDt.x* = [0:10/8:10];

muxDt = *trimf* (*x*, [1 5 7]);

TDt = *kron* (*muxDt*, *x*');

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.08 & 0.47 & 0.86 & 1.25 & 0.47 & 0 & 0 & 0 \\ 0 & 0.16 & 0.94 & 1.72 & 2.50 & 0.94 & 0 & 0 & 0 \\ 0 & 0.23 & 1.41 & 2.58 & 3.75 & 1.41 & 0 & 0 & 0 \\ 0 & 0.31 & 1.88 & 3.44 & 5.00 & 1.88 & 0 & 0 & 0 \\ 0 & 0.39 & 2.34 & 4.30 & 6.25 & 2.34 & 0 & 0 & 0 \\ 0 & 0.47 & 2.81 & 5.16 & 7.50 & 2.81 & 0 & 0 & 0 \\ 0 & 0.55 & 3.28 & 6.02 & 8.75 & 3.28 & 0 & 0 & 0 \\ 0 & 0.63 & 3.75 & 6.88 & 0.00 & 3.75 & 0 & 0 & 0 \end{pmatrix}$$

x = [0:10/8:10];

TDts = *kron* ([1 5 7], [0 1 0]');

$$TDts = \begin{pmatrix} 0 & 0 & 0 \\ 1.0 & 5.0 & 7.0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$TDts\ 0 = свертка\ (TDt) = \begin{pmatrix} 0.3385 & 0.7292 & 0 \\ 0.8854 & 3.3073 & 0 \\ 1.4323 & 5.8854 & 0 \end{pmatrix}$$

Нормы:

[*norm* (*TDts*, '*fro*') *norm* (*TDts*0, '*fro*')]

7.0042 8.6603

Отметим основные особенности ТП *TDt*, *TDts*, *TDts*0. Хотя они моделируют один и тот же объект – НП *примерно 5* с треугольной ФП, но у них практически совпадают только первые инварианты, все остальные характеристики разные, тензоры *TDts*, *TDts*0 не подобны, но эквивалентны, т.к. относятся к одному и тому же объекту. Поэтому при использовании тензорных моделей этот факт необходимо учитывать.

Собственные значения:

*TDts*0 = [0 3.5108 0.135]^T,

TDts =[5.0 0 0]^T.

Инварианты:

I_1 = 0 +3.5108+0.135=3.6458, I_1 = 5.0+0+ 0=5;

I_2 = 3.5108* 0.135=0.474, I_2=0;

I_3 = 0, I_3=0.

НП с Гауссовой, треугольной и трапециевидной ФП и их тензорные аналоги, представленные в виде диадных тензоров с матрицами 9×9 и 3×3, приведены на рис. 1 и рис. 2.

Математический объект $T_x = x \otimes \mu_x^T$, $\mu_x \to [0,1]$, называемый *тензор-переменной,* позволяет получить дополнительные знания, которые могут быть извлечены из нечеткости и использованы как результат структурирования

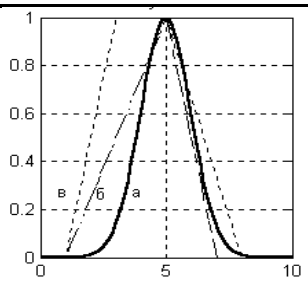


Рис. 1. Анализируемые ФП: а) Гауссова,б) треугольная, в) трапециевидная

неопределенности (знания, полученные из нечеткости) в виде алгебраической системы.

Прежде всего, отметим, что тензор, получаемый в результате *Kronecker*'ова произведения, является так называемым *диадным тензором* [13], матрица которого всегда является квадратной размерностью $n\times n$ (n – размерность векторов из тензорного произведения), все инварианты диадного тензора (кроме первого) равны нулю. Напомним, что 1-й инвариант (след) для матрицы

$$A = \begin{pmatrix} a_{11} & a_{1n} \\ & \ddots \\ a_{n1} & a_{nn} \end{pmatrix}$$

равен: $\mathrm{Tr}A = \sum_{i=1}^{n} a_{ii} = \sum_{i=1}^{n} \lambda_i$, где λ_i – собственные значения матрицы A.

Каждый тензор может быть разложен на симметричную A^{sym} и кососимметричную части A^{skw}:

$$A^{sym} = \frac{1}{2}(A + A^T),\ A^{skw} = \frac{1}{2}(A - A^T),$$

где $A = A^{skw} + A^{sym}$; девиаторную A^{dev} и изотропическую A^{iso} части:

$$A^{dev} = A - \frac{1}{3}(TrA)I,\ A^{iso} = \frac{1}{3}(TrA)I,$$

где I – тензор идентичности, в базисной нотации $I = \delta_{ij} e_i e_j$, δ_{ij} – символ *Kronecker'a*, $A = A^{iso} + A^{dev}$. Отметим, что A^{skw}, A^{sym} могут рассматриваться как тензорные функции, девиаторная A^{dev} и изотропическая A^{iso} части тензора представляют собой вариабельную и константную части тензора A, что особенно актуально в условиях моделирования неопределенности.

Известно, что в теории и практике применения НМ особую роль играют треугольные и трапециевидные ФП в силу их простоты и прозрачности применения. Тензорные модели НП с треугольными ФП представляют собой тензоры с матрицами 3×3. Для приближенных вычис-

лений, которые составляют основу нечеткой математики, прибегают к так называемой *свертке* тензоров,

$$\left(\text{A} \right)_n^n \to \left(\text{B} \right)_m^m, \; m < n .$$

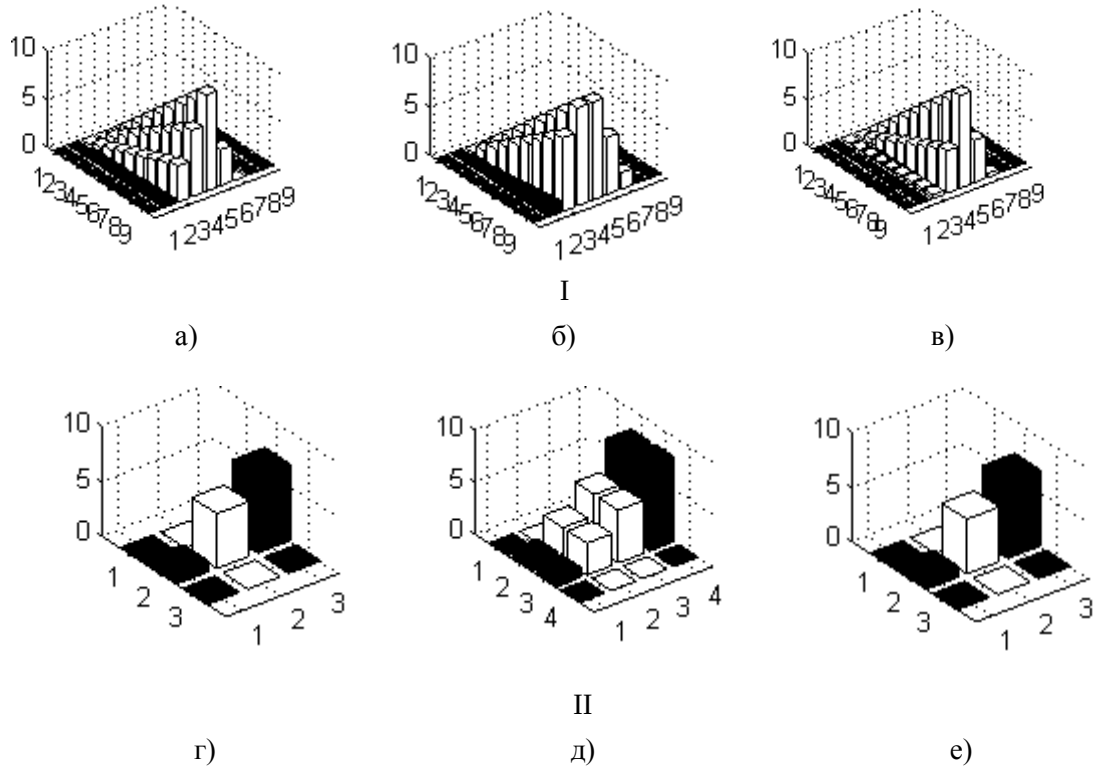


Рис. 2. Тензорные аналоги НП с ФП: I – тензоры с матрицей 9 × 9, II – тензоры с матрицей 3 × 3; а), г) – треугольная ФП; б), д) – трапециевидная ФП; в), е) – гауссова ФП

В соответствии с работой [14] операция получения тензора $b_{k\ldots m}$ из тензора $a_{ijk\ldots m}$ называется *свертыванием* тензора $a_{ijk\ldots m}$ по индексам i и j. Точно так же можно определить свертывание тензора $a_{ijk\ldots m}$ по любой другой паре индексов. При каждом свертывании тензора его валентность понижается на две единицы. Например, при свертывании двухвалентного тензора получают тензор a_{ii} нулевой валентности, то есть инвариант, являющийся следом тензора $[a_{ij}]$, $a_{ii} = tr([a_{ij}])$.

В отличие от диадного тензора с матрицей 3×3, который имеет только один ненулевой инвариант, свернутый тензор с такой же матрицей, имеет все три ненулевых инварианта, которые через собственные значения тензора представляются в виде:

$I_1 = \lambda_1 + \lambda_2 + \lambda_3$, $I_2 = \lambda_1\lambda_2 + \lambda_2\lambda_3 + \lambda_3\lambda_1$, $I_3 = \lambda_1\lambda_2\lambda_3$, или в следовой форме

$$Tr(A) = I_1, \; Tr(A^2) = I_1^2 - 2I_2,$$

$$Tr(A^3) = I_1^3 - 3I_1I_2 + 3I_3 .$$

Операция свертывания двух тензоров состоит в их умножении и свертывании полученного в результате умножения тензора по индексам, принадлежащим разным сомножителям. В результате свертывания тензоров валентностей p и q получается тензор валентности $(p + q - 2)$ (так называемое умножение со сверткой). Отметим, что свертывание тензоров можно выполнять по любому количеству r таких пар индексов. В результате этого свертывания получается новый тензор, валентность которого на $2r$ единиц меньше суммы валентностей исходных тензоров.

Тензорное представление НП, заданных в естественном виде. Объекты исследования можно также представлять в виде столбцов таблицы, например:

$$\tilde{\text{x}} = \{\text{x}_j/\mu_j^{\text{x}}\}, \to [\text{x}_1 \quad \mu_1^{\text{x}}; \; \text{x}_2 \quad \mu_2^{\text{x}}; \cdots \; \text{x}_J \quad \mu_J^{\text{x}}] \to$$

$$\to \begin{pmatrix} \text{x}_1 & \mu_1^{\text{x}} \\ \text{x}_2 & \mu_2^{\text{x}} \\ \cdots & \\ \text{x}_J & \mu_J^{\text{x}} \end{pmatrix}, \mu_j^{\text{x}} \to [0,1],$$

такое представление будем называть *естественным*, $\tilde{x} = \{x_j / \mu_j^x\} \to tx = []_2^J$. Для НП со ста-

ндартной треугольной ФП (без учета УМ) имеем: $\tilde{X}_{trimf} = (3\ \ 0; 5\ \ 1; 7\ \ 0)$. Способ образования ТП приводит к тому, что в первом случае (диадный тензор) и во втором (естественное представление НП), матрица тензора $\boldsymbol{A}$ вырождена (или не является квадратной) и обратная матрица не существует, поэтому следует использовать *псевдообратную* матрицу $\boldsymbol{A}^+$, которая определяется как такая матрица $\boldsymbol{A}^+$, что $\boldsymbol{AA}^+\boldsymbol{A} = \boldsymbol{A}$. Псевдобратная матрица – не единственная и ее вид зависит от способа построения, доминирование псевдообратной матрицы в тензорных моделях неопределенности предопределяет «мягкость» всех вычислительных процедур.

Эквивалентные и подобные матрицы. Две прямоугольные матрицы $\boldsymbol{A}$ и $\boldsymbol{B}$ одной размерности $I \times J$ *эквивалентны*, если существуют такие квадратные матрицы $\boldsymbol{S}$, размерности $I \times I$, и $\boldsymbol{T}$, размерности $J \times J$, что $\boldsymbol{B} = \boldsymbol{SAT}$.

Эквивалентные матрицы имеют один и тот же ранг. Две прямоугольные матрицы $\boldsymbol{A}$ и $\boldsymbol{B}$ одной размерности $N \times N$ ***подобны***, если существует такая невырожденная матрица $\boldsymbol{T}$, что $B = T^{-1}AT$. Матрица $\boldsymbol{T}$ называется преобразованием подобия. Подобные матрицы имеют один и тот же ранг, след, определитель и спектр (одинаковые инварианты, одинаковые характеристические полиномы).

В работе показано, что матрицы ТП, полученные в результате реализации ТП $T_x = x \otimes chf^{\,T}$, где x – усеченное подмножество УМ, $chf^{\,T}$ – характеристическая функция (левый столбец табл.) и $T_x = x \otimes \mu_x^T$, $\mu_x \to [0,1]$ – правый столбец таблицы, являются такими, что все инварианты, кроме первого, имеют нулевые значения (отметим, что необходимое условие подобия состоит в совпадении характеристических полиномов матриц A и B).

В ТНМ для НМ $\tilde{A}$, $\tilde{B}$ определены расстояния [4]: линейное (Хэмминга) и квадратичное (Евклида). В *Kronecker*'овой алгебре определена норма *Hilbert-Schmidt* (также называемая нормой *Frobenius'a*), которая определяется как $\| X - Y \|^2 = \mathrm{Tr}((X - Y)(X - Y)^T)$, где X, Y – матрицы на $\boldsymbol{R}$ размерностью $n \times m$. Эта проблема известна как проблема *ближайшего Kronecker'ова произведения*, в работе норма *Frobenius'a* принимается как расстояние между матрицами.

Алгебраические операции в *Kronecker'*-овой алгебре

Прежде всего, коротко представим определения *Kronecker*'овых операторов произведения и суммы. Для их детального математического описания следует обратиться к работе [15]. Пусть

$$A = \left(a_{ij} \right) \text{ и } B = \left(b_{ij} \right)$$

две матрицы порядка $n1 \times n2$ и $m1 \times m2$ соответственно. *Kronecker*'ово произведение $C = A \otimes B$ есть таким, что:

$$C = \begin{pmatrix} a_{11}B & \cdots a_{1n_2}B \\ \cdots\cdots & \cdots \\ a_{1n1}B & \cdots a_{n1n_2}B \end{pmatrix}$$

Тензорная сумма определена только для квадратных матриц, тензорная сумма $A \otimes B$ двух квадратных матриц A и B определена в терминах тензорного произведения как: $A \otimes B = A \otimes I_m + \ldots + I_n \otimes B$, где n – порядок матрицы A, m – порядок матрицы B, $\boldsymbol{I}_n$ и $\boldsymbol{I}_m$ – матрицы идентичности порядка n и m соответственно, «+» – обычный оператор матричного сложения. Прямая сумма тензоров (матриц тензоров) в блок-диагональной форме имеет вид:

$$\mathrm{A} \oplus \mathrm{B} = \begin{pmatrix} (\mathrm{A})_n^n & (0)_m^n \\ (0)_n^m & (\mathrm{B})_m^m \end{pmatrix} =$$

$$= \begin{pmatrix} \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} & \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix} \end{pmatrix}.$$

Детерминант прямой суммы –

$$\det(A \otimes B) = (\det A)(\det B),$$

след прямой суммы –

$$\mathrm{Tr}(A \otimes B) = (\mathrm{Tr}A)(\mathrm{Tr}B).$$

Детерминант тензорного произведения –

$$\det(A \otimes B) = \det(A)^m \det(B)^n$$

след тензорного произведения –

$$\mathrm{Tr}(A \otimes B) = (\mathrm{Tr}A)(\mathrm{Tr}B).$$

Часто возникает необходимость работы с ТП, у которых матрицы содержат только по одному ненулевому элементу, например,

$$g_1 = \begin{pmatrix} 0 & 0 \\ 0 & g_{i1} \end{pmatrix} \text{ и } g_2 = \begin{pmatrix} 0 & 0 \\ 0 & g_{i2} \end{pmatrix}.$$

Тензорная сумма $(g_1 \otimes g_2)$ в этом случае определяется таким образом:

$$\begin{pmatrix} 0 & 0 \\ 0 & g_{i1} \end{pmatrix} \oplus \begin{pmatrix} 0 & 0 \\ 0 & g_{i2} \end{pmatrix} =$$

$$= \begin{pmatrix} 0 & 0 \\ 0 & g_{i1} \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 \\ 0 & g_{i2} \end{pmatrix} =$$

$$= \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & g_{i2} & 0 & 0 \\ 0 & 0 & g_{i1} & 0 \\ 0 & 0 & 0 & g_{i1} + g_{i2} \end{pmatrix}.$$

Двойная свертка полученного выражения дает $(g_1 + g_2)$, то есть совпадает с естественным представлением. Матричное исчисление делает использование оператора векторизации матриц *vec* и оператора *Kronecker'*ова произведения очень важным. Оператор *vec* векторизует матрицу, «складывая» ее в столбец (при соглашении, что колонка предшествует строке). Пример рассмотренный алгебраических преобразований в *Kronecker'*овой алгебре представлен далее.

Пример.

Заданы две НП: <*примерно 5*> $\to \tilde{5} \underline{\underline{\Delta}} \{3/0, 5/1, 7/0\}$ и

<*примерно 7*> $\to \tilde{7} \underline{\underline{\Delta}} \{3/0, 5/1, 7/1, 9/0\}$, где $\underline{\underline{\Delta}}$ – равно по определению.

Вычислить в тензорном базисе:

<*примерно 5*> $\oplus$ <*примерно 7*> и <*примерно 5*> $\otimes$ <*примерно 7*>.

1^0. Определяем ТП – аналоги НП <*примерно 5*> и <*примерно 7*>:

$$\tilde{5} = \{3/0, 5/1, 7/0\} \to \mathrm{T}^{(5)} = [3\ 5\ 7] \otimes [0\ 1\ 0]^{\mathrm{T}} = \begin{pmatrix} 0 & 0 & 0 \\ 3 & 5 & 7 \\ 0 & 0 & 0 \end{pmatrix},$$

$$\tilde{7} = \{3/0, 5/1, 7/1, 9/0\} \to \mathrm{T}^{(7)} = [3\ 5\ 7\ 9] \otimes [0\ 1\ 1\ 0]^{\mathrm{T}} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 3 & 5 & 7 & 9 \\ 3 & 5 & 7 & 9 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

$$^0.\ T^{(5)} \oplus T^{(7)} = C_{\mathrm{c}} = \begin{pmatrix} 0 & 0 & 0 \\ 3 & 5 & 7 \\ 0 & 0 & 0 \end{pmatrix} \oplus \begin{pmatrix} 0 & 0 & 0 & 0 \\ 3 & 5 & 7 & 9 \\ 3 & 5 & 7 & 9 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} \left(\mathrm{T}^{(5)}\right)_3^3 & (0)_4^3 \\ (0)_3^4 & \left(\mathrm{T}^{(7)}\right)_4^4 \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} 0 & 0 & 0 \\ 3 & 5 & 7 \\ 0 & 0 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 & 0 \\ 3 & 5 & 7 & 9 \\ 3 & 5 & 7 & 9 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{pmatrix}.$$

$$3^0.\ T^{(5)} \oplus T^{(7)} = C_{\text{пр}} = \begin{pmatrix} 0 & 0 & 0 \\ 3 & 5 & 7 \\ 0 & 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 & 0 & 0 \\ 3 & 5 & 7 & 9 \\ 3 & 5 & 7 & 9 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 9 & 15 & 21 & 27 & 15 & 25 & 35 & 45 & 21 & 35 & 49 & 63 \\ 9 & 15 & 21 & 27 & 15 & 25 & 35 & 45 & 21 & 35 & 49 & 63 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Свойства тензора, остающиеся неизменными при преобразованиях координат, определяются системой его инвариантов, которые зависят от коэффициентов характеристического уравне-

ния. Инварианты это константы, значения которых сохраняются при смене системы координат. Возможность замены анализа неопределенности, представленной тензором, анализом его инвариантов, открывает новые пути для решении задач управления в условиях неопределенности. Отметим, что значения следов

$(C_{\text{с}} = T^{(5)} \oplus T^{(7)}$, $\text{Tr}\, C_{\text{с}} = 17$) и ($C_{\text{пр}} = T^{(5)} \oplus T^{(7)}$, $\text{Tr}\, C_{\text{пр}} = 60$)

совпадают с дефадзифицированным значением суммы и произведения НП <*примерно* 5> и <*примерно* 7>, то есть

$$\tilde{5} +_f \tilde{7} = 1\tilde{2}\,,\ \tilde{5} *_f \tilde{7} = 6\tilde{0}\,.$$

В общем случае для НП $\tilde{x} = \{x_i / \mu^{x_i}\}_{i=1}^{n}$ тензорное произведение векторов $[\text{x}_\text{i}] \otimes [\mu^{x_i}]$ обладает свойством, что след $\text{tr}([\text{x}_\text{i}] \otimes [\mu^{\text{x}_\text{i}}])$ совпадает с дефадзифицированным значением НП $\tilde{x}$, $def\ (\tilde{x}) = \sum_{\text{i}=1}^{\text{n}} \text{x}_\text{i} \mu^{\text{x}_\text{i}} \sum_{\text{i}=1}^{\text{n}} \mu^{\text{x}_\text{i}} = 1$.

Выводы

Предложено представлять объект в условиях неопределенности, в т.ч. НП, тензорными моделями, формируемыми как тензорное (Кронекерово) произведение компонент НП (значение ⊗ ФП) или как массив ”значение - ФП ”, размерностью $2\times n$, где n – количество α-уровней в представлении НП или количество значений, которыми представляется объект. Предложенный поход позволяет, с одной стороны, рассматривать объект в условиях неопределенности без обязательного использования ФП, с другой, позволяет все операции над НП выполнять на уровне формальных моделей матричного исчисления без необходимости применять разнообразные эвристические приемы и принципы типа принципа нечеткого расширения для пояснения и неформального обоснования применяемых методов и моделей.

Список литературы

1. Zadeh L.A. Fuzzy algorithms // Information and Control. – 1968 – №12. – P. 94 – 102.
2. Блэк М. Метафора // Теория метафоры. – М.: Прогресс, 1990. – С. 153 – 172.
3. Atanassov, K. Intuitionistic fuzzy sets // Fuzzy Sets and Systems. – 1986. – №20 – P. 87 – 96,
4. Atanassov K. More on intuitionistic fuzzy sets // Fuzzy Sets and Systems. – 1989. – №33. – P. 37 – 46.
5. Воробьев О.Ю. Эвентология – очеловеченная математика [Электронный ресурс] / Институт вычислительного моделирования СО РАН. – Красноярский государственный университет, 2005. – Режим доступа: http://eventology-theory.ru /000.htm.
6. Крон Г. Тензорный анализ сетей. – М. : Сов. радио, 1978. – 720 с.
7. Поспелов Д.А. Из истории развития нечетких множеств и мягких вычислений в России [Материалы круглых столов, проведенных Д.А.Поспеловым], (Москва, 1991, 1998); с послесловием В.Б.Тарасова // Новости Искусственного Интеллекта. – №2-3. – М. : ВЦ РАН, 2001. – 36 с.
8. Кофман А. Введение в теорию нечетких множеств.– М.: Радио и связь,1982.– 432 с.
9. Минаев Ю.Н. Нечеткая математика на основе тензорных моделей неопределенности. Часть 1 – тензор-переменная в системе нечетких множеств / Ю.Н. Минаев, О.Ю. Филимонова // Электронное моделирование. – К. : ИПМЭ НАН Украины, 2008. – № 1, т.30 – С. 43 – 59.
10. Минаев Ю.Н. Нечеткая математика на основе тензорных моделей неопределенности. Часть 2 – нечеткая математика в тензорном базисе / Ю.Н. Минаев, О.Ю. Филимонова // Электронное моделирование. – К. : ИПМЭ НАН Украины, 2008. – № 2, т.30 – С. 4 – 21.
11. Пойа Д. Математика и правдоподобные рассуждения / Д. Пойа; пер. с англ. И.А. Вайнштейна; под ред. С.А. Яновской. – М. :Издательство «НАУКА», 1975. – 465 с.
12. Kolda T.G. Tensor Decompositions and Applications [Siam Review] / T.G. Kolda, B.W.Bader // Society for Industrial and Applied Mathematics [Электронный ресурс]. – Vol. 51, № 3 – P. 455 – 500. – Режим доступа: http://www.siam.org/journals/sirev/51-3/70111.html.
13. Brannon R.M. Elementary and Intermediate // Vector and Tensor Analysis [UNM REPORT]. – Albuquerque : University of New Mexico, 2002. – 204 p.
14. Акивис М.А. Тензорное исчисление: учеб. пособие / М.А. Акивис, В.В. Гольдберг. – 3-е изд., перераб. - М. : ФИЗМАТЛИТ, 2003. – 304 с.
15. Graham A. Kronecker product and matrix calculus with applications / A. Graham. – USA, NY : Wiley, 1981. – 130 p.

УДК 519. 178

КУЗЬМУК А.В.,
КУЗЬМУК В.В.,
СУПРУНЕНКО О.О.

ПРИМЕНЕНИЕ УПРАВЛЯЮЩИХ СЕТЕЙ ПЕТРИ ДЛЯ МОДЕЛИРОВАНИЯ ПАРАЛЛЕЛЬНЫХ ПРОЦЕССОВ С МНОГОВАРИАНТНЫМ ВЫБОРОМ

В статье рассмотрена интерпретация управляющих сетей (SN), которая построена на основе безопасных сетей Петри (SPN). На примере задачи об эффективном размещении финансовых ресурсов продемонстрированы возможности SN, проведено сравнение с моделями на основе безопасных сетей Петри.

The article discusses the interpretation of the Control Networks (SN), which is based on safe Petri nets (SPN). In the example of the efficient allocation of resources demonstrated the possibility of SN, compared with models based on safe Petri nets.

1. Введение

В статье рассмотрены управляющие сети Петри с функциональными расширениями, позволяющими моделировать управление асинхронными параллельными потоками при многовариантном выборе в вершинах мест и вершинах переходов, а также при соблюдении запрещающих условий.

2. Моделирование сложных управляющих алгоритмов на основе сетей Петри

При разработке программного обеспечения для многопроцессорных и многомашинных комп'ютерных систем возникает задача оценки эффективности его работы. Для её решения часто применяют имитационное моделирование [1]. При выборе средства имитационного моделирования руководствуются рядом требований, таких как [2] возможность проверки работоспособности, возможность адаптации архитектуры к конкретной вычислительной системе, отображение корректного формирования и взаимодействия параллельных потоков, возможность отслеживания тупиковых ситуаций, возможность проверки корректности реализации задачи в параллельном алгоритме.

Для удовлетворения вышеназванных требований в теории сетей Петри разработаны наборы статических и динамических свойств [3], которые позволяют отслеживать топологию модели на этапе построения, устранять свойства небезопасности и конфликтности, определять неживые участки сети Петри при имитационном моделировании.

Перспективными с точки зрения изобразительной мощности являются управляющие сети Петри (SN), которые основываются на свойствах безопасной интерпретации сетей Петри (SPN) [4-5] с применением ингибиторных дуг [4, 10] и управляющих векторов для моделирования параллельных процессов с многовариантным выбором [6]. Такие задачи возникают при моделировании логистических задач, задач об эффективном размещении, задач об оптимальном использовании ресурсов [1, 7] и многих подобных задач.

3. Постановка задачи об эффективном размещении ресурсов

При формулировании задачи об эффективном размещении ресурсов важно задать ограничения, налагаемые внутренними и внешними обстоятельствами, а также начальные ресурсы и их распределение.

Задача состоит в оптимальном размещении свободных финансовых ресурсов (пакетов ресурсов) на депозитных вкладах в нескольких банках. Размещения проводиться на календарный год. Критерием оптимальности является максимизация финансовых ресурсов при минимизации рисков за заданный период времени.

При решении задачи должны быть учтены размеры пакетов, условия депозитов, риски от вложения финансовых ресурсов и предусмотрены алгоритмы минимизации рисков. Кроме того, по условию задачи вложения двух финансовых пакетов в один банк необходимо исключить.

4. Применение управляющих сетей петри в модели эффективного размещения ресурсов

При создании моделей систем управления параллельными процессами с неравномерным

распределением условий и ресурсов неизбежно возникает необходимость обработки вариантов решений и выбора наилучшего по установленному критерию решения для реализации дальнейших действий.

В случае построения модели поставленной задачи на основе безопасных сетей Петри возникают трудности с отображением и проверкой функционирования алгоритмов выбора альтернативного варианта из множества доступных. Так на рис. 1 представлена модель решения поставленной задачи для трёх финансовых пакетов (p_1, p_2, p_3) ресурсов вкладчика ***B***, которые могут быть размешены в трёх банковских учреждениях (A_1, A_2, A_3).

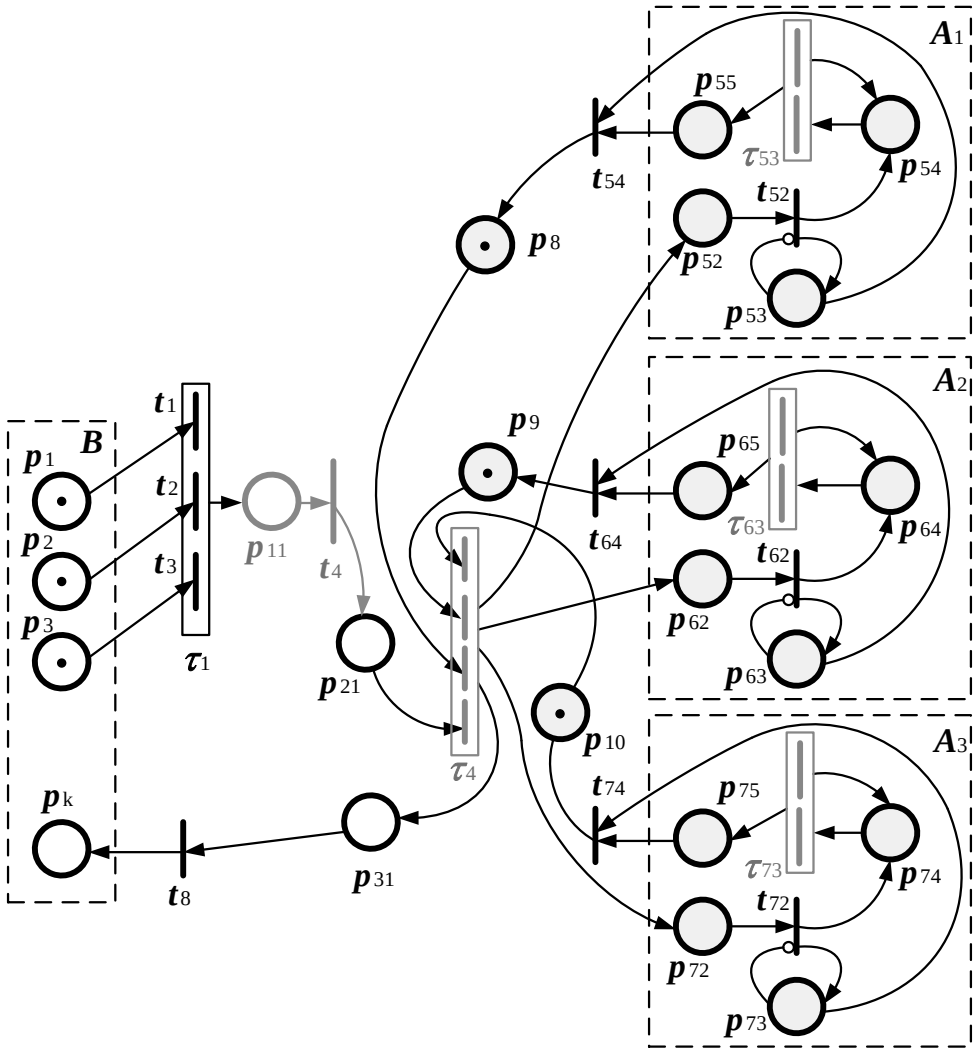


Рис. 1. Модель оптимизации размещения финансовых ресурсов на депозитах, представленная безопасной сетью Петри

На первом этапе построения модели необходимо отобразить конструкцию, которая позволит выбрать наибольший финансовый пакет и подобрать для него наилучшие условия размещения. Дальнейшие действия по размещению оставшихся пакетов должны быть отображены с использованием тех же алгоритмов, что и для первого пакета. Для отображения таких алгоритмов в безопасных сетях Петри необходимо применить макропереходы τ_i [6], которые позволяют производить совместную обработку условий и осуществлять перекрещивание потоков действий.

Начальными (входными) вершинами в модели задачи являются p_1, p_2, p_3, которые содержат финансовые пакеты (метки), предоставленные для вложений. Пакеты концентрируются через соответствующие переходы t_1, t_2, t_3 в макропереходе τ_1 и последовательно поступают в вершину места p_{11}, далее для составления списка передаются в вершину перехода t_4, где и проводиться выбор наибольшего. Данный пакет (метка) поступает в макропереход τ_4, в котором проводиться анализ поступившей от банков информации по депозитам с целью вы-

бора наилучших условий для получения максимальной прибыли при минимальных рисках:

$$F(x,y) = \max_i \frac{D_i(x,y) + S_i(x,y)}{R_i(x,y)}$$

где $F(x,y)$ – планируемая прибыль от вложения финансового пакета размером x на время y, $D_i(x,y)$ – планируемая прибыль по депозиту за расчётный период y, $S_i(x,y)$ – дополнительная планируемая прибыль по депозиту за расчётный период y при соблюдении дополнительных условий, $R_i(x,y)$ – убытки с учётом рисков по депозиту за расчётный период y [8].

После срабатывания макроперехода τ_4, метки передаются в вершины мест p_{52}, p_{62}, p_{72}, p_{31}. Информация, которая переноситься этими метками, будет различной, поскольку пакет может быть вложен только в один банк, в остальные банки приходят метки с нулевой суммой, и в конечную вершину p_k также, поскольку до совершения операций, в ней не должны появиться финансовые пакеты. Метки с нулевыми суммами сжигаются в вершинах переходов t_{i2} и t_8.

Допустим, наибольшее значение функции $F(x)$ получено для банка A_1, тогда метка, которая будет передана в p_{52}, содержит сумму вклада, а метки, поступившие в вершины p_{62}, p_{72}, p_{31}, нулевые значения сумм (эти метки будут удалены из сети в вершинах переходов t_{62}, t_{72} и t_8). Метка из вершины места p_{52} активизирует переход t_{52}, поскольку в вершине p_{53}, от которой идёт ингибиторная (запрещающая [4, 6]) дуга к t_{52}, нет метки. Переход t_{52} моделирует подписание депозитного договора, а перемещение метки в вершину p_{54} – начало работы депозита в данном банке; в p_{53} – блокирование новых пакетов для поступления на депозит на протяжение срока работы текущего депозита. Дальнейшее движение финансового пакета для вкладчика можно представить как мониторинг состояния банка и условий депозита, которые могут меняться за период депозита. Если условия остаются в пределах допустимых отклонений, определяемых состоянием финансового рынка и банковскими рисками, то метка, активизируя макропереход τ_{53}, возвращается в вершину места p_{54}. Если отклонения превышают допустимые пределы, метка из макроперехода τ_{53} направляется в вершину p_{55}, активизируется переход t_{54} погашая метки в вершинах мест p_{55} и p_{53}. При удалении метки из p_{53}, получаем новую возможность активизировать переход t_{52}, срабатывание которого моделирует подписание нового депозита. Из вершины перехода t_{54}, моделирующей окончание депозитного договора (из-за возросших финансовых рисков или по окончанию периода депозита), метка передаётся в вершину места p_8. Далее метка возвращается в вершину макроперехода τ_4, из которой может опять перейти в одну из вершин p_{52}, p_{62}, p_{72} для депозитного вклада, если такое вложение обосновано (рассчитывается $F(x,y)$ по величине финансового пакета x и оставшемуся времени вклада y_n), иначе метка будет передана через вершины p_{31} (окончание периода депозитов) и t_8, в которой производится суммирование полученных финансовых ресурсов, в выходную вершину p_k – итоговые ресурсы вкладчика.

Условие запрета вложения двух финансовых пакетов в один банк реализовано в модели конструкцией с ингибиторной дугой [4, 6, 10], которая на время поступления одного пакета на депозит банка, блокирует входящую в подмодель банка A_i вершину t_{i2} (метка в вершине p_{i3}) и не позволяет до окончания вклада (до активизации вершины перехода t_{i4}) её разблокировать.

При представлении модели управляющей сетью Петри (рис. 2) большинство макро-переходов представляем управляемыми переходами [6, 9], в которых реализуются алгоритмы расчёта условий выбора альтернатив или проводиться перекрещивание потоков действий (моделируемых движением меток). Также избавляемся от избыточных вершин, которые на рис. 1 выделены серым цветом.

При рассмотрении модели, представленной управляющей сетью Петри, нужно обратить внимание на различные функции, моделируемые управляющими векторами X_i.

Вектор X_1 моделирует выбор финансового пакета с максимальным номиналом и передаёт его в вершину места p_{21} для выбора его дальнейшего размещения, которое производиться в вершине t_4. Вектор X_2 позволяет скоординировать передачу метки только в одну из вершин

p_{52}, p_{62}, p_{72}, p_{31}, что избавляет от ненужных операций с передачей и погашением нулевых пакетов. Вектора X_5, X_6 и X_7 позволяют моделировать изменение внешних финансовых условий, которые влияют на расчёт показателя $F(x,y)$, и принимать решения о сохранении или прекращении срока депозита в связи с изменением внешних и внутренних показателей.

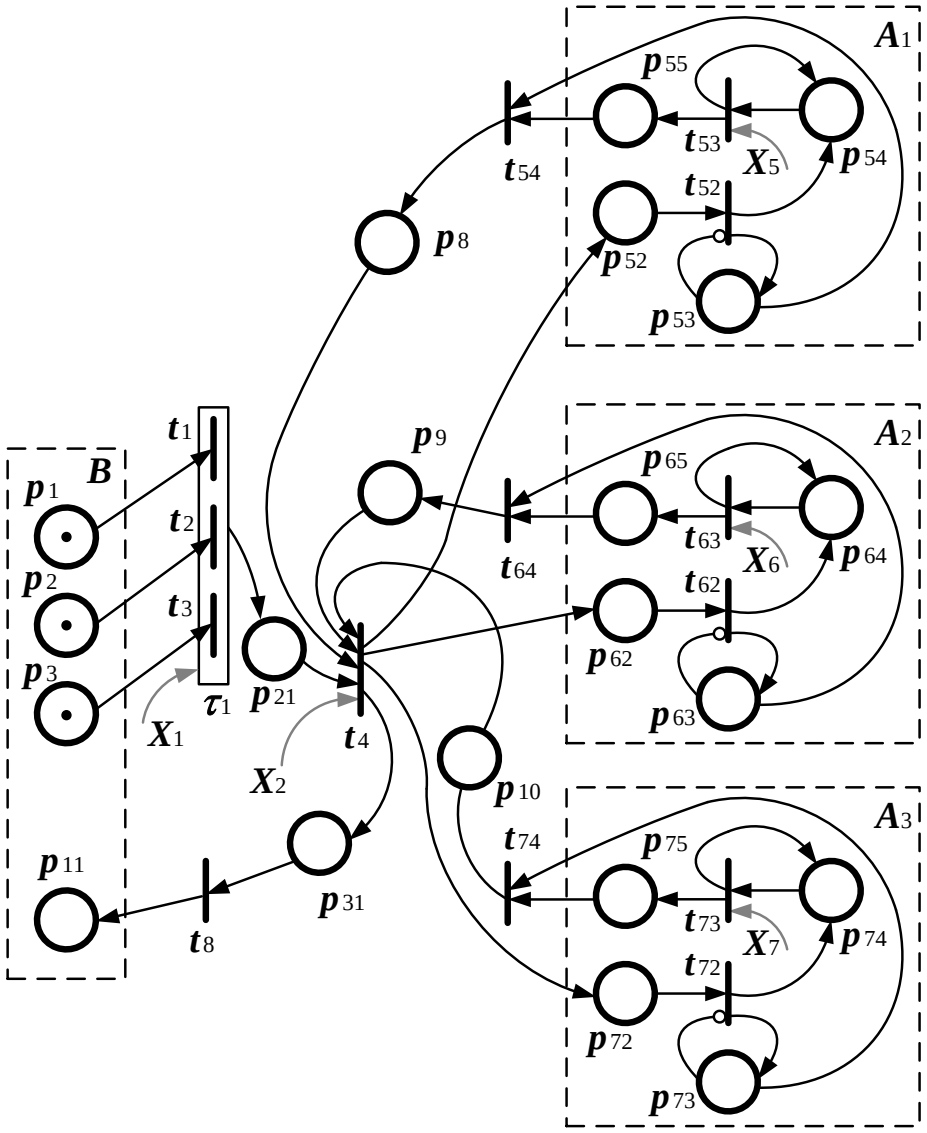


Рис. 2. Модель оптимизации размещения финансовых ресурсов на депозитах, представленная управляющей сетью Петри

5. Результаты исследования

Рассмотренный пример позволяет сравнить полученные модели задачи, построенные безопасной и управляющей сетями петри. Рассматривая рис. 1 и рис. 2 можем наглядно убедиться в лаконичности управляющей сети, которая, избавляя от мелких элементов модели, всё же позволяет достичь адекватного визуального отражения основных внутренних и внешних факторов, более обстоятельно провести анализ модели, устранить критические свойства, не наращивая при этом количество служебных (не смысловых) элементов сети петри.

6. Заключение

На примере задачи об эффективном размещении финансовых ресурсов продемонстрированы преимущества управляющей сети (SN) в сравнении с безопасной сетью Петри (SPN). При использовании управляющей сети удаётся наглядно отобразить многовариантный выбор альтернатив, передачу управляющей информации от внешних систем, более лаконично построить модель исследуемой системы. Важным свойством управляющих сетей Петри является установка приоритетов в вершинах переходов и мест [6], основанная на логических конструкциях, параметрами которых являются как внутренние, так и внешние переменные. Это позво-

ляет говорить о возможности построения адаптивных моделей сложных систем с многопоточным параллелизмом, оперативно реагирующих на изменение внутренних и внешних условий. Так, в рассмотренных моделях удалось реализовать алгоритм активного отслеживания финансовых показателей с целью минимизации рисков депозитных вкладов. Вместе с тем, представленное средство моделирования (SN) позволяет проводить в динамике (при перемещении меток) контроль критических свойств в модели, указывать локализацию небезопасных и конфликтных участков, что является одной из важнейших составляющих проектирования и разработки программных систем.

Список литературы

1. Бородакий В.Ю., Окороченко Г.Е. Анализ средств имитационного моделирования распределённых информационных систем. // Компьютерные системы и технологии: Научная сессия МИФИ. – 2007. – Том 12. – С. 129-130.
2. Любченко В. Создание и тестирование многопоточных программ. [Электронный документ]. Режим доступа: http://www.citforum.ru/programming/digest/multithreading/ Проверено: 19.11.2011.
3. Кузьмук В.В. Сети Петри и моделирование параллельных процессов. – Киев: ИПМЕ, 1985. – 64 с. (Препринт / АН УССР, Ин-т проблем моделирования в энергетике; 17).
4. Васильев В.В., Кузьмук В.В. Сети Петри, параллельные алгоритмы и модели мультипроцессорных систем – Киев: Наукова думка, 1990 – 216 с.
5. Кузьмук В.В. Парнюк А.М., Супруненко О.О. Класифікація мереж Петрі та приклади їх застосування для розв‘язання прикладних задач. // Восточно-европейский журнал передовых технологий. – 2011. – № 2/9 (50). – С. 40-43.
6. Кузьмук В.В., Супруненко О.О. Модифицированные сети Петри и устройства моделирования параллельных процессов: Монография. – Киев: Маклаут, 2010. – 252 с.
7. Хачатуров А.Р., Бабалова И.Ф. Моделирование процессов разработки программного обеспечения учебного назначения при помощи сетей Петри. // Компьютерные системы и технологии: Научная сессия МИФИ-2007. Т.12 Информатика и процессы управления. Компьютерные системы и технологии, 2007. – С. 159-160.
8. Кравцова Г.И., Румянцева О.И., Кузьменко Г.С. Деньги. Кредит. Банки. – Минск: Изд-во БГЭУ, 2007. – 444 с.
9. Кузьмук В.В., Кузьмук А.В., Супруненко О.А, Тараненко Е.А. Модифицированные сети Петри и современные методы моделирования параллельных процессов в сложных системах. // Управління розвитком складних систем. – 2011. – № 5. – С. 66-72.
10. А. с. 1416984 СССР, МКЛ G06 F 15/20. Устройство для моделирования графов Петри / Васильев В.В., Кузьмук В.В., Лисицин Е.Б., Шумов В.А.(СССР). – 4122948; заявл. 23.09.1986; опубл. 15.08.1988, БИ № 30. -199с.

УДК 004.932.72'1

ЗАЙЧЕНКО Ю. П.,
ДЬЯКОНОВА С. В.

ПРИМЕНЕНИЕ НЕЧЕТКОГО КЛАССИФИКАТОРА NEFCLASS К ЗАДАЧЕ РАСПОЗНАВАНИЯ ЗДАНИЙ НА СПУТНИКОВЫХ ИЗОБРАЖЕНИЯХ СВЕРХВЫСОКОГО РАЗРЕШЕНИЯ

В работе рассматривается применение нечеткого классификатора NEFClass к задаче распознавания зданий с разновидными формами крыши на спутниковых изображениях сверхвысокого разрешения. Проводится сравнительный анализ методов обучения нейронной сети: генетического, градиентного и метода сопряженных градиентов касательно поставленной задачи.

This article discusses the application of fuzzy classifier NEFClass to the problem of recognition buildings in high resolution satellite imagery. Comparative analysis methods for neural network training: genetic, gradient and conjugate gradient method on the problem of recognizing buildings with different forms of the roof.

Введение

Задача автоматического распознавания и извлечения зданий из спутниковых изображений сверхвысокого разрешения является важной частью в решении задачи автоматической интерпретации данных, получаемых из систем дистанционного зондирования земли. Автоматическая система распознавания зданий позволит сократить временные и материальные затраты на обновление базы данных ГИС (географической информационной системы), анализ использования земельных ресурсов и поддержание городских геодезических данных в актуальном состоянии.

Целью данной статьи является подробное рассмотрение и анализ метода распознавания разновидных крыш зданий на спутниковых изображениях сверхвысокого разрешения. В основе предлагаемого метода распознавания лежит нечеткий классификатор NEFClass [4] с различными алгоритмами обучения. В статье проводится сравнительный анализ результатов работы различных алгоритмов обучения с целью выбора наилучшего для решения задачи классификации разновидных форм крыши.

Предложенные ранее работы в области распознавания зданий разработаны с учетом того, что исследования проводятся на спутниковых изображениях, взятых из конкретного региона, где все здания имеют определенный оттенок и форму крыши. Рассмотрим некоторые из этих работ. В работе [1] было предложено решение, в котором для распознавания крыш зданий используется множество самоорганизующихся карт (SOM). Используя представление о степени насыщенности оттенков цветного спутникового изображения, множество самоорганизующихся карт обучается распознавать прямоугольные крыши различных цветов. Этот подход дает правильный результат только в 53% случаях, решение предполагает, что все крыши зданий должны иметь прямоугольную форму. В [2] был разработан подход, в котором в качестве признаков были использованы оттенки крыш и расположение теней зданий. Эксперимент был проведен для 177 зданий с корректным завершением в 86,6% случаев. Данный подход предполагает, что крыши зданий имеют прямоугольную форму и хотя бы часть из этих крыш красного оттенка. В работе [3] предлагается алгоритм, основанный на геометрических признаках и площади крыш зданий. После сегментации изображения применяется линейный классификатор для идентификации крыш зданий в каждом из полученных сегментов. В статье были преведены результаты для 240 протестированных зданий, большинство из которых имели красную крышу, алгоритм имел проблемы с идентификацией зданий, имеющих крыши других оттенков.

Описываемый в данной статье подход позволяет использовать систему распознавания для разновидных форм крыши зданий, находящихся в различных регионах. В качестве системы распознавания используется нейронная сеть, что дает гибкую и обучаемую систему, которая легко адаптируется для выполнения поставленной задачи.

Вычисление признаков объектов

В данной части статьи рассматриваются признаки, которые станут основой для распознавания крыш зданий на спутниковых изображениях.

В работе рассматривается 3 основных вида крыш: треугольная, плоская и круглая крыши. На Рис. 1 приведены примеры разновидных форм каждого из рассматриваемых видов крыши.

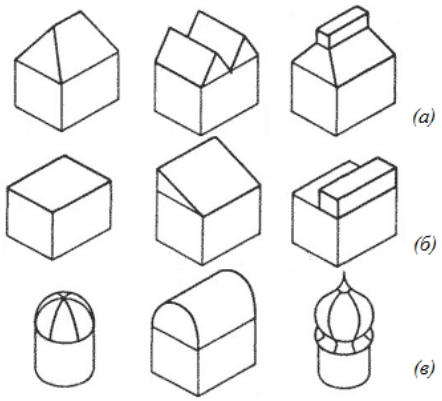


Рис. 1. Примеры разновидных форм крыши:
(а) здания с треугольной крышей
(б) здания с плоской крышей
(в) здания с круглой крышей

На основе знаний о геометрической форме крыш зданий были подобраны оптимальные признаки для распознавания. Рассмотрим подробнее используемые в работе признаки.

- Округлость крыши здания для каждого сегмента рассчитывается как отношение площади к периметру. Этот признак может принимать значения от 0 до 1. Округлость крыши вычисляется по следующей формуле:

$$Округлость = \frac{4\pi * Площадь}{Периметр^2}$$

  где Площадь сегмента рассчитывается как количество принадлежащих ему пикселей, а Периметр сегмента рассчитывается как количество пикселей лежащих на границе области.

- Среднее значение углов [3]. Углы являются важным признаком, позволяющим дифференцировать крышу здания от других объектов. Для того чтобы определить значение углов в выделенном сегменте необходимо построить шестиугольник по следующей схеме: 1. Проводится главная ось между двумя точками контура, которые находятся на максимальном расстоянии друг от друга. 2. Определяются две вспомогательные оси с каждой стороны главной оси. Для этого строим вертикальные линии, лежащие на главной оси и имеющие максимальное расстояние от контура до главной оси. 3. Проводим поперечную ось, это должна быть вертикальная линия на главной оси, которая соединяет две точки контура находящиеся на максимальном расстоянии друг от друга. 4. Полученные шесть точек соединяем линиями в шестиугольник. В качестве признаков возьмем среднее значение двух углов, находящиеся по обе стороны главной оси.

- Длина крыши здания определяется как отношение длины главной оси к длине поперечной оси, полученного выше шестиугольника [3]:

$$Длина = \frac{m_{20} + m_{02} + \sqrt{(m_{20} - m_{02})^2 + 4m_{11}^2}}{m_{20} + m_{02} - \sqrt{(m_{20} - m_{02})^2 + 4m_{11}^2}}$$

- Полнота вычисляется как отношение общего количества пикселей, которые лежат внутри шестиугольника и не принадлежат к исследуемой области к количеству пикселей, которые лежат за пределами шестиугольника и принадлежат исследуемому сегменту.

- Градиент яркости поверхности крыши. Для определения градиента яркости будет использован оператор Собеля, который вычисляет приближенное значение градиента яркости изображения. В результате применения оператора Собеля, в каждой точке изображения будет получен вектор градиента яркости в данной точке. Таким образом, будет найдено направление наибольшего увеличения яркости и величина её изменения в данном направлении. В разработанном алгоритме сегмент делится на 20 равных частей, в центре каждой из полученных частей вычисляется значение оператора Собеля. В зависимости от полученных направлений увеличения яркости можно говорить об одном из исследуемых видов крыш.

Экспериментальные исследования

Для проверки эффективности работы рассматриваемой нейронной сети были проведены

тесты на изображениях с различными видами крыш. Для проведения экспериментов было использовано 90 изображений, из них 60 изображений использовались для обучения нейронной сети. Спутниковые изображения, на которых преобладают здания с круглыми крышами, отбирались из регионов, имеющих пустынный ландшафт, с треугольными крышами - из пригородных территорий, с плоскими крышами – из территорий крупных городов. Для тестирования было использовано 30 спутниковых изображений.

В разработанной системе NEFClass входам нейронной сети соответствуют описанные выше 5 признаков: округлость крыши, среднее значение двух углов, длина крыши, полнота, градиент яркости поверхности крыши. Выходам сети соответствуют 3 исследуемых вида крыш: круглая, треугольная, плоская.

Для обучения была использована гауссовская функция принадлежности [4], поскольку она дает более реальные данные и обеспечивает гибкость настройки нейронной сети.

$$\mu(x) = \exp[-(\frac{x-c}{\sigma})^2] \qquad (1)$$

Эксперимент был поставлен следующим образом. Каждое изображение было предварительно обработано и сегментировано. Каждый полученный сегмент содержит изображение крыши одного из видов. Для генерации базы правил каждому сегменту на изображении ставится в соответствие подходящая форма крыши. Таким образом, создается база правил. Проводим обучение нейронной сети одним из выбранных методов: генетическим, градиентным или методом сопряженных градиентов [4]. Далее на обученную сеть подаем тестовое изображение, на выходе получаем изображение, на котором отмечены распознанные крыши различными цветами, в зависимости от формы крыши. Так круглые крыши будут отмечены желтым цветом, треугольные – синим, а плоские – красным.

Если сравнить реализованные алгоритмы обучения между собой, то можно увидеть, что не все три метода справляются хорошо и можно отметить следующие тенденции. Метод сопряженных градиентов в начале обучения сходится значительно быстрее, чем обычный градиентный метод и в итоге дает лучший результат. Генетический же метод сходится гораздо медленнее градиентного метода и метода сопряженных градиентов. Кроме того, генетический метод отличается этапным улучшением суммарной погрешности, то есть при общей тенденции сходимости и медленном уменьшении ошибки, есть моменты довольно резкого уменьшения ошибки.

Необходимо отметить, что градиентный метод и метод сопряженных градиентов сошлись к ошибке, которая была ниже установленного порога, поэтому оказались лучше генетического алгоритма. Однако в случае большего количества итераций и, возможно, других данных генетический алгоритм мог оказаться лучше, о чем свидетельствуют результаты, полученные при выполнении работы и теоретические данные [4].

Сказанное выше проиллюстрировано на Рис. 2, который показывает зависимость суммарной погрешности от пройденных итераций по каждому из выбранных методов.

Рассмотрим график на Рис. 3, который показывает процесс обучения нейронной сети для каждого вида крыши, на примере обучения методом сопряженных градиентов, поскольку этот метод дал наилучший результат.

Можно увидеть, что наиболее легким примером для обучения нейронной сети были здания с плоскими крышами. Среди зданий этого типа наибольшее количество распознанных. В тоже время треугольная форма крыша была наиболее сложным примером для обучения и дольше других удерживала значение погрешности выше желаемого порога.

Также треугольная форма крыши чаще всего встречалась в нераспознанных образцах, что можно объяснить более сложной геометрической формой.

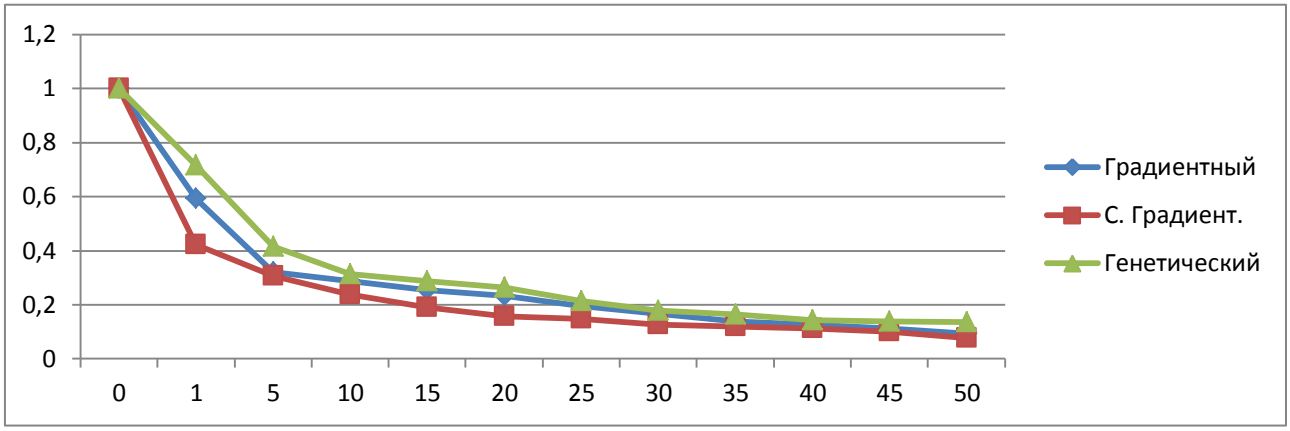


Рис. 2. Графическое представление зависимости суммарной погрешности от пройденных итераций

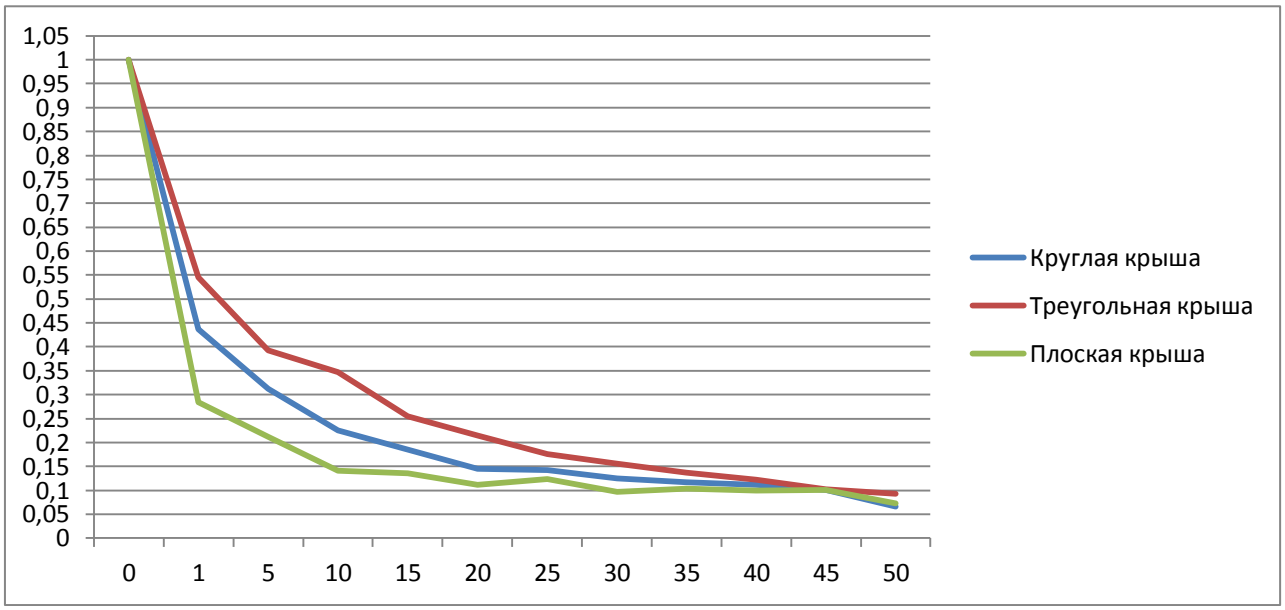


Рис. 3. Графическое представление зависимости погрешности от пройденных итераций для каждого вида крыши

Для эксперимента было отобрано 90 изображений – 60 для обучения и 30 для тестирования. Результаты обучения и классификации приведены в Табл. 1. По полученным результатам эксперимента можно сделать вывод, что наилучший результат был получен при обучении с помощью метода сопряженных градиентов.

Необходимо отметить, что система не всегда правильно определяет тип крыши, что говорит о необходимости коррекции используемых признаков. Кроме того, наличие вокруг зданий на близком расстоянии деревьев в большом количестве увеличивает процент ошибочно классифицированных образцов. Это верно как для критерия количества неправильно классифицированных символов, так и по скорости обучения.

Табл. 1. Результаты обучения и тестирования

Алгоритмы обучения весовых коэффициентов	Обучение		Тестирование	
	Величина ошибки	% ошибочно классифицированных	Величина ошибки	% ошибочно классифицированных
Генетический	0,13	0	0,19	23,3
Градиентный	0,09	0	0,12	13,3
Сопряженного градиента	0,07	0	0,09	6,6

При тестировании было ошибочно классифицировано 14,4 % из проверочного набора данных.

Выводы и перспективы дальнейших исследований

Практическая реализация модели NEFClass показала, что эта система дает достаточно высокие результаты применительно к задаче распознавания крыш зданий на спутниковых изображениях. Программное обеспечение, разработанное при выполнении работы, показывает основные возможности нечетких нейронных сетей типа NEFClass и может служить базой для развития темы распознавания зданий в дальнейшем. Кроме того, разработанное программное обеспечение может быть использовано для проведения экспериментов по усовершенствованию методов построения и обучения нечетких нейронных сетей. Реализовав на практике градиентный, генетический алгоритмы, и алгоритм сопряженных градиентов, и проанализировав полученные данные, можно заключить, что лучшим для решения поставленной перед системой задачи оказался метод сопряженных градиентов, который быстро сходится и дает наилучшие результаты. Необходимо отметить, что требуются дальнейшие исследования, направленные на усовершенствование нейросетевой архитектуры и методов ее обучения, об этом говорит достаточно высокий процент ошибочно классифицированных зданий. Это улучшение может быть получено путём коррекции признаков, которые поступают на вход сети и увеличения обучающей выборки. Также возможно улучшение результата путем дообучения базы правил с помощью алгоритма «наилучший для каждого класса» [4]. Такой подход позволит снизить процент неверно классифицированных крыш зданий. В заключение следует ещё раз отметить, что по результатам проведенных исследований можно говорить о достаточно высокой результативности применения нечеткого классификатора к задаче распознавания зданий на спутниковых изображениях сверхвысокого разрешения.

Список литературы

1. Persson M., Sandvall M., Duckett T. Automatic Building Detection from Aerial Images for Mobile Robot Mapping. – Finland: International Symposium on In Computational Intelligence in Robotics and Automation 2005, pp. 273-278.
2. Sirmacek B. Unsalan C. Building Detection from Aerial Images Using Invariant Color Features and Shadow Information. – Turkey: International Symposium on Computer and Information Sciences 2008, pp.1-5.
3. S. Müller and D. W. Zaum, Robust building detection from aerial images, IAPRS, Vol. XXXVI, Part 3/W24, Vienna, Austria, 2005. – pp. 143-148
4. Зайченко Ю.П. Нечёткие модели и методы в интеллектуальных системах. Учебное пособие для студентов высших учебных заведений. – К.: «Издательский Дом «Слово»», 2008. – с. 344

УДК 004.056

КУССУЛЬ О.М.,
НОВИКОВ А.Н.

МНОГОКРИТЕРИАЛЬНАЯ ОПТИМИЗАЦИЯ ПЛАНИРОВАНИЯ ВЫПОЛНЕНИЯ ЗАДАЧ В СТРУКТУРНО-СЛОЖНЫХ СИСТЕМАХ НА ОСНОВЕ МОДЕЛЕЙ РЕПУТАЦИИ

В данной работе решается задача многокритериальной оптимизации планирования выполнения задач в структурно-сложных системах на основе моделей репутации. Предложен новый подход для интеграции модели репутации в подсистему планирования с использованием нелинейной схемы компромиссов. Полученные результаты экспериментов демонстрируют эффективность разработанного подхода.

In this paper we consider a problem of multi-criteria optimization of task scheduling in structural-complex systems using reputation models. We propose a new approach for integrating reputation into scheduler by applying a non-linear tradeoff scheme. Results of experiments are presented which show the effectiveness of the proposed approach.

1. Введение

В последние годы бурное развитие вычислительной техники и информационных технологий привело к широкому распространению сервис-ориентированных распределенных систем. Многие из них представляют собой инфраструктуру для совместного использования данных, алгоритмов, ресурсов хранения и вычислительных мощностей [1]. Примерами таких систем являются система систем GEOSS[1], Grid-системы [2], системы на основе технологии Sensor Web [3], системы электронного взаимодействия (e-collaboration) [4]. Основу работы таких систем зачастую составляют виртуальные организации (ВО) – временные или постоянные объединения географически распределенных отдельных пользователей, групп, подразделений организаций или целых организаций, которые совместно используют ресурсы и информацию для достижения общих целей [5]. Виртуальные организации динамически формируются, существуют некоторое время и распадаются.

В последнее время особое внимание уделяется управлению доверием в сервисориентированных структурно-сложных системах [6, 7, 8, 9]. В работе [10] показано, что доверие – это высокоэффективная технология, и ее внедрение позволит обезопасить электронные транзакции. При этом доверие описывается как важное и сложное понятие, связанное с честностью, правдивостью и надежностью доверенного лица или сервиса. Тем не менее, единого формального определения понятия доверия не существует [11].

Системы управления доверием можно условно разделить на две группы [8]: основанные на использовании политик безопасности и основанные на вычислении репутации. В системах управления доверием основанных на использовании политик безопасности отношения доверия между субъектами основываются на конкретных, заранее оговоренных, политиках безопасности. Такие системы являются излишне жесткими и не адаптивными. В системах управления доверием, основанных на вычислении репутации, вводятся определенные формализмы для оценки доверия, которое вычисляется как функция от репутации. Репутация – это предположение о поведении агента на основе имеющейся информации или наблюдений о его поведении в прошлом [12].

Среди известных моделей репутации следует выделить известную модель, описанную в [13]. Недостатком этой модели является непродуманность инициализации модели для нового ресурса или пользователя и неустойчивость к атакам. В работах [14, 15] эта модель была модифицирована путем введения методики активного эксперимента для определения начальной репутации ресурса (поставщика услуг), учета взаимосвязей между пользователями и провайдерами ресурсов во избежание группового сговора [16], оценивания репутации на интервале времени, а не в конкретный момент [9, 16], а также *учета сложности* разных типов сервисов [17]. В частности, успешное выполнение задач со сложным потоком выполнения (workflow) или распараллеленных программ (например, в задачах экологического мониторинга, таких как

[1] GEOSS, http://www.earthobservations.org

численный прогноз погоды [18]) обеспечивает поставщику ресурсов более высокую репутацию, чем выполнение более простой задачи. Кроме того, в работах [14, 15] было усовершенствовано определение функции полезности путем включения статистической модели поведения пользователя, разработанной ранее для компьютерных сетей и распределенных систем [19, 20, 21, 22].

Данная статья посвящена применению разработанной модели для планирования выполнения задач в структурно-сложных системах. В частности, будет показано, как использование модели репутации брокером ресурсов позволяет повысить эффективность управления ресурсами сервис-ориентированных систем. Задача планирования распределения задач между ресурсами формулируется как многокритериальная задача оптимизации. Модель репутации предлагается интегрировать в подсистему планирования с использованием нелинейной схемы компромиссов [23], что обеспечивает дополнительные преимущества по сравнению с традиционной мультипликативной схемой для формирования обобщенного критерия.

2. Анализ существующих подходов

На сегодняшний день предложено несколько концепций построения и использования репутации в сервис-ориентированных системах и интеграции репутации в планировщик задач. В данной статье описывается подход, основанный на модели репутаций, которая вычисляется на основе функции полезности и может применяться для оценки репутации как пользователей, так и поставщиков ресурсов [13]. Эта модель основана на вычислении функции полезности, которая выражает удовлетворенность одного объекта его взаимодействием с другими объектами, учитывая ключевые качества, свойственные оцениваемому объекту. При вычислении репутации пользователей оценивается, как и какие ресурсы используются пользователем и насколько его поведения соответствует заданной политике безопасности ВО. В свою очередь, репутация поставщика ресурсов оценивается на основе параметров качества обслуживания (QoS), которое он предоставляет. Для оценки эффективности предложенной модели авторами проведено имитационное моделирование для статического планировщика задач: для заданного набора задач, отправленных пользователями в систему, каждый ресурс обрабатывал количество запросов, прямо пропорциональное его репутацию. В работе показано, что при использовании планировщика с учетом репутации общее время выполнения задач можно уменьшить на 25%.

В работе [6] особое внимание уделяется экономическим аспектам Grid-систем с учетом информационной асимметрии. Информационная асимметрия представляет собой ситуацию, когда поставщик ресурсов и пользователь получают разную информацию о параметрах качества обслуживания. Эти аспекты учитываются при разработке модели репутации с последующим коммерческим использованием Grid-систем. Вопросы информационной ассиметрии в Grid детально рассмотрены в работе [24]. Предложен механизм на основе репутаций, позволяющий эффективно учитывать скрытую (недоступную) информацию.

В работе [7] предложена модель доверия на основе методов нечеткой логики. Разработанная модель агрегирует параметры безопасности распределенных ресурсов в Grid-системе путем их фаззификации. В работе использованы следующие критерии для оценки уровня доверия Grid-узла: репутация узла (процент успешно выполненных задач, кумулятивный уровень использования узла, среднее время простоя задач, среднее время выполнения задач) и уровень защищенности узла (набор программного и аппаратного обеспечения для организации безопасности узла). Для безопасного планирования большого количества автономных и унитарных задач по Grid-узлам разработана система Secure Grid Outsourcing (SeGO). Для динамического планировщика задач используется эвристический алгоритм Min-Min. Результаты экспериментов показали существенный прирост производительности при агрегации доверия в планировщик задач.

В работе [25] для управления ресурсами Grid-сситемы с учетом репутации предложен каркас HOURS. Система ориентирована на автоматический перезапуск задач, самозащищенность, совместное использование гетерогенных ресурсов, резервирование и установление гарантированного уровня качества обслуживания в Grid-вычислениях. Для уменьшения количества повторно отправляемых в систему задач и уровня невыполнения задач разработан планировщик ресурсов, учитывающий репутацию вычислительного узла. Эксперименты проводились на основе реальных данных, полученных в системе TeraGrid. Результаты моделирования

показали, что использование планировщика задач с репутацией может снизить уровень невыполнения задач с 3.82 до 0.70 относительно стандартного планировщика.

В статье [16] представлена модель доверия для Grid-систем, которая используется для планирования задач с учетом требуемых параметров безопасности. Для интеграции репутации в эвристические планировщики задач используется мультипликативная схема. Формально показано, что время обработки всех задач при использовании планировщика с учетом репутации всегда будет меньше времени выполнения при использовании планировщика без репутации (при условии, что базовая эвристика распределения задач в обоих случаях - одинакова).

В работе [26] разработан генетический алгоритм для планирования задач, который учитывает разнородность механизмов обеспечения отказоустойчивости в вычислительных Grid-системах. Риск запуска задачи на конкретном узле оценивается на основе выдвигаемых требований безопасности и уровнем доверия. Предложенный алгоритм обеспечивает меньшее время выполнения задач и меньшее количество невыполненных задач по сравнению с алгоритмами Min-Min и Sufferage.

Система PathTrust [27] – это система репутаций, предложенная для выбора членов ВО на этапе ее формирования. Для того, чтобы войти в ВО, организация должна зарегистрироваться с инфраструктурой сети предприятия (enterprise network) путем предоставления некоторых сертификатов. Помимо управления пользователями сеть предприятия предоставляет централизованный сервис репутаций. Когда ВО распадается, каждый член оставляет определенные значения обратной связи для сервера репутаций для других членов, с кем он вступал во взаимодействие. Эти значения обратной связи могут быть положительными или отрицательными. Данной схеме не хватает динами, поскольку данные обратной связи агрегируются только в момент распада виртуальной организации.

Модель доверия GridEigenTrust [28] является расширением модели EigenTrust [29], разработанной ранее для систем P2P (peer-to-peer). В модели GridEigenTrust используется метод получения глобальной репутации с учетом иерархий и вычисления доверия на основе собственных чисел (eigenvalue-based trust calculation algorithm). Преимуществами данного подхода являются быстрая сходимость и экономность по сравнению с вычислением глобального значения доверия для отдельных субъектов для каждой ситуации. В случае, когда организация предоставит неправдивую информацию о своих субъектах, она будет оштрафована путем понижения глобального уровня доверия всей организации. В дальнейшем значение репутации интегрируется в систему управления обеспечением качества обслуживания (QoS management system), предоставляя возможность переопределить механизм выбора ресурсов и гарантированного уровня качества обслуживания (SLA).

3. Модель репутации на основе вычисления функции полезности в сервис-ориентированных системах

В качестве модели репутации будем исследовать модификацию известной модели [13], предложенную в [14, 15].

3.1. Модель репутации для поставщиков ресурсов

Модель репутации поставщиков ресурсов, предложенная и детально описанная в [14, 15], основана на функции полезности, которая определяет уровень удовлетворенности пользователя предоставленным сервисом. Она содержит две составляющие: модель репутации поставщика ресурсов и модель репутации пользователей. Обе составляющие основываются на вычислении функции полезности, введенной в [14, 15], и определяющей уровень сервиса. Приведем формальное описание этих моделей.

Функция полезности имеет вид:

$$utility : Event \rightarrow \mathbf{R},$$

где пространство событий *Event* представляет собой декартово произведение времени, ресурсов, пользователей ВО и качества обслуживания:

$$Event = T \times \bigcup_l u_l \times \bigcup_k r_k \times \bigcup_m vo_m \times \{\text{QoS name}\} \times \mathbf{R} \quad (1)$$

где T – временной интервал.

Функция полезности вычисляется следующим образом:

$$utility(\{t, u, r, vo, QoS, v\}) = \begin{cases} h(u,r)s(r), & \text{если } SLA \text{ предоставлен} \\ penalty\ (v, SLA)h(u,r)s(r), & \text{иначе} \end{cases}, \quad (2)$$

где t – время, u – пользователь, r – ресурс, vo – ВО, QoS – качество обслуживания, v – реальное значение параметра качества обслуживания, h – функция принадлежности пользователя и ресурса к одной организации, SLA – заранее со-

гласованное значение уровня качества услуг между пользователем и поставщиком ресурсов, *penalty*(*v, SLA*) – функция штрафа, наложенная на поставщик ресурсов, если заранее согласованный уровень качества услуг не выдерживается.

Вид функции штрафа зависит от требований к предоставляемому качеству услуг QoS. Например, для метрик времени, которые в основном необходимо минимизировать, функция штрафа может принимать следующий вид:

$$penalty(v, SLA) = \begin{cases} 1, & \text{если } v \le SLA \\ \dfrac{SLA}{v}, & \text{если } v > SLA \end{cases}. \quad (3)$$

Вычислим значение функции полезности на основе последовательности событий $Trace|_{(vo, r, t)}$:

$$O_{(vo, r, t)} = \{ z(t, t_c) \cdot utility(\{t, u, r, vo, QoS, v\}) \mid \{t, u, r, vo, QoS, v\} \in Trace|_{(vo, r, t)}\}. \quad (4)$$

Репутация – это математическое ожидание функции полезности:

$$rep(vo, r, t) = \mathrm{E}[\, utility(O_{(vo, r, t)}) \,] = \int utility\,(O_{(vo,r,t)})\, p_{utility}\,(O_{(vo,r,t)})\, dO_{(vo,r,t)} \,. \quad (5)$$

Для аппроксимации математического ожидания будем использовать выборочное среднее (4):

$$rep(vo, r, t) = \frac{1}{\left|O_{(vo,r,t)}\right|} \sum_{x \in O_{(vo,r,t)}} x \quad , \quad (6)$$

где $|\cdot|$ – мощность множества.

Репутация организации *o* в ВО – это агрегация репутации всех ресурсов, которые эта организация предоставляет в ВО (6):

$$rep(vo, t) = \frac{1}{\left|f_{vo}^{-1}(o)\right|} \sum_{r \in f_{vo}^{-1}(o)} rep\,(vo, r, t)\,. \quad (7)$$

Репутацию ресурса во всех ВО можно оценить следующим образом (6):

$$rep(r, t) = \frac{1}{\left|VO\,\right|_r} \sum_{vo \in VO\,|_r} rep\,(vo, r, t)\,. \quad (8)$$

3.2. Модель репутации для пользователей

В работе [13] модель репутации для пользователя строится с использованием функции штрафа. Если пользователь совершает действия, которые не соответствуют политике безопасности ВО или ресурса, то на пользователя накладывается штраф. Этот штраф используется для оценки функции полезности для пользователя, а соответственно и его репутации. Но аудит действий пользователя, а также сравнение его действий с политикой безопасности ВО или ресурса является достаточно сложной и трудоемкой задачей, особенно с точки зрения реализации в реальной системе. Должны быть четкие и согласованные критерии, а также соответствующие программные компоненты, позволяющие реализовать такой аудит и проверку с заданными политиками безопасности.

Для того чтоб оценить репутацию пользователя в данной работе были использованы методы и модели, которые традиционно применялись в системах выявления вторжений (IDSs). На данный момент уже предпринимались попытки использования систем выявления вторжений для Grid-систем [30, 31].

В данной работе была использована статистическая модель поведения пользователя (SMUB), изначально разработанная для компьютерных сетей, а в дальнейшем расширена для распределенных сетей [19, 20, 21, 22]. Эта модель основана на анализе статистической информации, которая собирается после выполнения пользователем действий в системе. Данная модель верифицировалась на реальных данных, собранных в инфраструктуре GILDA проекта EGEE. При проведеннии экспериментов были получены следующие результаты: с помощью модели можна было определить аномальное поведение пользователей в среденем в 86%, ошибка первого рода составила 7.48%, а второго – 20.9%.

Выход модели может интерпретироваться, как «репутация пользователя» в том смысле, что текущие действия пользователя соответствуют его поведению в прошлом. Такая модель пользователя будет специфической для каждой виртуальной организации в зависимости от ее целей и типов задач, которые выполняются в ней. Например, ВО может быть ориентирована на приложения, которые требуют выполнения большого количества задач со сравнительно небольшим объемом данных. В таких ВО задачи, требующие всей оперативной и виртуальной памяти ресурса, будут рассматриваться как аномальные шаблоны поведения. С другой стороны, другие ВО могут быть ориентированы на приложения, где одна задача состоит из нескольких элементарных подзадач,

каждая из которых требует обработки большого количества данных. Такие задачи характерны для области наук про Землю и обработки спутниковых данных [1, 32].

К преимуществам такой модели стоит отнести возможность выявлять отклонения от моделей поведения пользователя, отличать разных пользователей и интегрировать известные шаблоны поведения в процесс настройки параметров модели. Недостатками такой модели являются достаточно высокий уровень ложных срабатываний и необходимость переобучать модель через некоторое время. Несмотря на то, что у предложенной модели есть как достоинства, так и недостатки необходимо отметить, что ее целесообразно использовать вместе с другими механизмами безопасности, что позволит более эффективно оценить репутацию пользователя.

Приведем формальное описание модели репутации для пользователя, основанной на статистической модели поведения пользователя.

Для пользователя определим набор *Event* следующим образом:

$$Event = \{t, u, r, vo, \mathbf{x}\}. \quad (9)$$

где **x** – вектор параметров статистической модели поведения пользователя, описанный в [14, 15].

Последовательность событий (*Events)* имеет следующий вид:

$$Trace = \bigcup_p Event_p = \bigcup_p \{t, u, r, vo, \mathbf{x}\}_p . \quad (10)$$

Функцию полезности *utility*() определим следующим образом (10):

$$utility : Event \rightarrow \mathrm{R}, \quad (11)$$

$$utility(\{t, u, r, vo, \mathbf{x}\}) = SMUB_{(u, vo)}(\mathbf{x}), \quad (12)$$

где $SMUB_{(u, vo)}(\mathbf{x})$ – выход статистической модели поведения пользователя, который может принимать значения из отрезка [0; 1].

Важно понимать, что в общем случае в качестве функции полезности для пользователя *utility*() могут быть использованы другие модели поведения пользователя, например [33, 34], или ансамбль разных моделей для того, чтобы учесть различные аспекты поведения пользователя.

Статистическая модель поведения пользователя основана на использовании нейронных сетей [35] и является специфической для пользователя и ВО [15]. Определим набор параметров, которые используются для оценки репутации пользователя *u* в ВО *vo* до текущего времени *t* следующим образом:

$$Trace|_{(vo, u, t)} = \left\{ \begin{array}{l} \{t', u', r', vo', \mathbf{x}'\} \in Trace \ : \\ u = u', vo = vo\_i\ d', t' \le t \end{array} \right\} \quad (13)$$

Вычислим значение функции полезности на основе последовательности событий $Trace|_{(vo,u,t)}$ (12-13):

$$O_{(vo, u, t)} = \{ z(t, t_c) \cdot utility(\{t, u, r, vo, \mathbf{x}\}) \mid \{t, u, r, vo, \mathbf{x}\} \in Trace|_{(vo, u, t)}\}. \quad (14)$$

Значение репутации вычисляется следующим образом:

$$rep(vo, u, t) = \mathrm{E}[\, utility(O_{(vo, u, t)}) \,] = \int utility\ (O_{(vo,u,t)})\, p_{utility\ (O_{(vo,u,t)})}\, dO_{(vo,u,t)} \ . \quad (15)$$

Для аппроксимации математического ожидания будем использовать выборочное среднее (14)

$$rep(vo, u, t) = \frac{1}{\left|O_{(vo,u,t)}\right|} \sum_{x \in O_{(vo,u,t)}} x \quad . \quad (16)$$

Репутация организации в ВО (с точки зрения пользователя) – это агрегация репутаций всех пользователей, которые принимаю участие в ВО (16):

$$rep(vo, t) = \frac{1}{\left|g_{vo}^{-1}(o)\right|} \sum_{r \in g_{vo}^{-1}(o)} rep\ (vo, u, t) . \quad (17)$$

Репутация пользователя во всех ВО можно оценить следующим образом (16):

$$rep(r, t) = \frac{1}{\left|VO\,|_u\right|} \sum_{vo \in VO\,|_u} rep\ (vo, u, t) . \quad (18)$$

4. Интеграция репутации в планировщик задач для grid-систем

В данной работе рассматривается задача динамического планирования, при которой задачи в Grid-системе распределяются по ресурсам сразу после поступления (в отличии от статического планирования, которое использовалось в работе [13]). Для интеграции репутации в алгоритм планирования предлагается новый подход с использованием нелинейной схемы компромиссов [23].

Обозначим критерий, ассоциированный с планировщиком посредством *y*, т.е. этот критерий минимизируется при распределении задач на ресурсы Grid-системы. В качестве такого критерия могут выступать: минимальное время выполнения задачи, приемлемое время выполнения [36], текущее количество задач в очереди ресурса или уровень отказов. Пусть $rep(r_i)$ – это значение репутации ресурса r_i. При интеграции репутации в планировщик ресурсов возникает следующая многокритериальная задача опти-

мизации: необходимо выбрать ресурс, который минимизирует значение $y(r_i)$, но при этом имеет максимальную репутацию $rep(r_i)$. Таким образом, необходимо синтезировать в единую скалярную функцию частные критерии, которая бы в различных ситуациях выражала бы разные принципы оптимальности. Для этого предлагается использовать нелинейную схему компромиссов [23].

В данной схеме нормализованные частные критерии y_k синтезируются в единую свертку, используя следующее выражение:

$$Y(x) = \sum_{k=1}^{s} \alpha_k \left[1 - y_k(x)\right]^{-1}; \alpha_k \geq 0, \sum_{k=1}^{s} \alpha_k = 1, \quad (19)$$

где α_k – параметры имеющие двоякий физический смысл [23]. С одной стороны – это весовые коэффициенты, выражающие значения отдельных критериев, с другой – это коэффициенты регрессии содержательной регрессионной модели функции полезности, построенной на основе концепции нелинейной схемы компромиссов. Преимуществом использования в качестве интегрирующей функции (19) является ее способность в промежуточных случаях приводить к парето-оптимальным решениям, дающим различные меры частичного удовлетворения критериев.

Репутация может быть интегрирована в планировщик ресурсов следующим образом:

$$Y(r_i) = \frac{\alpha_1}{1 - y_n(r_i)} + \frac{\alpha_2}{rep(r_i)}, \quad (20)$$

где $y_n(r_i)$ – нормализированное значение критерия, ассоциированного с планировщиком.

Таким образом, задача отправляется на ресурс, который минимизирует функцию (20):

$$r^* = \arg\min_{r_i} Y(r_i). \quad (21)$$

В проведенных экспериментах для распределения задачи на ресурс использовалась эвристика *ECT* (определение минимального времени выполнения задачи). Пусть $ECT(r_i)$ – оценка время выполнения задачи на ресурсе r_i, а $ECT_n(r_i)$ – соответствующее нормализированное значение:

$$ECT_n = \frac{ECT(r_i)}{ECT_{\max}}, \quad (22)$$

где ECT_{max} – максимальное значения ECT.

Планировщик *ECT-reputation*, учитывающий репутацию, отправляет задачу на тот ресурс, который минимизирует следующее выражение:

$$r^* = \arg\min_{r_i} \left\lceil \frac{\alpha_1}{1 - ECT_n(r_i)} + \frac{\alpha_2}{rep(r_i)} \right\rceil. \quad (23)$$

Для оценки эффективности предложенная схема сравнивалась мультипликативной схемой, при которой задача отправляется на ресурс, минимизирующий следующее выражение:

$$r^* = \arg\min_{r_i} \left[ECT(r_i)(1 - rep(r_i))\right]. \quad (24)$$

5. Результаты экспериментов

В данном разделе представлены результаты экспериментов для оценки эффективности предложенной модели. Эффективность оценивалась с точки зрения улучшения управления ресурсами в Grid-системе. Для проведения имитационного моделирования различных сценариев было разработано специализированное программное обеспечение.

5.1. Описание данных

Для генерации загруженности системы при проведении экспериментов использовались реальные данные, полученные проектом Grid Observatory[2]. В рамках этого проекта предоставляются данные «жизненного цикла» задач в Grid-инфраструктуре EGEE. В частности, были использованы данные, собранные системой Real Time Monitor (RTM), которая интегрирует различную информацию о задачах, выполняемых в Grid. Для каждой задаче имеется 37 атрибутов, которые делятся на следующие категории: информационные (Information), метки времени (Timestamps) и метрики (Metrics) [37].

5.2 Параметры экспериментов

Все эксперименты проводились для Grid-инфраструктуры, состоящей из 20 ресурсов. Производительность ресурсов (в условных единицах) равномерно выбиралась из промежутка [1, 200]. Сложность задач генерировалась на основе данных проекта Grid Observatory и находилась в отрезке [1, 56000]. Распределение сложности задачи представлено на рис. 1. Время выполнения задачи на ресурсе оценивалось как отношение сложности задачи к производительности ресурса *jobComplexity*/*resource Productivity*. Время между поступлениями задач и нагрузка на вычислительную систему также генерировались на основе статистических данных Grid Observatory. На рис. 2 и 3 представле-

[2] Grid Observatory: www.grid-observatory.org

но изменение общего количества отправленных задач от времени и скорость поступления задач соответственно.

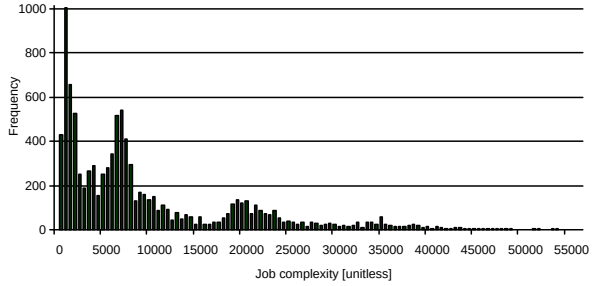


Рис. 1. Распределение сложности задач (общее число задач - 10000)

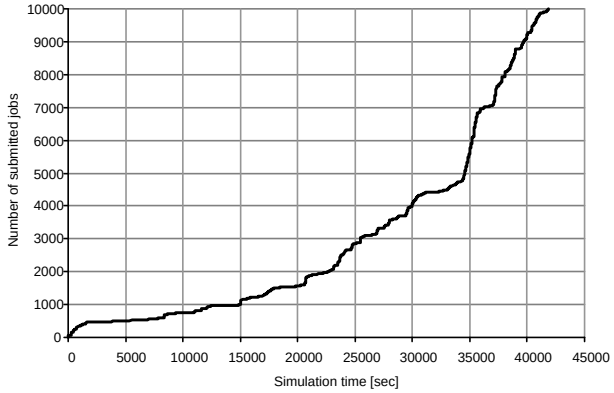


Рис. 2. Динамика изменения общего количества отправленных на выполнение задач от времени

В рамках экспериментов использовались следующие параметры качества обслуживания QoS: время ожидания задачи, время обработки задачи и общее время выполнения задачи. Значение согласованного уровня качества обслуживания SLA формировалось таким образом: время ожидания задачи выбиралось случайным образом из отрезка [1, 30000] сек, время обработки задачи выбиралось как отношение сложности задачи к минимальной производительности ресурса *jobComplexity*/*minResourceProductivity*. Для моделирования сценария, при котором ресурс не выдерживал согласованное время выполнения, применялся следующий подход: к случайному числу из отрезка [1, 2500] сек прибавлялось реальное значение времени выполнения. Функция штрафа и репутация оценивались с использованием выражений (3) и (6) соответственно. Функция полезности (2) вычислялась с использованием в качестве параметра качества обслуживания QoS общее время выполнения задачи.

5.3. Анализ эффективности предложенного подхода

В приведенных экспериментах задачи распределялись по ресурсам сразу после поступления в систему. В рамках экспериментов сравнивались следующие планировщики с учетом и без учета репутации:

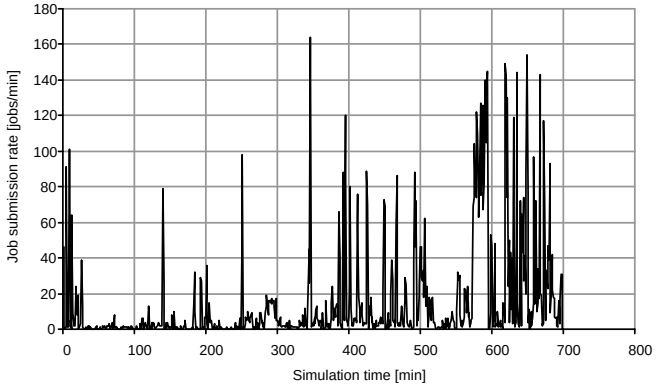


Рис. 3. Интенсивность поступления задач в систему

– Евристический планировщик, который отправляет задачу на ресурс, минимизирующий ожидаемое время выполнения задачи (*ECT*).

– Планировщик, интегрирующий репутацию на основе нелинейной схемы компромиссов (23). В проведенных экспериментах были установлены следующие значения параметров: $\alpha_1=\alpha_2=0.5$.

– Планировщик, интегрирующий репутацию с использованием мультипликативной схемы (24).

Эффективность планировщиков сравнивалась по следующим критериям:

– *время выполнения всех задач* (*Makespan)*: разница между временем завершения последней задачи и временем поступления первой задачи;

– *среднее время выполнения задачи* (*Average Job Execution Time*): среднее время, которое требуется для выполнения задачи;

– *среднее время ожидание задачи в очереди* (*Average Job Queue Waiting Time*);

– *среднее время превышения выполнения задачи* (*Average Job Excess Time*): среднее время превышения согласованного времени обслуживания (выполнения задачи).

– *количество задач, для которых предоставлен ненадежный сервис* (*SLA missed*), т.е. при обработке задачи не было предоставлено согласованное качество обслуживания;

– *среднее значение функции полезности* (*Average utility*).

–*использование ресурса* (*Resource utilization*): количество задач, выполненных данным ресурсом.

В рамках экспериментов варьировались следующие параметры: нагрузка (общее количе-

ство отправленных на выполнение задач) и надежность ресурса.

В первом случае, разное количество моделируемых задач соответствовало разной интенсивности (Табл. 1)

Табл. 1. Количество и интенсивность поступления задач в рамках проведенных экспериментов

Количество задач	Интенсивность поступления задач [задач/мин]
1000	4
2000	5
3000	7
4000	8
5000	9
6000	10
7000	11
8000	12
9000	13
10000	14

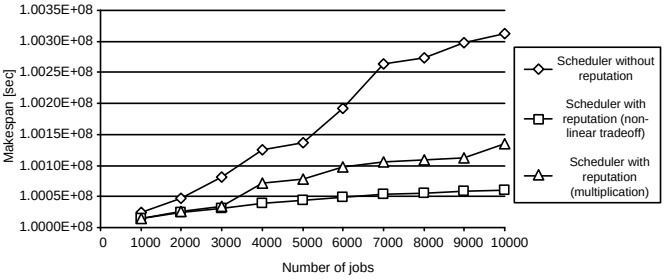


Рис. 4. Время выполнения задач для разных планировщиков (с и без учета репутации) в зависимости от нагрузки на систему

Из 20 вычислительных ресурсов 4 ресурса (20%) были установлены ненадежными, т.е. согласованный уровень качества услуг SLA всегда нарушался этими ресурсами. Начальная репутация этих ресурсов была равна 0.1. На рис. 4-10 показано сравнение планировщиков по разным критериям в зависимости от нагрузки на систему.

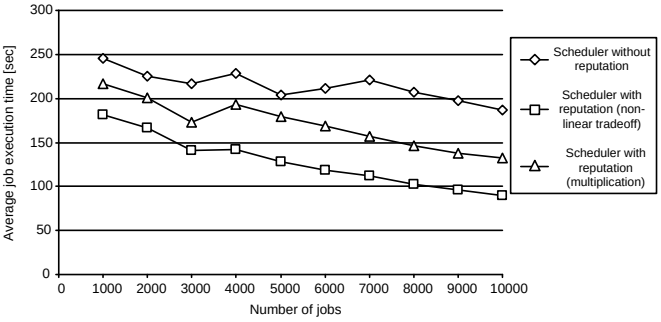


Рис. 5. Среднее время выполнения задачи для разных планировщиков (с и без учета репутации) в зависимости от нагрузки на систему

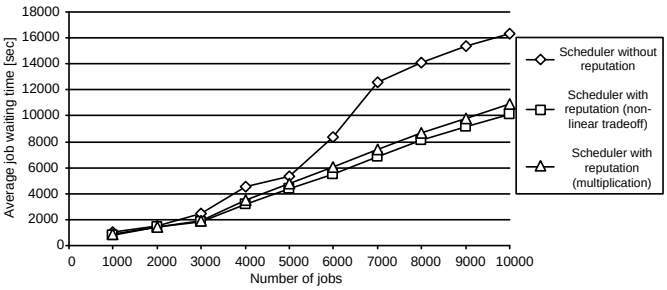


Рис. 6. Среднее время ожидания задачи в очереди для разных планировщиков (с и без учета репутации) в зависимости от нагрузки на систему

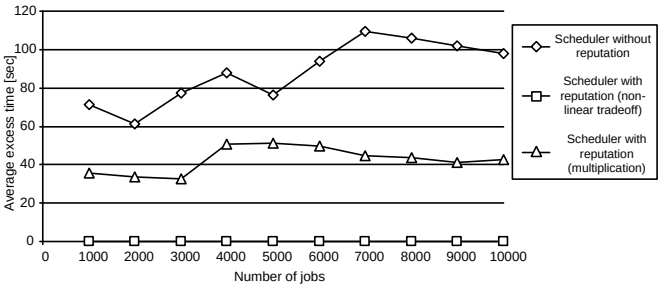


Рис. 7. Среднее время превышения выполнения задачи для разных планировщиков (с и без учета репутации) в зависимости от нагрузки на систему

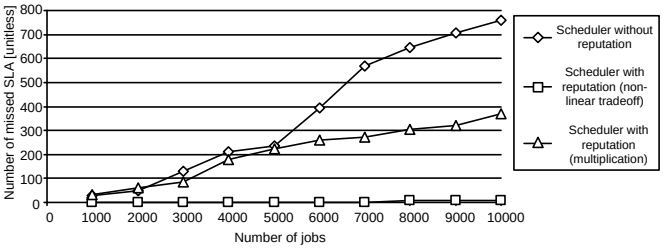


Рис. 8. Количество задач, для которых был предоставлен ненадежный сервис, для разных планировщиков (с и без учета репутации) в зависимости от нагрузки на систему

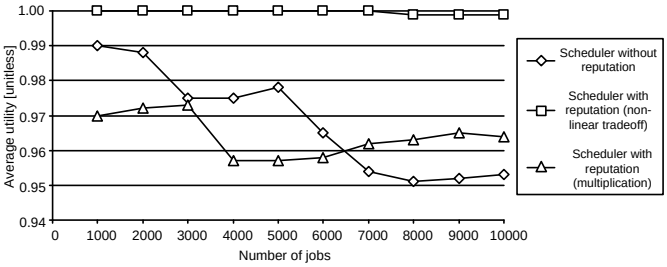


Рис. 9. Среднее значение функции полезности для разных планировщиков (с и без учета репутации) в зависимости от нагрузки на систему

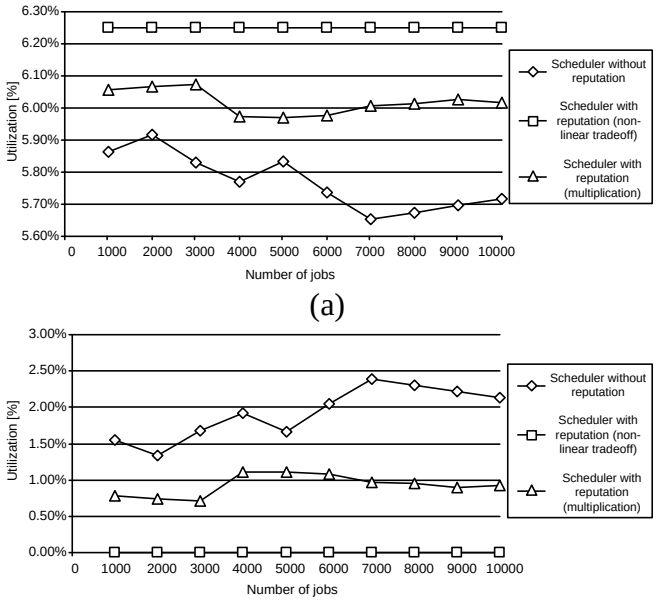


Рис. 10. Среднее количество выполненных задач для разных планировщиков (с и без учета репутации) в зависимости от нагрузки на систему (a – надежные ресурсы; b – ненадежные ресурсы)

Полученные результаты показали, что планировщик, интегрирующий репутацию на основе нелинейной схемы компромиссов, имеет большую эффективность по сравнению с другими планировщиками по всем критериям. Относительное улучшение значений критериев эффективности использования репутации в планировщике (с использованием обеих схем) по сравнению со стандартным планировщиком представлено в табл. 2. По сравнению с мультипликативной схемой использование нелинейной схемы компромиссов для интеграции репутации в планировщик задач позволило повысить эффективность его работы на 25%.

Стоит отметить, что для повышения эффективности при планировании задач репутация должна оцениваться на основе соответствующих параметров качества обслуживания. Иначе, повышение производительности по некоторым из приведенных выше критериев может не наблюдаться.

Табл. 2. Относительное повышение значений критериев эффективности при интеграции репутации в планировщик

Критерий эффективности	**Среднее улучшение [%]**	
	Нелинейная схема компромиссов	**Мультипликативная схема**
Среднее время выполнения	40.9	20.9
Среднее время ожидания	30.5	26.1
Среднее время превышения выполнения задачи	100.0	50.9
Количество задач, для которых предоставлен ненадежный сервис	99.6	26.9
Средняя значение функции полезности	3.2	0.4
Время выполнения всех задач	0.1	0.1
Усредненное значение по всем критериям	45.7	20.9

В рамках второй серии экспериментов варьировалась надежность ресурсов. Как и в первом случае, 20% ресурсов были установлены ненадежными, однако с разным *уровнем надежности* (*trustworthiness rate*). Например, если уровень надежности ресурса равен 0.6, это означает, что в среднем в 60% случаях заранее согласованный уровень обслуживания будет предоставлен. Для моделирования такой ситуации использовался следующий подход: в тех случаях, когда ненадежный ресурс выполнял задачу, генерировалось равномерно распеределенное случайное число из отрезка [0; 1]. Если это значение меньше, чем уровень надежности ресурса, тогда ресурс предоставлял согласованный уровень качества обслуживания SLA. В противном случае, предполагалось, что поставщик ресурсов нарушил согласованный уровень качества обслуживания SLA. На рис. 11-17 показано сравнение планировщиков по разным критериям в зависимости от среднего уровня надежности ненадежных ресурсов.

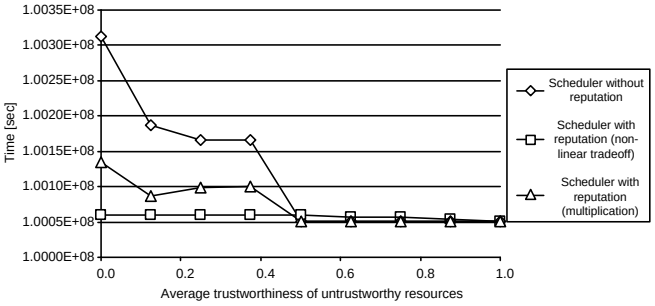


Рис. 11. Время выполнения задач для разных планировщиков (с и без учета репутации) в зависимости от среднего уровня надежности ненадежных ресурсов

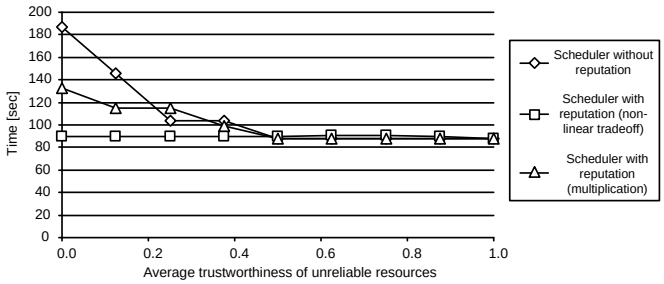


Рис. 12. Среднее время выполнения задачи для разных планировщиков (с и без учета репутации) в зависимости от среднего уровня надежности ненадежных ресурсов

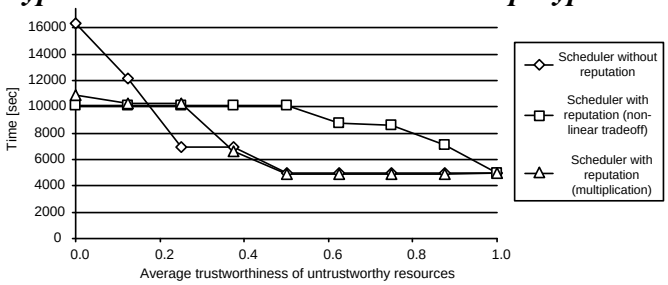


Рис. 13. Среднее время ожидания задачи в очереди для разных планировщиков (с и без учета репутации) в зависимости от среднего уровня надежности ненадежных ресурсов

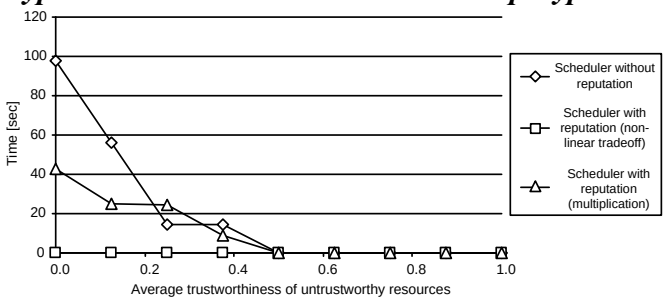


Рис. 14. Среднее время превышения выполнения задачи для разных планировщиков (с и без учета репутации) в зависимости от среднего уровня надежности ненадежных ресурсов

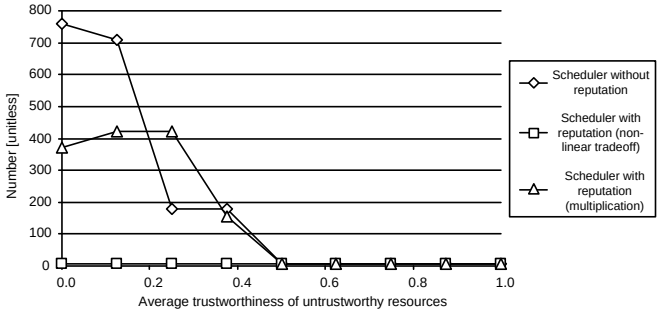


Рис. 15. Количество задач, для которых был предоставлен ненадежный сервис, для разных планировщиков (с и без учета репутации) в зависимости от среднего уровня надежности ненадежных ресурсов

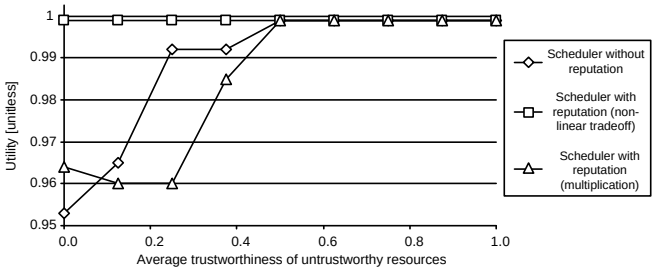
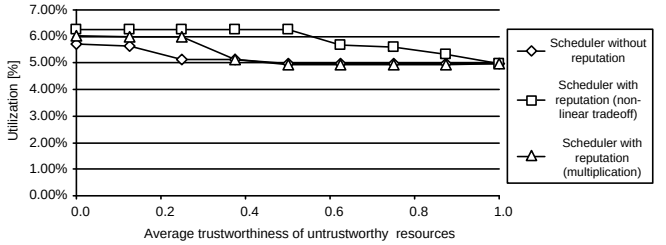
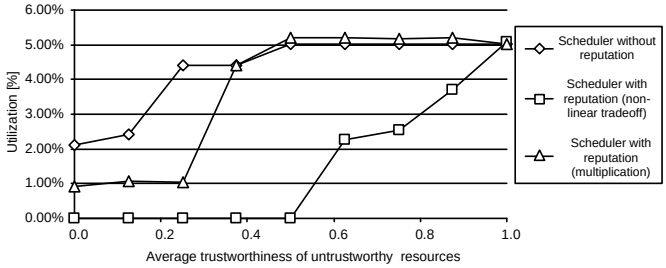


Рис. 16. Среднее значение функции полезности для разных планировщиков (с и без учета репутации) в зависимости от среднего уровня надежности ненадежных ресурсов



(a)



(b)

Рис. 17. Среднее количество выполненных задач для разных планировщиков (с и без учета репутации) в зависимости от среднего уровня надежности ненадежных ресурсов (a – надежные ресурсы; b – ненадежные ресурсы)

Полученные результаты показали, что планировщик, интегрирующий репутацию на основе нелинейной схемы компромиссов, обеспечил более эффективное распределение задач (рис. 11-17). Например, при его использовании задачи не отправлялись на ненадежные ресурсы до тех пор, пока средний уровень надежности ненадежных ресурсов не достиг значения 0.5 (рис. 17b). Более того, такой планировщик можно адаптировать, настраивая коэффициенты α_1 и α_2 в выражении (23).

6. Выводы

В данной работе решается задача многокритериальной оптимизации планирования выполнения задач в структурно-сложных системах на основе моделей репутации. Предложен новый подход для интеграции модели репутации в

подсистему планирования выполнения задач с использованием нелинейной схемы компромиссов. Преимуществом данной схемы является получение парето-оптимальных решений, обеспечивающих различные меры частичного удовлетворения критериев. Полученные экспериментальные результаты свидетельствуют о том, что планировщик, построенный на основе многокритериальной оптимизации с учетом модели репутации обеспечивает более высокую эффективность планирования по сравнению с другими планировщиками без учета репутации на 20%. В свою очередь использование нелинейной схемы компромиссов в задаче многокритериальной оптимизации позволило повысить эффективность планирования задач по сравнению с мультипликативной схемой еще на 25%. Таким образом, общее повышение эффективности планирования при использовании нелинейной схемы компромиссов с учетом модели репутации составляет примерно 45% по сравнению с традиционными системами планирования. Дополнительным преимцуществом предложенного планировщика с учетом модели репутации на основе нелинейной схемы компромиссов является повышение уровня информационной безопасности, поскольку при его использовании задачи не отправляются на ненадежные ресурсы до тех пор, пока средний уровень надежности ресурсов не достигает определенного значения.

Список литературы

1. Kussul N. Grid Technologies for Satellite Data Processing and Management within International Disaster Monitoring Projects / N. Kussul, A. Shelestov, S. Skakun [S. Fiore, G. Aloisio (eds.)] Grid and Cloud Database Management. – Springer-Verlag, Berlin Heidelberg, 2011, – P. 279-306.
2. Lecca G. Grid computing technology for hydrological applications / G. Lecca, M. Petitdidier, L. Hluchy [et al.] // J. of Hydrol. – 2011. – № 403(1-2). – P.186-199.
3. Kussul N. Grid and sensor web technologies for environmental monitoring / N. Kussul, A. Shelestov, S. Skakun // Earth Sci. Inf. – 2009. – №2(1-2). – P. 37-51.
4. Bouras C. e-Collaboration concepts, systems and applications / C. Bouras, E. Giannaka, E. Tsiatos // Information Science Reference E-Collaboration: Concepts, Methodologies, Tools, and Applications / Kock N. – IGI Global, 2009. – vol. 1, Section I, Chapter 1.2.
5. Foster I. The Anatomy of the Grid: Enabling Scalable Virtual Organizations / I. Foster, C. Kesselman, S. Tuecke // Int. J. of Supercomput. Appl. – 2001. – №15(3). – P. 200-222.
6. Eymann T. A Framework for Trust and Reputation in Grid Environments / T. Eymann, S. König, R. Matros // J. of Grid Comput. – 2008. – №6(3). – P. 225-237.
7. Song S. Trusted Grid Computing with Security Binding and Trust Integration / S. Song, K. Hwang, Y.-K. Kwok // J. of Grid Comput. – 2005. – №3(1-2). – P. 53-73.
8. Chakrabarti A. Grid Computing Security / Chakrabarti Anirban. - Berlin Heidelberg, Springer-Verlag. – 2007. – 331 p.
9. Silaghi G. A Utility-Based Reputation Model for Service-Oriented Computing / G. Silaghi, A. Arenas, L. Silva // Toward Next Generation Grids / Priol, T., Vanneschi, M. – Springer, 2007. – P. 63-72
10. Grandison T. A Survey of Trust in Internet Applications / T. Grandison, M. Sloman // IEEE Commun. Surv. and Tutor. – 2000. – №4(4). – P. 2-16.
11. The Meaning of Trust / D.H. McKnight, N.L. Chervany// Technical Report MISRC Working Paper Series 96-04. – University of Minnesota. Management Information Systems Research Center. – 1996.
12. Abdul-Rahman A, Supporting trust in virtual communities / A. Abdul-Rahman, S. Hailes // Proc. of the IEEE 33rd Hawaii Int. Conf. on Syst. Sci. (HICSS ’00). – 2000. – vol. 6. – P. 6007.
13. Arenas A. Reputation management in grid-based virtual organisations / A. Arenas, B. Aziz, G.C. Silaghi // Proc. Int. Conf. on Secur. and Cryptogr. (SECRYPT 2008). – 2008. – P. 538-545.
14. Куссуль О.М. Исследование эффективности применения моделей доверия на основе репутации в Grid-системах / О.М. Куссуль, А.Н. Новиков, С.С. Швец // Наукові праці ДонНТУ Серія "Інформатика, кібернетика та обчислювальна техніка". – 2010. – Випуск 12(165). – С. 126-134.
15. Kussul O. Utility-based reputation model for VO in Grids / O. Kussul, O. Novikov // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – 2009. – №50. – С. 137-145.
16. Azzedin F. Integrating Trust into Grid Resource Management Systems / Azzedin F., Maheswaran M. // Int. Conf. on Parallel Process. (ICPP 2002). – IEEE Computer Society Washington, DC, USA, 2002. – P. 47-54.
17. Gomez Marmol, F. Security threats scenarios in trust and reputation models for distributed systems / F. Gomez Marmol, G. Martınez Perez // Comput. & Secur. – 2009. – №28. – P. 545-556.

18. Kussul N. Grid system for flood extent extraction from satellite images / Kussul N., Shelestov A., Skakun S. // Earth Sci. Inf. – 2008. - №1(3–4). – P. 105–117.
19. Shelestov A. Intelligent Model of User Behaviour in Distributed Systems / A. Shelestov, S. Skakun, O. Kussul // Int. J. on Inf. Theory and Appl. – 2008. – №15(1). – P. 70-76.
20. Kussul N. Intelligent System for Users' Activity Monitoring in Computer Networks / N. Kussul, S. Skakun // IEEE Intelligent Data Acquisition and Advanced Computing Systems (IDAACS 2005): Technology and Applications. – 2005. – P. 306-309.
21. Skakun S. Implementation of the Neural Network Model of Users of Computer Systems on the Basis of Agent Technology / S. Skakun, N. Kussul, A. Lobunets // J. of Autom. and Inf. Sci. – 2005. – №37(4). – P. 11-18
22. Kussul N. Neural Network Approach for User Activity Monitoring in Computer Networks / N. Kussul, S. Skakun // Proc. of the Int. Joint Conf. on Neural Networks (Budapest, Hungary). – 2004. – vol. 2. – P. 1557-1562.
23. Voronin A.N. A multicriteria problem of distribution of bounded resources / A.N. Voronin // Cybern. and Syst. Anal. – 2011. – №47(3). – P. 490-493.
24. Papaioannou T.G. Reputation-Based Estimation of Individual Performance in Grids / T.G. Papaioannou, G.D. Stamoulis // Eighth IEEE Int. Symp. on Cluster Comput. and the Grid (CCGRID), . – IEEE Computer Society Washington, DC, USA, 2008. – №47(3). – P. 500-509.
25. Liang Z. A reputation-driven scheduler for autonomic and sustainable resource sharing in Grid computing / Z. Liang, W Shi. // J. of Parallel and Distrib. Comput. – 2010. – №70. – P. 111-125.
26. Wu C.-C., An integrated security-aware job scheduling strategy for large-scale computational grids / Wu C.-C., Sun R.-Y. // Future Generation Comput. Syst. – 2010. – №26. – P. 198-206.
27. PathTrust: A Trust-Based Reputation Service for Virtual Organization Formation / F. Kerschbaum, J. Haller, Y. Karabulut, P. Robinson // Lect. Notes in Comput. Sci. – 2006. – №3986/2006. – P. 193-205.
28. von Laszewski G. Towards Reputable Grids. Scalable Comput / G. von Laszewski, B. Alunkal, I Veljkovic. // Pract. and Exp. – 2005. – №6(3). – P. 95-106.
29. Kamvar S. The EigenTrust algorithm for reputation management in P2P networks / S. Kamvar, M. Schlosser, H. Garcia-Molina // Proc. of the 12th Int. Conf. on World Wide Web. – ACM Press, New York, NY, USA, 2003. – №6(3). – P. 640-651.
30. Intrusion Detection for Grid and Cloud Computing / K. Vieira, A. Schulter, C. Westphall // IT Prof. – 2007. – №12(4). – P. 38-43.
31. Intrusion Detection for Computational Grids / A. Schulter [et al.]// Proc. 2nd Int. Conf. New Technol., Mobil., and Secur. – IEEE Press, 2008. – №22(7). – P. 1-5.
32. Shelestov A. Grid Technologies in Monitoring Systems Based on Satellite Data / A. Shelestov, N. Kussul, S. Skakun // J. of Autom. and Inf. Sci. – 2006. – №38(3). – P. 69-80.
33. Oh S.H. An anomaly intrusion detection method by clustering normal user behaviour / S.H. Oh, W.S. Lee // Comput. & Secur. – 2003. – №22(7). – P. 596-612.
34. Anomaly Detection in Grid Computing Based on Vector Quantization / H.-W. Sun, K.-Y. Lam, S.-L. Chung [et al.] // Lect. Notes in Comput. Sci. – 2004. – №3251/2004. – P. 883-886.
35. Haykin S. Neural Networks: A Comprehensive Foundation / Haykin S. – Saddle River, New Jersey, Prentice Hall, 1999. – 842 p.
36. Kretsis A. Developing Scheduling Policies in gLite Middleware / A. Kretsis, P. Kokkinos, E. Varvarigos // Proc. of the 9th IEEE/ACM Int. Symp. on Cluster Comput. and the Grid. – IEEE Computer Society Washington, DC, USA, 2009. –P. 20-27.
37. The Grid Observatory / C. Germain-Renaud, A. Cady, P. Gauron [et al.] // 11th IEEE/ACM Int. Symp. on Cluster, Cloud and Grid Comput., CCGrid. – 2011. – P. 114–123.

ДК 621.372

КОРНАЧЕВСЬКИЙ Я.І.,
ЛАДОГУБЕЦЬ В.В.

ГЕНЕТИЧНІ АЛГОРИТМИ В САПР

Розглядаються особливості паралельної реалізації генетичних алгоритмів для розв'язання задачі оптимізації. Дана оцінка алгоритмів з точки зору їх застосування в різних галузях САПР.

Article is de voted to the features of parallel implementation of genetic algorithms for solution of the optimization problem. Application of these algorithms for CAD systems is evaluated.

Вступ

Генетичні алгоритми (ГА) використовуються в САПР не так часто (порівняно з іншими алгоритмами). Разом з цим, ГА мають багато переваг (наприклад, відсутність обмежень на характер цільової функції, зокрема нечутливість до "яружності" та перервності), які роблять їх дуже зручними для задач, які характерні для систем автоматизованого проектування. Тому оцінка можливості застосування ГА для САПР є дуже важливою.

1. Елементи еволюційної теорії

Генетичні алгоритми були запропоновані в 1975 році Джоном Голландом [1]. Прототипом обрано природний добір, який в рамках еволюційної теорії постійно покращує види, або, умовно кажучи, перетворює їх до оптимальної форми. Суть природного добору полягає у тому, що найбільш пристосовані представники краще виживають і приносять більше потомства, ніж менш пристосовані. Відзначимо, що сам по собі природний відбір ще не забезпечує розвитку біологічного виду. Справді, якщо припустити, що всі нащадки народжуються приблизно однаковими, то різні покоління будуть відрізнятися тільки по чисельності, але не по пристосованості. Тому дуже важливо вивчити, яким чином відбувається спадкування, тобто як властивості нащадка залежать від властивостей батьків.

Основний закон спадкування полягає в тому, що нащадки схожі на батьків. Зокрема, нащадки більш пристосованих батьків будуть, напевне, одними з найбільш пристосованих у своєму поколінні.

Майже в кожній клітині тварини є набір хромосом, що несуть у собі всю спадкову інформацію. Інформація кодується генами – відрізками ланцюга ДНК, відповідальними за певну визначену властивість особини, наприклад, за колір очей чи тип волосся. В процесі утворення клітини нового організму відбувається таке: нитки ДНК розриваються в декількох випадкових місцях і хромосоми обмінюються своїми частинами (рис 1). Цей процес забезпечує появу нових варіантів хромосом і називається "кросинговер".

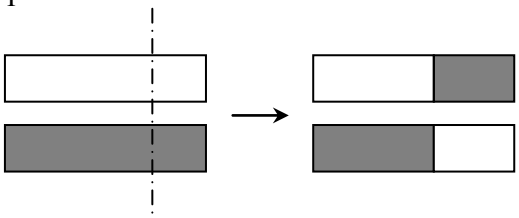


Рис. 1. Умовна схема кросинговеру

На спадковість впливають також мутації, які виражаються в зміні деяких ділянок ДНК. Мутації випадкові.

2. Задача оптимізації

Як уже було відзначено вище, еволюція – це процес постійної оптимізації біологічних видів. Природний добір гарантує, що найбільш пристосовані особини дадуть багато нащадків. А завдяки генетичному спадкуванню ми можемо бути впевнені, що частина цих нащадків не тільки збереже високу пристосованість батьків, але буде володіти і деякими новими властивостями. Якщо ці нові властивості виявляться корисними, то, з великою імовірністю, вони перейдуть і в наступне покоління. Таким чином, відбувається нагромадження корисних якостей і поступове підвищення пристосовності біологічного виду в цілому. Знаючи, як розв'язується задача оптимізації видів у природі, ми тепер застосуємо схожий метод для вирішення різних реальних задач.

Введемо позначення і приведемо кілька класичних прикладів. Як правило, в задачі оптимізації ми можемо керувати декількома параметрами (позначимо їх через x_1, x_2, ..., x_n), а нашою метою є максимізація (чи мінімізація) деякої цільової функції, $f(x_1, x_2, ..., x_n)$, що залежить від цих параметрів. Наприклад, якщо потрібно максимізувати цільову функцію "прибуток компанії", то керованими параметрами будуть число співробітників компанії, обсяг виробництва, витрати на рекламу, ціни на кінцеві продукти і т.д. Важливо відзначити, що ці параметри зв'язані між собою – наприклад, при зменшенні числа співробітників швидше всього зменшиться й обсяг виробництва.

Звичайно, математики здавна займалися подібними задачами. Розроблено багато методів їхнього рішення. У випадку, якщо цільова функція досить гладка і має тільки один локальний мінімум (унімодальна), то оптимальне рішення можна одержати градієнтними або квазіньютонівськими методами. Недоліком таких підходів є порівняно невелика швидкість збіжності, досить велика трудоємкість одного кроку, порівняно жорсткі вимоги до властивостей цільових функцій Аналогічні проблеми виникають і з застосуванням інших математичних методів. У багатьох важливих задачах параметри можуть приймати лише визначені значення, причому, у всіх інших точках цільова функція не визначена. Звичайно, у такому випадку не може бути і мови про її гладкість. Тому вимагаються принципово інші підходи.

Найпростіший спосіб знайти оптимальне рішення – перебрати всі можливі значення параметрів. При цьому не потрібно робити ніяких припущень про властивості цільової функції, а задати її можна просто за допомогою таблиці. Однак, щоб вирішити таким способом задачу комівояжера хоча б для 20 міст, буде потрібно перебрати близько 10^{19} маршрутів, що зовсім нереально для будь-якого обчислювального центру. Отже, виникає необхідність у якому-небудь новому методі оптимізації, придатному для практики. Покажемо, як можна застосувати механізми еволюційного процесу до таких задач. Фактично ж організуємо штучну еволюцію в спеціально побудованому світі.

3. Робота ГА

Уявимо собі штучний світ, населений безліччю істот (осіб), причому кожна істота – це деяке рішення нашої задачі. Будемо вважати особу тим більше пристосованою, чим краще відповідне рішення (чим більше значення цільової функції воно дає). Тоді задача максимізації цільової функції зводиться до пошуку найбільш пристосованої істоти. Звичайно, ми не можемо оселити в наш віртуальний світ всі істоти відразу, тому що їх дуже багато. Замість цього ми будемо розглядати багато поколінь, що змінюють одне одного. Тепер, якщо ми зуміємо ввести в дію природний добір і генетичне спадкування, то отриманий світ буде підкорятися законам еволюції. Відзначимо, що, відповідно до нашого визначення пристосованості, метою цієї штучної еволюції буде саме створення найкращих рішень. Очевидно, еволюція – нескінченний процес, у ході якого пристосованість осіб поступово підвищується. Примусово зупинивши цей процес через досить довгий час після його початку і вибравши найбільш пристосовану особу у поточному поколінні, ми одержимо не абсолютно точний, але близький до оптимального розв'язок. Перейдемо тепер до точних визначень і опишемо робо-

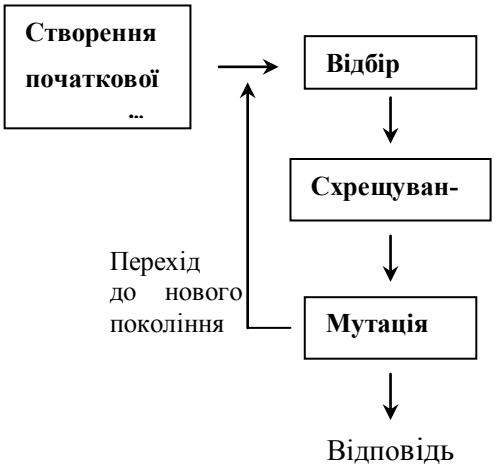


Рис. 2. Блок-схема генетичного алгоритму

ту ГА більш детально.

Для того, щоб говорити про генетичне спадкування, потрібно постачати наші істоти хромосомами. У ГА хромосома – це деякий числовий вектор, що відповідає параметру, який підбирається, а набір хромосом даної особи визначає рішення задачі. Які саме вектори варто розглядати в конкретній задачі, вирішує сам користувач. Кожна з позицій вектора хромосоми називається геном.

Визначимо тепер поняття, які відповідають мутації і кросинговеру в ГА.

Мутація – це перетворення хромосоми, що випадково змінює одну чи декілька її позицій

(генів). Найбільш розповсюджений вид мутацій – випадкова зміна лише одного з генів хромосоми [1, 2].

Кросинговер (у літературі по ГА також вживається назва **кросовер** чи **схрещування**) – це операція, при якій із двох хромосом породжується одна чи декілька нових хромосом. У найпростішому випадку кросинговер у ГА реалізується так само, як і в біології (див. рис. 1). При цьому хромосоми розрізаються у випадковому місці і обмінюються частинами між собою. Наприклад, якщо хромосоми (1, 2, 3, 4, 5) і (0, 0, 0, 0, 0) розрізати між третім і четвертим генами і обміняти їх частини, то вийдуть нащадки (1, 2, 3, 0, 0) і (0, 0, 0, 4, 5).

Блок-схема ГА зображена на рис. 2. Спочатку генерується початкова популяція осіб (індивідуумів), тобто деякий набір рішень задачі. Як правило, це робиться випадковим способом. Потім ми повинні змоделювати розмноження всередині цієї популяції. Для цього випадково відбираються кілька пар індивідуумів, відбувається схрещування між хромосомами в кожній парі, а отримані нові хромосоми вміщуються в популяцію нового покоління. У ГА зберігається основний принцип природного добору – чим більш пристосований індивідуум (чим більше відповідне йому значення цільової функції), тим з більшою імовірністю він буде брати участь у схрещуванні. Тепер моделюються мутації – у декількох випадково обраних особах нового покоління змінюються деякі гени. Потім стара популяція частково чи цілком знищується і ми переходимо до розгляду наступного покоління. Популяція наступного покоління в більшості реалізацій ГА містить стільки ж осіб, скільки й початкова, але в силу добору пристосованість у ній у середньому вище. Тепер описані процеси добору, схрещування і мутації повторюються уже для цієї популяції і т.д.

4. Особливості ГА для САПР

Найбільш істотні особливості задача параметричної оптимізації наступні:

1. Складний багатоекстремальний характер цільових функцій. Найчастіше це приводить до того, що значення цільової функції у знайденій точці локального оптимуму не задовольняє вимогам завдання, і виникає задача визначення координат іншого, більш придатного локального екстремуму. Крім цього, виникає велика залежність результату оптимізації від координат початкової точки оптимізації.

2. Складність одержання аналітичної залежності цільової функції від варійованих параметрів, що виключає можливість використання прямих методів визначення екстремуму.

3. Істотний перепад в абсолютних значення варійованих параметрів (наприклад, ємності можуть мати значення порядку 10^{-9}-10^{-12}, тоді як опір – 10^{3}-10^{6}). Це висуває підвищені вимоги до чисельної стійкості процедур, які використовуються при оптимізації і нерідко вимагає застосування спеціальних методів масштабування.

4. Неможливість або дуже висока трудомісткість оцінки значень похідних цільової функції. Це ускладнює застосування для пошуку оптимальних точок швидкозбіжних процедур, які використовують градієнти та похідні вищих порядків.

5. Яружний характер цільових функцій електронних схем вимагає розробки підходів для просування "вздовж ярів", а ще краще – усунення такої залежності.

6. Необхідність працювати з цільовими функціями для задач, які взагалі важко формалізувати, і які внаслідок цього вимагають розробки спеціальних методів.

З появою потужних багатоядерних обчислювальних систем виникла можливість досить ефективно використовувати методи випадкового пошуку. Однак останні мають суттєві обмеження [3, 4], тому пропонується побудувати оптимізаційну процедуру на основі генетичного алгоритму, який буде працювати на системах із масовим паралелізмом (MPP), зокрема на кластерних системах.

Оскільки основна частина часу в САПР витрачається на розрахунок цільових функцій, пропонується розробити планувальник, який буде реалізовувати генетичний алгоритм і завантажувати вузли кластеру розрахунком цільових функції.

Будемо діяти таким чином: нехай у розмір популяції N, а кількість процесів (ядер, вузлів) – p. Організуємо ітераційний процес, як зображено на рис. 3.

Після старту процедури організуємо цикл від 1 до N з кроком p (блок 2). Для кожного процесу з k=2,p (блок 3) передамо значення вектора X для розрахунку цільової функції (блок 4). Потім запустимо процес розрахунку цільової функції для мастер-процесу (блок 5). Після цього

зберемо дані від процесів з k=2,p(блоки 6, 7) і запишемо їх у масив значень цільових функцій індивідів.

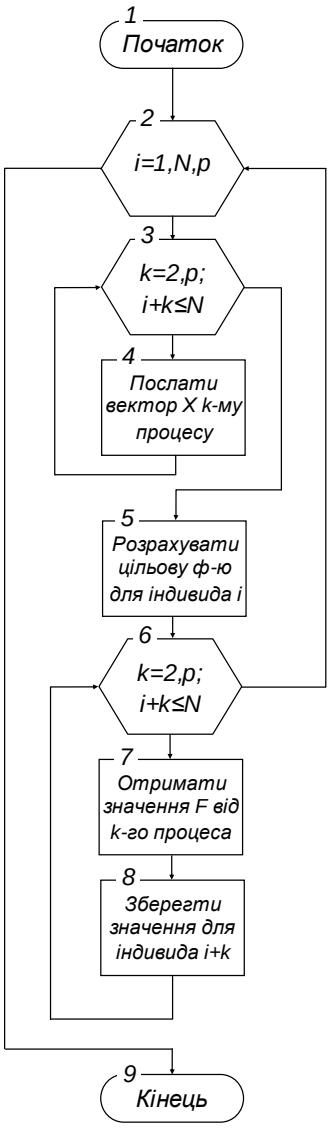


Рис. 3. Схема роботи паралельного ГА

Для пересилання даних використано функції MPI_Send() та MPI_Recv(), Спочатку здійснюється обмін даними розміром NV· ·MPI_DOUBLE, де NV – кількість параметрів цільової функції (MPI_DOUBLE – розмір даних подвоєної точності), потім – 1·MPI_DOUBLE, оскільки назад пересилається лише значення цільової функції. Таким чином, при роботі з p процесами на кожній ітерації ми пересилаємо (NV·MPI_DOUBLE+MPI_DOUBLE)(p-1) = MPI_DOUBLE·(1+ NV)(p-1) байт. Навіть при роботі з відносно повільною електронною мережею затрати на пересилання такої кількості даних мізерно малі порівняно з можливим часом розрахунку цільової функції. Однак слід зазначити, що перед початком розрахунку необхідно на кожен вузол завантажити всі необхідні для початку моделювання дані (наприклад, опис оптимізованої схеми). Цей процес може зайняти чималий час, хоча і виконується лише один раз.

Перед запуском генетичного алгоритму необхідно розрахувати кілька службових змінних. Перша – це довжина хромосоми, точніше довжини генів, які припадають на кожну змінну (рис. 4).

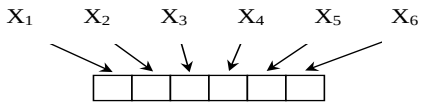


Рис. 4. Хромосома

Перед стартом користувач задає точність дискретизації ділянки пошуку, діапазон зміни кожного параметра (наприклад, ε=0.001, [-2…3; -1...8]). На основі цих даних знаходиться ширина ділянки пошуку (у прикладі відповідно 3-(-2) = 5; 8-(-1) = 9), далі розраховується кількість відліків (5/ 0.001 = 5000; 9/ 0.001 = 9000 відліків), а потім – довжини генів (log 2 (5000) ≈12,287 → 13; log 2 (9000) ≈13,135 → 14). Довжина хромосоми очевидно буде сумою довжин генів.

Оскільки майже завжди довжина генів в хромосомі буде надлишковою для заданого діапазону (наприклад, 213 = 8192), то необхідно перерозрахувати значення ε для кожного гена так, щоб максимальне його значення відповідало правому кінцю діапазону зміни параметра (наприклад, εн = 5/8192 = 0,00061).

Генерація стартової популяції полягає в генерації певної кількості індивідів. Після генерації кожного індивіду відбувається перевірка на його присутність в популяції. Якщо такий індивід в популяції вже існує – він не додається. Замість "дубліката" генерується новий індивід. Побудова індивіда полягає формуванні хромосоми. Побудова гена хромосоми полягає в генерації випадкового числа (від 0 до кількості відліків для поточного гена), перетворенні цього числа в код Грея і в подальшому записі його на позицію, яка відповідає позиції гена в хромосомі.

Після формування популяції і оцінки функції пристосованості індивіди сортуються по спаданню своєї пристосованості. Після цього починається операція кросинговеру.

Спочатку в нову популяцію записується певна кількість кращих старих індивідів (для забезпечення стратегії елітизму). Потім відбирається два різних індивіда, випадковим чином визначається позиція в хромосомі і відбувається-

ся схрещування – обмін бітами (одноточковий кросинговер, див. Розділ 2).

Відбір виконується так, щоб з більшою ймовірністю в подальшому кросинговері приймали участь ті індивіди, які мають кращу пристосованість. Для цього рівномірний закон розподілу ймовірності (вбудований в математичну бібліотеку С) перетворено на гіперболічний. Типова частота використання індивідів приведена на рис. 5.

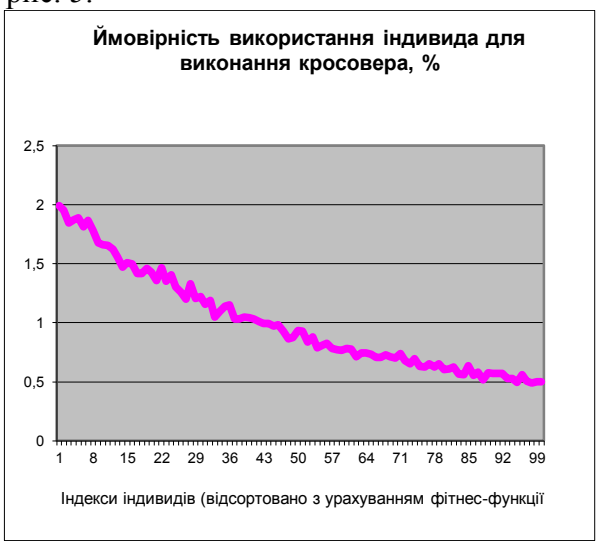


Рис. 5. Типова частота використання індивідів в залежності від фітнес-функції

Після кросинговеру генерується випадкове число (ймовірність мутації), і, якщо воно менше порогу мутації, один з нащадків мутує (двійкова мутація, див. розділ 2). Причому, якщо це число вдвічі менше порогу мутації, то мутує перший нащадок, інакше – другий. Якщо обидва індивіди нові (не існують в новій популяції), то їх додають в нову популяцію.

Індивіди сортуються і алгоритм знову готовий розпочати генерацію нової популяції. Але перед цим відбувається перевірка на значення кращої хромосоми. Якщо хромосома не змінюється протягом певної кількості ітерацій, то, очевидно, що це значення і є розв'язком. У такому випадку алгоритм зупиняє розрахунок і видає результати користувачеві.

5. Тестування ГА для САПР

Для тестування ефективності алгоритму запропоновано виконати оптимізацію для тестової функції (час на розрахунок одного значення цільової функції) і визначити витрати часу для різної кількості процесів. Для симуляції тривалого часу розрахунку складної схеми в програму введено спеціальну змінну, яка уповільнює розрахунок цільової функції. При цьому було отримано наступні результати (при роботі на дво- та чотириядерній платформі):

Табл. 1. Двоядерна платформа

1 потік, с	2 потоки, с	прискорення, разів
0,089	0,315	0,283
0,092	0,315	0,292
0,291	0,332	0,877
0,869	0,630	1,379
1,830	1,052	1,740
3,207	1,804	1,778
4,930	2,670	1,846
20,633	10,506	1,964
80,003	40,196	1,990

Рис. 6. Коефіцієнт прискорення ГА (2 ядра)

Табл. 2. Чотириядерна платформа

Кількість потоків / час виконання (секунди)				Кількість потоків / прискорення (рази)		
1	2	3	4	2	3	4
1,00	0,62	0,50	0,45	1,61	2,02	2,25
2,00	1,13	0,84	0,70	1,77	2,38	2,86
3,00	1,64	1,20	0,96	1,83	2,50	3,13
4,00	2,16	1,53	1,23	1,85	2,61	3,25
5,00	2,59	1,87	1,42	1,93	2,67	3,52
10,00	5,11	3,57	2,68	1,96	2,80	3,73
20,00	10,43	7,18	5,34	1,92	2,79	3,75
30,00	15,06	10,32	7,66	1,99	2,91	3,92
40,00	20,15	13,72	10,16	1,99	2,92	3,94
50,00	25,07	17,10	12,67	1,99	2,92	3,95

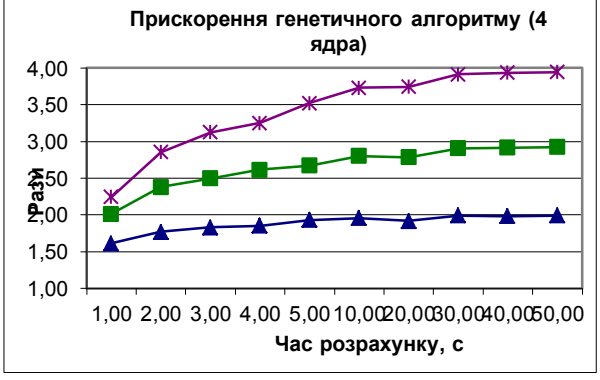


Рис. 7. Коефіцієнт прискорення ГА (4 ядра)

Висновки

Приведені результати повністю підтверджують закон Амдала – при малому часі розрахунку цільових функцій значну частину загального часу займає сам генетичний алгоритм і обмін даними між процесами. Коли ж час розрахунку великий, то прискорення досягає практично теоретичної межі, що свідчить про правильність припущень і запропонованих та реалізованих підходів до побудови паралельного генетичного алгоритму.

Список літератури

1. John H. Holland. Adaptation in Natural and Artificial Systems / John H. Holland. – Ann Arbor : University of Michigan Press, 1975. – 183 p.
2. Goldber, David E. Genetic algorithms in search, optimization, and machine learning / Goldber, David E. – Addison-Wesley Publishing Company, Inc, 1989. – 414 p.
3. Cantu-Paz E. Efficient and accurate parallel genetic algorithms / Cantu-Paz E. – Springer, 2000. – 162 p.
4. Lance Chambers. The Practical Handbook of GENETIC ALGORITHMS / Lance Chambers. – CRC Press, Inc, 1998. – 592 p.

УДК 004.9

ЗАЙЧЕНКО Ю.П.,
МУРГА М.О.

УДОСКОНАЛЕННЯ МЕТОДУ ОПТИМІЗАЦІЇ НЕЧІТКОГО ФОНДОВОГО ПОРТФЕЛЮ З НОВИМИ ФУНКЦІЯМИ РИЗИКУ

Розглядається задача оптимізації інвестиційного портфелю математична модель якого побудована з використанням теорії нечітких множин. Досліджено питання вигляду функцій ризику, прискорення оптимізації та оптимізації в умовах, коли множина розв'язків може містити більше, ніж один елемент.

A problem of investment portfolio optimization which mathematical model is built using fuzzy sets theory is investigated. Problems of risk function views, optimization's acceleration, and optimization in conditions of more than one-element solution set are studied.

Вступ

Розвиток ринкових відносин, пришвидшення укладання торгівельних угод, ускладнення економічної ситуації у світі зумовлюють перегляд моделей функціонування економіки. Класичні підходи, котрі ґрунтуються на симетричному вигляді функцій корисності різних економічних сутностей поступово відходять на задній план. Економіка сьогодні – це повна невизначеність ситуацій та суперечність інформації, що циркулює у ній. Саме ці міркування зумовлюють все глибше проникнення у економічний аналіз моделей, що будуються на підставі нечіткої логіки. Не оминув цей процес і методики формування інвестиційного портфелю. Теорія нечіткого інвестиційного портфелю вже описана у багатьох роботах. Досліджено ряд важливих питань, як то: вигляд прямої та двоїстої задачі оптимізації [1-7], властивості залежності ризик-дохідність інвестиційного портфелю [1-7], багатокритеріальна задача оптимізації портфелю, де в якості критеріїв виступає як дохідність портфелю, так і його ризик [6], вивчена множина розв'язків задач оптимізації [7], показано, що у загальному випадку задача оптимізації багатоінструментного портфелю може бути зведена до портфелю, що оперує двома фінансовими інструментами [7].

Незважаючи на вже досить солідний термін дослідження проблеми нечіткого інвестиційного портфелю та великий об'єм роботи, що була зроблена, ще залишається відкритим ряд питань. Це: що таке функція ризику нечіткого інвестиційного портфелю, та які ще різновиди функцій ризику нечіткого інвестиційного портфелю можна розглядати? Чи існує простий у обчислювальному сенсі критерій, щоб дозволив визначати вид залежності ризик-дохідність у випадку двох активів без побудови сітки значень? Це б дозволило отримувати розв'язки багатьох задач оптимізації, не розв'язуючи їх. І, нарешті, у всіх методах оптимізації, які використовуються на даний момент для формування нечіткого портфелю, неявно припускається, що множина розв'язків складається з одного елемента. Але, як було показано в роботі [7], це не так. Тому останнє питання, на яке відповідають автори роботи: це вигляд методу оптимізації інвестиційного портфелю, за умови, коли множина розв'язків складається більш, ніж з одного елемента?

Таким чином, метою роботи є розробка швидкого методу оптимізації нечіткого інвестиційного портфелю, за умови існування в множині оптимальних розв'язків задачі

більше одного елемента.

Структура роботи побудована наступним чином. Стаття містить три частини. В першій частині розглядаються питання фізичного змісту функції ризику нечіткого інвестиційного портфелю та вивчаються її властивості. В другій частині вводиться простий в обчислювальному сенсі критерій визначення вигляду залежності ризик-дохідність портфелю. В третій частині описаний метод формування інвестиційного портфелю за умови, що множина розв'язків задачі оптимізації може містити більш ніж один елемент.

1. Постановка оптимізаційної задачі. Функції ризику

В формальному вигляді задача оптимізації інвестицій в різні активи має вигляд **(1)**:

$$\begin{cases} \sum_{i=1}^{n} \bar{r}_i x_i \to \max, \\ \beta\left(x, r, r^*\right) \le \beta_1, \\ \sum_{i=1}^{n} x_i = 1, \\ x_i \ge 0, \forall i = 1...n. \end{cases} \quad \textbf{(1)}$$

Цільова функція містить наступні компоненти: $\bar{r}_i$ – очікувана дохідність від інвестиції у i-й актив у прогнозі на проміжок часу T, x_i – доля інвестицій у i-й актив від загальної суми інвестування, а n – загальна кількість активів. Таким чином, критерій – максимізація очікуваної дохідності всього інвестиційного портфелю.

Перше обмеження означає вимогу того, щоб ризик неефективного інвестування не перевищував певний рівень. Останні два обмеження описують долі інвестування загальної суми у фінансові інструменти

Далі мова буде йти про нечітку портфельну теорію з функціями належності дохідностям окремих активів трикутного вигляду, і тому функцію ризику можна записати наступним чином **(2)**:

$$\beta\left(x, r, r^*\right) = \beta\left(x_1,..., x_n, r_{\min 1},..., r_{\min n}, \bar{r}_1,..., \bar{r}_n, r_{\max 1},..., r_{\max n}, r^*_{\min}, \bar{r}^*, r^*_{\max}\right). \quad \textbf{(2)}$$

Окремого роз'яснення в формулі **(2)** вимагає $\left(r_{\min i}, \bar{r}_i, r_{\max i}\right)$ – позначення нечіткого (трикутного) числа дохідності i-го активу – відповідно мінімальне значення, найбільш очікуване значення та максимальне значення. $\left(r^*_{\min}, \bar{r}^*, r^*_{\max}\right)$ – трикутне нечітке число критеріального значення дохідності – значення дохідності портфелю, нижче якого портфель вважається неефективним.

Останні два обмеження позначають, що x_i – доля інвестицій у i-й актив.

В основі визначення функції ризику нечіткого портфелю лежить вірогідність потрапляння портфелю в зону неефективних вкладень.

На Рис. 1 зображене розташування функцій належностей критеріального значення дохідності портфеля та дохідності, що очікується.

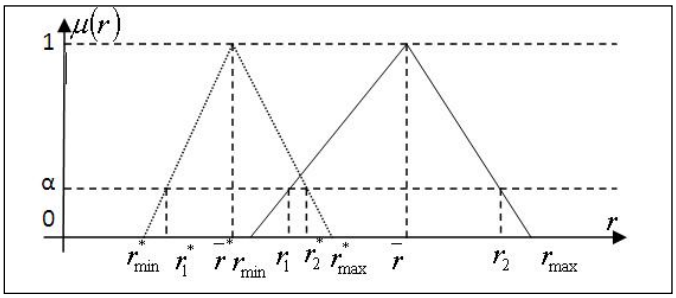


Рис. 1. Ліворуч – функція належності критеріального значення; праворуч – дохідності портфеля.

$$r_{\min} = \sum_{i=1}^{n} x_i r_{\min i}, \; r_{\max} = \sum_{i=1}^{n} x_i r_{\max i}, \; \bar{r} = \sum_{i=1}^{n} x_i \bar{r}_i.$$

Взагалі кажучи, взаємне розташування на площині графіків Рис. 1 можливо яке завгодно. Тут приводиться даний випадок для наочності.

Зрізу рівня α на Рис. 1 та у його позначеннях на площині значення дохідності портфеля – критеріальне значення відповідає Рис. 2.

На Рис. 2 через S_α позначено площину можливих неефективних випадків дохідності.

Якщо через $\varphi(\alpha)$ позначити геометричну ймовірність потрапляння портфелю в зону неефективності в залежності від рівня α, то очевидно, що справедлива наступна формула:

$$\varphi(\alpha) = \frac{S_\alpha}{\left(r_2(\alpha) - r_1(\alpha)\right)\left(r_2^*(\alpha) - r_1^*(\alpha)\right)}. \quad \textbf{(3)}$$

Нехай далі у викладеннях c – деяка константа.

А функцію ризику можна визначити як математичне сподівання потрапляння портфелю в зону неефективності по всім рівням α (енергетичним рівням), за формулою **(4)**:

$$\beta = \int_0^1 p(\alpha)\varphi(\alpha)d\alpha. \quad \textbf{(4)}$$

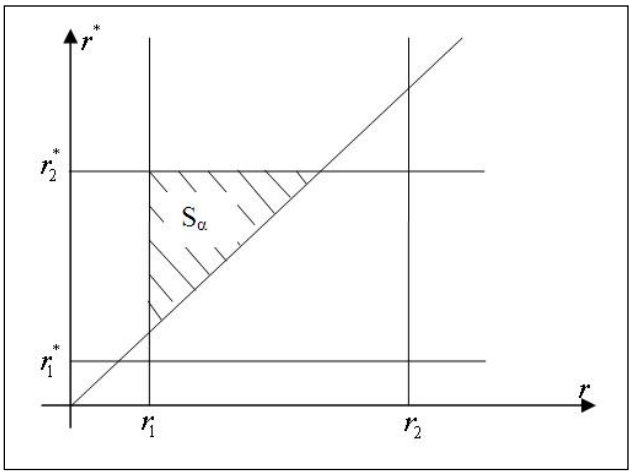


Рис. 2. Зображення зони неефективності портфеля.

Для того, щоб функція ризику змінювалася в діапазоні від 0 до 1 необхідне введення ще умови нормування **(5)**:

$$\int_0^1 p(\alpha)d\alpha = 1 . \quad \textbf{(5)}$$

$p(\alpha)$ – щільність ймовірності потрапляння портфелю на енергетичний рівень α .

В роботі [8] приведено, а в роботах [1-7] досліджено випадок, коли $p(\alpha) = const$. У цьому випадку формула **(4)** набуває вигляду:

$$\beta = \int_0^1 p(\alpha)\varphi(\alpha)d\alpha = \int_0^1 \frac{c}{\int_0^1 cdt}\varphi(\alpha)d\alpha = \int_0^1 \varphi(\alpha)d\alpha . \quad \textbf{(6)}$$

Можна запропонувати ще дві функції ризику.

Перша – коли щільність ймовірності потрапляння на енергетичний рівень змінюється пропорційно величині рівня. Це ситуація природна, так як для більшої волатильності дохідності портфеля потрібна більша енергія ззовні, що робить ризик потрапляння на нульовий рівень (рівень, що вимагає найбільше енергії) – нульовим.

Така функція ризику визначається формулою:

$$\beta = \int_0^1 p(\alpha)\varphi(\alpha)d\alpha = \int_0^1 \frac{\alpha}{\int_0^1 tdt}\varphi(\alpha)d\alpha = \frac{\int_0^1 \alpha\varphi(\alpha)d\alpha}{\int_0^1 tdt} = 2\int_0^1 \alpha\varphi(\alpha)d\alpha . \quad \textbf{(7)}$$

Також в роботі пропонується підхід, в якому мінімізується робота з чіткими значеннями функції ризику. Суть його полягає у тому, що для випадків, коли $\varphi(\alpha) = 0$ або $\varphi(\alpha) = 1$ щільність ймовірності дорівнює нулю – $p(\alpha) = 0$, тому ці випадки відкидаються як непродуктивні. В цьому випадку, якщо за α_1 позначити рівень, коли $\varphi(\alpha) \neq 0$ чи $\varphi(\alpha) \neq 1$, то формула **(4)** набуває вигляду:

$$\beta = \int_0^{\alpha_1} p(\alpha)\varphi(\alpha)d\alpha = \int_0^{\alpha_1} \frac{c}{\int_0^{\alpha_1} cdt}\varphi(\alpha)d\alpha = \frac{c\int_0^{\alpha_1}\varphi(\alpha)d\alpha}{\int_0^{\alpha_1} cdt} = \frac{\int_0^{\alpha_1}\varphi(\alpha)d\alpha}{\alpha_1} . \quad \textbf{(8)}$$

Твердження 1. Функція **(4)** є опуклою функцією.

Доведення. Нижче приводиться доведення твердження для часткового випадку, коли функція ризику описується рівномірним розподілом ймовірностей «енергетичних рівнів» та має вигляд **(6)**. Окрім того, розглядається випадок, коли критеріальне значення дохідності портфелю – чітке та дорівнює – r^* . Стосовно співвідношення критеріального значення та дохідності портфелю в цілому виконується наступна система нерівностей: $r_{\min} \le r^* \le \bar{r}$.

Доведення для інших випадків аналогічне та опущене у зв'язку з обмеженнями на об'єм статті.

Отже, у даному випадку функція ризику портфелю описується формулою [1-6]:

$$\beta\left(\vec{x}\right)=\frac{1}{\sum_{i=1}^{N}x_i r_{i\max}-\sum_{i=1}^{N}x_i r_{i\min}}\left(\left(r^*-\sum_{i=1}^{N}x_i r_{i\min}\right)+\left(\sum_{i=1}^{N}x_i\bar{r}_i-r^*\right)\ln\left(\frac{\sum_{i=1}^{N}x_i\bar{r}_i-r^*}{\sum_{i=1}^{N}x_i\bar{r}_i-\sum_{i=1}^{N}x_i r_{i\min}}\right)\right). \textbf{(9)}$$

Для спрощення викладення доведення твердження необхідно ввести наступні позначення:

$$A(x)=r^*-\sum_{i=1}^{N}x_i r_{i\min}\ , \textbf{(10)}$$

$$B(x)=\sum_{i=1}^{N}x_i\bar{r}_i-r^*, \textbf{(11)}$$

$$C(x)=\sum_{i=1}^{N}x_i\bar{r}_i-\sum_{i=1}^{N}x_i r_{i\min}\ , \textbf{(12)}$$

$$D(x)=\frac{1}{\sum_{i=1}^{N}x_i r_{i\max}-\sum_{i=1}^{N}x_i r_{i\min}}, \textbf{(13)}$$

$$\varphi(x)=A(x)+B(x)\ln\tfrac{B(x)}{C(x)}. \textbf{(14)}$$

Для доведення твердження про опуклість функції $\beta(x)$, спочатку необхідно довести опуклість функцій $\varphi(x)$ та $D(x)$. Окрім того, необхідно довести їх невід'ємність.

Спочатку розглянемо функцію $D(x)$.

Використовуючи факт того, що $r_{i\max}\geq r_{i\min}$, знайдемо та оцінимо наступні часткові похідні:

$$\frac{\partial D(x)}{\partial x_i}=-\frac{r_{i\max}-r_{i\min}}{\left(\sum_{i=1}^{N}x_i\left(r_{i\max}-r_{i\min}\right)\right)^2}<0\ , \textbf{(15)}$$

$$\frac{\partial^2 D(x)}{\partial x_i^2}=\frac{2\left(r_{i\max}-r_{i\min}\right)^2}{\left(\sum_{i=1}^{N}x_i\left(r_{i\max}-r_{i\min}\right)\right)^3}>0\ , \textbf{(16)}$$

$$\frac{\partial^2 D(x)}{\partial x_i\partial x_j}=\frac{2\left(r_{i\max}-r_{i\min}\right)\left(r_{j\max}-r_{j\min}\right)}{\left(\sum_{i=1}^{N}x_i\left(r_{i\max}-r_{i\min}\right)\right)^3}>0\ . \textbf{(17)}$$

Тепер опуклість функції $D(x)$ доводиться з наступної системи рівностей:

$$\frac{\partial^2 D(x)}{\partial x_i^2}\cdot\frac{\partial^2 D(x)}{\partial x_j^2}-\left(\frac{\partial^2 D(x)}{\partial x_i\partial x_j}\right)^2=$$

$$=\frac{4\left(r_{i\max}-r_{i\min}\right)^2\left(r_{j\max}-r_{j\min}\right)^2-4\left(r_{i\max}-r_{i\min}\right)^2\left(r_{j\max}-r_{j\min}\right)^2}{\left(\sum_{i=1}^{N}x_i\left(r_{i\max}-r_{i\min}\right)\right)^6}=0 \quad . \textbf{(18)}$$

Таким чином, функція $D(x)$ є опуклою.

Використовуючи цю ж процедуру доводиться опуклість функції $\varphi(x)$.

Для доведення цього факту необхідно ввести допоміжні позначення:

$$C_i'=\frac{\partial C(x)}{\partial x_i}=\bar{r}_i-r_{i\min}\ , \textbf{(19)}$$

$$B_i'(x)=\frac{\partial B(x)}{\partial x_i}=\bar{r}_i. \textbf{(20)}$$

Тоді:

$$\begin{aligned}\frac{\partial}{\partial x_i}\left[B(x)\ln\frac{B(x)}{C(x)}\right]&=B_i'(x)\ln\frac{B(x)}{C(x)}+\\&+B(x)\frac{C(x)}{B(x)}\frac{B_i'(x)C(x)-C_i'(x)B(x)}{C^2(x)}=\\&=B_i'(x)\ln\frac{B(x)}{C(x)}+\frac{B_i'(x)C(x)-C_i'(x)B(x)}{C(x)}=\\&=\bar{r}_i\ln\frac{B(x)}{C(x)}+\bar{r}_i-\left(\bar{r}_i-r_{i\min}\right)\frac{B(x)}{C(x)}.\end{aligned} \quad \textbf{(21)}$$

Далі береться друга часткова похідна та після перетворень вона набуває вигляду **(22)**:

$$\frac{\partial^2}{\partial x_i^2}\left[B(x)\ln\frac{B(x)}{C(x)}\right]=\frac{\left(\bar{r}_i\left(\sum_{i=1}^{N}x_i\left(\bar{r}_i-r_{\min}\right)-\left(\bar{r}_i-r_{i\min}\right)\left(\sum_{i=1}^{N}x_i\bar{r}_i-r^*\right)\right)\right)^2}{\left(\sum_{i=1}^{N}x_i\bar{r}_i-r^*\right)\left(\sum_{i=1}^{N}x_i\left(\bar{r}_i-r_{i\min}\right)\right)^2}\geq 0\ . \textbf{(22)}$$

Виходячи з тих фактів, що $\bar{r}_i>\left(\bar{r}_i-r_{i\min}\right)$ та $\sum_{i=1}^{N}x_i\left(\bar{r}_i-r_{i\min}\right)>\sum_{i=1}^{N}x_i\bar{r}_i-r^*$, можна прийти до висновку, що вираз **(22)** строго більше за 0.

$$\frac{\partial^2}{\partial x_i^2}\left[B(x)\ln\frac{B(x)}{C(x)}\right]>0\ ,$$

і, тому:

Отже,

$$\frac{\partial^2}{\partial x_i^2}\left[A(x)+B(x)\ln\frac{B(x)}{C(x)}\right]>0\ . \textbf{(23)}$$

Наступним кроком є розрахунок змішаних часткових похідних. Результат розрахунку – формула **(24)**:

$$\frac{\partial^2}{\partial x_i \partial x_j}\left[B(x)\ln\frac{B(x)}{C(x)}\right] = \frac{\overline{r_i}\,\overline{r_j}\left(\sum_{i=1}^{N} x_i\left(\overline{r_i}-r_{i\min}\right)^2\right)}{\left(\sum_{i=1}^{N} x_i\overline{r_i}-r^*\right)\left(\sum_{i=1}^{N} x_i\left(\overline{r_i}-r_{i\min}\right)^2\right)} - \frac{\left(2\overline{r_i}\,\overline{r_j}-\overline{r_i}r_{j\min}-\overline{r_j}r_{i\min}\right)\left(\sum_{i=1}^{N} x_i\left(\overline{r_i}-r_{i\min}\right)\right)\left(\sum_{i=1}^{N} x_i\overline{r_i}-r^*\right)}{\left(\sum_{i=1}^{N} x_i\overline{r_i}-r^*\right)\left(\sum_{i=1}^{N} x_i\left(\overline{r_i}-r_{i\min}\right)^2\right)} + \frac{\left(\overline{r_i}-r_{i\min}\right)\left(\overline{r_j}-r_{j\min}\right)\left(\sum_{i=1}^{N} x_i\overline{r_i}-r^*\right)^2}{\left(\sum_{i=1}^{N} x_i\overline{r_i}-r^*\right)\left(\sum_{i=1}^{N} x_i\left(\overline{r_i}-r_{i\min}\right)^2\right)}. \quad \textbf{(24)}$$

Раніше було визначено, що $r_{\min}=\sum_{i=1}^{N} x_i r_{i\min}$, $\overline{r}=\sum_{i=1}^{N} x_i\overline{r_i}$ та $r_{\max}=\sum_{i=1}^{N} x_i r_{i\max}$. Окрім того, для спрощення подальшого викладення необхідно ввести позначення:

$$E(x)=\left(\left(\sum_{i=1}^{N} x_i\overline{r_i}-r^*\right)\left(\sum_{i=1}^{N} x_i\left(\overline{r_i}-r_{i\min}\right)\right)\right)^2, \quad \textbf{(25)}$$

$$\frac{\partial^2}{\partial x_i \partial x_j}\left[B(x)\ln\frac{B(x)}{C(x)}\right]=\frac{H(x)}{E(x)}, \quad \textbf{(26)}$$

$$F(x)=\overline{r_i}\left(\overline{r}-r_{\min}\right)-\left(\overline{r_i}-r_{\min}\right)\left(\overline{r}-r^*\right), \quad \textbf{(27)}$$

$$G(x)=\overline{r_j}\left(\overline{r}-r_{\min}\right)-\left(\overline{r_j}-r_{j\min}\right)\left(\overline{r}-r^*\right). \quad \textbf{(28)}$$

Таким чином,

$$\frac{\partial^2}{\partial x_i^2}\left[B(x)\ln\frac{B(x)}{C(x)}\right]\cdot\frac{\partial^2}{\partial x_j^2}\left[B(x)\ln\frac{B(x)}{C(x)}\right]-\left(\frac{\partial^2}{\partial x_i\partial x_j}\left[B(x)\ln\frac{B(x)}{C(x)}\right]\right)^2 = \frac{F^2G^2-H^2}{E^2}=\frac{(FG-H)(FG+H)}{E^2}. \quad \textbf{(29)}$$

Отже, невід'ємність **(29)** полягає у $FG-H>0$.

$$\begin{aligned} FG-H &= \left(\overline{r_i}\left(\overline{r}-r_{\min}\right)-\left(\overline{r_i}-r_{i\min}\right)\left(\overline{r}-r^*\right)\right)\times \\ &\times\left(\overline{r_j}\left(\overline{r}-r_{\min}\right)-\left(\overline{r_j}-r_{j\min}\right)-\left(\overline{r_j}-r_{j\min}\right)\left(\overline{r}-r^*\right)\right)- \\ &-\overline{r_i}\,\overline{r_j}\left(\overline{r}-r_{\min}\right)^2+\left(2\overline{r_i}\,\overline{r_j}-\overline{r_i}r_{j\min}-\overline{r_j}r_{i\min}\right)\left(\overline{r}-r_{\min}\right)\left(\overline{r}-r^*\right)- \\ &-\left(\overline{r_i}-r_{i\min}\right)\left(\overline{r_j}-r_{j\min}\right)\left(\overline{r}-r^*\right)^2= \\ &=\left(\overline{r}-r_{\min}\right)\left(\overline{r}-r^*\right)\left(-\overline{r_j}\left(\overline{r_i}-r_{i\min}\right)-\overline{r_i}\left(\overline{r_j}-r_{j\min}\right)+2\overline{r_i}\,\overline{r_j}-\overline{r_i}r_{j\min}-\overline{r_j}r_{i\min}\right)=0 \end{aligned} \quad \textbf{(30)}$$

Тобто функція $B(x)\ln\frac{B(x)}{C(x)}$ є опуклою, а, отже опуклою є функція **(14)**.

Тепер, для остаточного доведення твердження залишилося показати, що опуклою є функція $\beta(x)=D(x)\cdot\varphi(x)$.

Варто нагадати, що, як було показано раніше, $\varphi(x)$ та $D(x)$ є додатними функція та, окрім того, $D(x)$ є монотонно спадною функцією, оскільки, $D_i'(x)<0$.

Необхідно довести ще факт того, що $\frac{\partial\varphi(x)}{\partial x_i}=\varphi_i'(x)<0$.

$$\frac{\partial\varphi(x)}{\partial x_i} = \frac{\partial}{\partial x_i}\left(A(x) + B(x)\ln\frac{B(x)}{C(x)}\right) =$$

$$= A'(x) + B'(x) + B(x)\frac{C(x)}{B(x)}\frac{B'(x)C(x) - C'(x)B(x)}{C^2(x)} = \textbf{(26)}$$

$$= A'(x) + B'(x)\ln\frac{B(x)}{C(x)} + B'(x) - \frac{B(x)C'(x)}{C(x)}.$$

При підстановці значень $A'(x)$ та $B'(x)$, вираз **(26)** набуває вигляду:

$$\frac{\partial\varphi(x)}{\partial x_i} = -r_{i\min} + \bar{r}_i\ln\frac{B(x)}{C(x)} + \bar{r}_i - \left(\bar{r}_i - r_{i\min}\right)\frac{B(x)}{C(x)} =$$

$$= \bar{r}_i\left(1 + \ln\frac{B(x)}{C(x)} - r_{i\min} - \left(\bar{r}_i - r_{i\min}\right)\frac{B(x)}{C(x)}\right) \textbf{. (27)}$$

Оскільки $\frac{B(x)}{C(x)} < 1$, то $-r_{i\min} + r_{i\min}\frac{B(x)}{C(x)} < 0$.

Звідки, після спрощення **(27)**, можна отримати **(28)**:

$$\frac{\partial\varphi(x)}{\partial x_i} = \bar{r}_i\left(1 + \ln\frac{B(x)}{C(x)} - \frac{B(x)}{C(x)}\right). \textbf{(28)}$$

Якщо повернутися до початкових змінних,то

$$1 + \ln\frac{B(x)}{C(x)} - \frac{B(x)}{C(x)} = 1 + \ln\frac{\bar{r} - r^*}{\bar{r} - r_{\min}} - \frac{\bar{r} - r^*}{\bar{r} - r_{\min}}. \textbf{(29)}$$

Тут слід згадати, що $r^* > r_{\min} = \sum_{i=1}^{N} x_i r_{i\min}$ та $\bar{r} > r^*$. Необхідно показати, що вираз **(29)** є меншим за 0. Якщо ввести позначення $\bar{r} - r^* = a$ та $y = r^* - r_{\min}$, то отримаємо **(30)**:

$$\bar{r} - r_{\min} = \bar{r} - r^* + \left(r^* - r_{\min}\right) = a + y \textbf{. (30)}$$

Підставляючи **(30)** в **(29)** можна отримати, що:

$$1 + \ln\frac{\bar{r} - r^*}{\bar{r} - r_{\min}} - \frac{\bar{r} - r^*}{\bar{r} - r_{\min}} = 1 + \ln\frac{a}{a + y} - \frac{a}{a + y}. \textbf{(31)}$$

Необхідно показати, що Очевидно, що $\Delta(0) = 1 + \ln\frac{a}{a + y} - \frac{a}{a + y} = 0$ при $y = 0$. Окрім того $\Delta(y)$ монотонно спадна, так як:

$$\Delta(y) = 1 + \ln\frac{a}{a + y} - \frac{a}{a + y} < 0, \forall y > 0.$$

$$\Delta'(y) = -\frac{1}{a + y} + \frac{a}{(a + y)^2} = -\frac{y}{(a + y)^2} < 0, \forall y > 0 \textbf{. (32)}$$

Таким чином, $\Delta(y) < 0, \forall y > 0$, і, наприкінці, $\frac{\partial\varphi(x)}{\partial x_i} < 0$.

Далі проводиться обчислення перших похідних та проаналізуємо їх на підставі доведених раніше фактів про знаки функцій $\varphi(x)$ та $D(x)$ і їх похідних:

$$\frac{\partial(\varphi(x)D(x))}{\partial x_i} = \varphi'(x)D(x) + D'(x)\varphi(x) < 0 \textbf{. (33)}$$

Розраховуються другі похідні:

$$\frac{\partial^2(\varphi(x)D(x))}{\partial x_i^2} = \varphi''_{ii}(x)D(x) + D'_i(x)\varphi'_i(x) + D''_{ii}(x)\varphi(x) + \quad \textbf{(34)}$$
$$+ D'_i(x)\varphi'_i(x) = \varphi''_{ii}(x)D(x) + 2D'_i(x)\varphi'_i(x) + D''_{ii}(x)\varphi(x)$$

Але, як було показано раніше, $D''_{ii}(x) > 0, \varphi''_{ii}(x) > 0, D'_i(x) < 0, \varphi'_i(x) < 0$. Таким чином,

$$\frac{\partial^2(\varphi(x)D(x))}{\partial x_i^2} > 0 \textbf{.} \quad \textbf{(35)}$$

А умова (35) є достатньою умовою того, що функція $\beta(x) = \varphi(x)D(x)$ є опуклою.

Твердження доведено.

Зауваження 1. По-перше, в твердженні доведено, що функція є опуклою, а не строго опуклою. Отже вона може мати більш ніж один екстремум. Окрім того, за результатами роботи [7] задача **(1)** у загальному випадку має континуум розв'язків, навіть у випадку, коли функція має один екстремум, а отже, для отримання одного розв'язку цієї задачі необхідно виконувати дооптимізацію по критерію ризику.

2. Аналіз залежності ризик-дохідність для портфелю з двох фінансових інструментів

Для задачі оптимізації у вигляді **(1)**, як було показано в роботі [7], дуже важливим є випадок $n = 2$. А для даного випадку критичним є вигляд залежності дохідність-ризик. Тобто, якщо по осі абсцис відкласти значення дохідності портфелю, а по осі ординат – ризику, то на даній площині можливо зобразити залежність $\beta(\bar{r})$. В роботі [7] були приведені випадки, коли ця залежність була спадаючою, зростаючою та, просто, мала опуклий вигляд. У роботах [1-8] ще зазначається опуклість функції ризику. А тут постає цікава задача: якщо з взяти два розподіли активів в портфелі – $x_1 = (x_{11} \quad x_{12} \quad \dots \quad x_{1n}), \sum_{i=1}^{n} x_{1i} = 1$ та $x_2 = (x_{21} \quad x_{22} \quad \dots \quad x_{2n}), \sum_{i=1}^{n} x_{2i} = 1$ й побудувати на них симплекс $x_3(\alpha) = \alpha x_1 + (1-\alpha)x_2$, $0 \le \alpha \le 1$, то який критерій можна використати, щоб, не будуючи сітки значень залежності $\beta(\bar{r}(x_3(\alpha)))$, визначити її вигляд по двом точкам – x_1 та x_2.

І тут існує два підходи до розв'язання цієї задачі: взяти значення проекції похідної функції ризику на симплекс, побудований на точках, у цих двох точках по α та подивитися на її знак, чи порахувати узагальнену дивергенцію **(37)** функції ризику по змінним $r_{\min}$, $\bar{r}$ та $r_{\max}$ (тобто, по мінімальному, найбільш очікуваному та максимальному значенню дохідності портфеля) у точках x_1 та x_2 та подивитися на її знаки. Очевидно, що у обчислювальному сенсі останній випадок є більш ефективним.

Нехай точкам x_1 та x_2 відповідають трикутні числа дохідності портфелю $(r_{\min 1}, \bar{r}_1, r_{\max 1})$ та $(r_{\min 2}, \bar{r}_2, r_{\max 2})$. Нехай, для визначеності, $\bar{r}_1 \le \bar{r}_2$.

Тоді узагальнена дивергенція визначається наступною системою формул **(36)-(37)**.

$$\beta = \beta\left(r^*_{\min}, \bar{r}, r^*_{\max}, r_{\min}(x), \bar{r}(x), r_{\max}(x)\right). \quad \textbf{(36)}$$

$$div_{x_1x_2}\beta(x) = sign\,(r_{\min 2} - r_{\min 1})\frac{\partial}{\partial r_{\min}}\beta(x) + \frac{\partial}{\partial \bar{r}}\beta(x) + sign\,(r_{\max 2} - r_{\max 1})\frac{\partial}{\partial r_{\max}}\beta(x) \textbf{.} \quad \textbf{(37)}$$

Тепер очевидно, що якщо $div_{x_1x_2}\beta(x_1) > 0$ та $div_{x_1x_2}\beta(x_2) > 0$ – залежність $\beta(\bar{r}(x_3(\alpha)))$ має зростаючий характер, коли $div_{x_1x_2}\beta(x_1) < 0$ та $div_{x_1x_2}\beta(x_2) < 0$ – залежність $\beta(\bar{r}(x_3(\alpha)))$ має спадаючий характер. У випадку, коли $div_{x_1x_2}\beta(x_1) > 0$ та $div_{x_1x_2}\beta(x_2) < 0$ – залежність має увігнутий вниз характер. Далі в роботі узагальнена дивергенція буде називатися просто – дивергенція.

3. Метод оптимізації портфелю

На підставі дивергенції **(37)** та опуклості функції ризику стає можливою побудова оптимального розподілу на симплексі, утвореному на двох розподілах x_1 та x_2. Процедура побудови даного розподілу далі в роботі буде називатися процедурою F. Її алгоритм наводиться нижче.

По перше, якщо $\bar{r}_1 > \bar{r}_2$, виконується перепозначення x_1 в x_2, а x_2 в x_1 (дане перейменування не впливає на сам метод оптимізації, а в роботі лише має методичне значення).

Алгоритм процедури F.

1. Якщо $\beta(x_1) \le \beta_1$ та $\beta(x_2) \le \beta_1$, то, в силу опуклості функцій ризику та доходності, оптимальним рішенням задачі (1) на симплексі

$\{x : \gamma x_1 + (1-\gamma)x_2, 0 \le \gamma \le 1\}$ є або точка x_2 у випадку, коли $\bar{r}_1 < \bar{r}_2$, або увесь симплекс, коли $\bar{r}_1 = \bar{r}_2$.

2. Якщо $\beta(x_1) > \beta_1$, а $\beta(x_2) \le \beta_1$, то очевидно, що оптимальним рішенням буде x_2.

3. Якщо $\beta(x_1) \le \beta_1$, а $\beta(x_2) > \beta_1$, то будується точка $x_C = \frac{1}{2}x_1 + \frac{1}{2}x_2$. Утворюється два нових симплекса $C_1 = \{x : \gamma x_1 + (1-\gamma)x_C, 0 \le \gamma \le 1\}$ та $C_2 = \{x : \gamma x_C + (1-\gamma)x_2, 0 \le \gamma \le 1\}$, на кожному з яких рекурентно запускається процедура F при заданій точності, а з двох оптимальних розподілів обирається одне з найбільшою дохідністю.

4. Якщо $\beta(x_1) \le \beta_1$ та $\beta(x_2) > \beta_1$, то:

4.1. У випадку $div_{x_1x_2}\beta(x_1) < 0$ та $div_{x_1x_2}\beta(x_2) > 0$ виконується процедура оптимізації, описана у пункті 3.

4.2. У протилежному до пункту 4.1 випадку задача не має рішень.

На застосуванні процедури F, ґрунтуючись на властивості опуклості функцій ризику та дохідності та, враховуючи *Зауваження 1*, стає можливим побудувати метод оптимізації нечіткого фондового портфелю, що може містити n активів.

Метод оптимізації нечіткого фондового портфелю.

Допустимою множеною рішень (якщо відкинути умову на обмеження по ризику) задачі **(1)** є симплекс $C = \left\{x : x = \sum_{i=1}^{n}\gamma_i x_i, \sum_{i=1}^{n}\gamma_i = 1\right\}$, де $x_1 = \underbrace{(1,0,0,...,0)}_{n}$, $x_2 = \underbrace{(0,1,0,...,0)}_{n}$, …, $x_n = \underbrace{(0,0,0,...,1)}_{n}$, а n – максимальна кількість активів в портфелі. Саме ці позначення будуть використані при описанні алгоритму методу.

Алгоритм методу.

1. Нехай $D = \{x_i, i = 1,..n\}$. Множина D розбивається на дві підмножини: $A = \{x_i : \beta(x_i) > \beta_1\}$ та $B = \{x_i : \beta(x_i) \le \beta_1\}$.

2. Для точок множини A попарно застосовується процедура F; оптимальні рішення зберігаються в множині S.

3. На точках множини B попарно шукаються розподіли з мінімальним ризиком:

- якщо для x_i та x_j значення узагальненої дивергенції – $div_{x_ix_j}$ – різних знаків, то застосовується метод дихотомічного пошуку для знаходження розподілу, коли дивергенція дорівнює нулю. Отримана точка мінімуму додається до множини B;
- в протилежному випадку – мінімальне значення є на одній з точок і ніякі додаткові дії не виконуються.

4. В силу строгої опуклості функції ризику глобальний мінімум ризику на множині B знаходиться наступним чином: мінімальні попарні значення функції ризику зберігаються в множині K та до множини K застосовуються ті самі дії, що і до множини B на кроці 3 ітеративно до тих пір, поки не буде досягнуто множини точок, що не змінюється, або межі точності. Оптимальне рішення зберігається в множині B, якщо воно там відсутнє.

5. Обираються по одній точці з множин A та B та до них попарно застосовується процедура F; оптимальні рішення зберігаються в множині S.

6. Простим лінійним перебором серед точок множини S вибираються точки з найбільшими значеннями очікуваних дохідностей портфелю. Ці точки утворюють множину M.

7. Якщо множина M складається з єдиної точки – знайдено оптимальне рішення. Інакше – на крок 8.

8. Попарно на точках $x_i, x_j \in M$ будуються симплекси $C_{ij} = \{x : x = \gamma x_i + (1-\gamma)x_j, 0 \le \gamma \le 1\}$ та на даній множині виконується наступна задача оптимізації 8.1-8.4:

8.1. Якщо $div_{x_ix_j}\beta(x_i) = 0$ та $div_{x_ix_j}\beta(x_j) = 0$, то оптимальним рішенням буде увесь симплекс C_{ij} і точки x_i та x_j залишаються в множині M.

8.2. Якщо $div_{x_ix_j}\beta(x_i) < 0$ та $div_{x_ix_j}\beta(x_j) \le 0$, то точка x_i викидається з множини M після закінчення кроку 6 алгоритму.

8.3. Якщо $div_{x_ix_j}\beta(x_i) \ge 0$ та $div_{x_ix_j}\beta(x_j) > 0$, то точка x_j викидається з множини M після закінчення кроку 6 алгоритму.

8.4. Якщо $div_{x_ix_j}\beta(x_i) < 0$ та $div_{x_ix_j}\beta(x_j) > 0$, то, використовуючи метод дихотомічного пошуку та рекурентно викликаючи умови 8.1-8.4. при заданому обмеженні на точність шукається $x_{ij}^0 : div_{x_ix_j}\beta(x_{ij}^0) = 0$. Дана точка зберігається в множині M, а точки x_i та x_j викидаються з множини M після закінчення кроку 6 алгоритму.

9. Якщо M складається з одного елемента, чи діаметр множини менше за межу точності,

чи для всіх точок з множини рівні ризику рівні, то знайдено оптимальну множину розв'язків – симплекс, натягнутий на точки множини M. В іншому випадку – крок 7.

Зауваження 2. Якщо $\forall x_i, i = 1,.., n : \beta(x_i) \le \beta_1$, то $S = B$ і алгоритм з кроку 1 відразу переходить на крок 4.

Твердження 2. Метод збігається за кінцеву кількість кроків.

Доведення. На етапі 2 буде застосовано $\frac{card\ (A)\cdot(card\ (A)-1)}{2}$ процедур F, котра збігається за кінцеву кількість кроків в силу збіжності методу дихотомічного пошуку за кінцеву кількість кроків при заданій точності.

В гіршому випадку на етапі 3 буде застосовано $\frac{card\ (B)\cdot(card\ (B)-1)}{2}$ процедур дихотомічного пошуку.

Об'єм множини K на кожному кроці етапу 4 буде зменшуватись, а отже етап 4 в гіршому випадку зупиниться при досягненні заданої точності.

На етапі 5 буде застосовано $card\ (A)\cdot card\ (B)$ процедур F.

На етапі 6 буде виконано $\frac{card\ (S)\cdot(card\ (S)-1)}{2}$ лінійних порівнянь.

Етапі 7 у гіршому випадку застосовує $card\ (M)\cdot(card\ (M)-1)$ процедур дихотомічного пошуку при заданій точності. А діаметр множини M, що оцінюється на кроці 8, постійно зменшується з кожною ітерацією циклу 8-9, а отже, при заданій точності, кількість ітерацій цього циклу теж обмежена.

Таким чином, метод збігається за скінченну кількість кроків.

Твердження 3. В ході роботи алгоритму методу оптимальне рішення не відкидається.

Доведення.

В силу опуклості функцій ризику та дохідності та у відповідності до етапів методу оптимізації, при відкиданні точки початкового симплексу, оптимальне рішення може бути представлено у вигляді лінійної комбінації точок з певної підмножини множини M, а отже оптимальне рішення в ході роботи алгоритму втрачено не буде.

Висновки

В роботі було розглянуто функцію ризику нечіткого інвестиційного портфелю як найбільш очікуваний випадок потрапляння активу у зону неефективності. На підставі цього було досліджено відому та введено дві нові функції ризику нечіткого інвестиційного портфелю. Також було введено новий простий у обчислювальному сенсі критерій визначення по двох «крайніх» точках виду залежності ризик-дохідність для двох інструментів у портфелі. Слід зауважити, що даний критерій ґрунтується саме на властивості опуклості залежності ризик-дохідність. Запропоновано алгоритм методу оптимізації нечіткого фондового портфелю, у якому використовується даний критерій та припускається, що множина розв'язків задачі пошуку оптимального розподілу у портфелі може містити більше одного елемента.

Список літератури

1. Зайченко Ю.П., Малихех Есфандиярфард. Анализ инвестиционного портфеля для различных видов функции принадлежности // Системні дослідження та інформаційні технології – №2, – Київ,- 2008. – С.59-76
2. Зайченко Ю.П., Малихех Есфандиярфард, Заика А.И. Анализ инвестиционного портфеля на основе прогнозирования курсов акций // Вісник Національного технічного університету України «КПІ». Інформатика, управління та обчислювальна техніка – №47,- Київ,- 2007. – С. 168-179
3. Зайченко Ю.П., Малихех Есфандиярфард, Ови Нафас Агаи Аг Гамиш. Анализ модели оптимизации нечёткого портфеля // Вісник національного технічного університету України «КПІ». Інформатика, управління та обчислювальна техніка – №51, – Київ,- 2010.- С. 197-203
4. Зайченко Ю.П., Малихех Есфандиярфард. Исследование задачи оптимизации инвестиционного портфеля в нечётких условиях // Information science & Computing, International Book Series – #10, – 2009. – Р. 74-82
5. Зайченко Ю.П., Ови Нафас Агаи Аг Гамиш. Исследование двойственной задачи оптимизации инвестиционного портфеля в нечётких условиях // Системні дослідження та інформаційні технології – №3, – Київ,- 2011. – С.63-76

6. Малихех Есфандиярфард, Юрий Зайченко, Ови Нафас Агаи Аг Гамиш. Исследование многокритериальной задачи оптимизации инвестиционного портфеля в нечётких условиях // Applicable information models, International Book Series – #22, – 2011. – P. 83-89
7. Мурга Н.А. Нечёткий фондовый портфель. Исследование и оптимизация/ Н.А. Мурга // Системні дослідження та інформаційні технології. – 2010. – №3. – С. 60-71.
8. Недосекин А.О. Нечётко множественный анализ риска фондовых инвестиций. – СПб.: 2002. – 181 с.

УДК 004.942

КАЛИНОВСКИЙ Я.А.,
БОЯРИНОВА Ю.Е.,
ГОРОДЬКО Н.А.

ПОСТРОЕНИЕ МАТРИЧНЫХ ПРЕДСТАВЛЕНИЙ ИЗОМОРФНЫХ ГИПЕРКОМПЛЕКСНЫХ ЧИСЛОВЫХ СИСТЕМ

В статье рассматривается метод построения матричных представлений гиперкомплексных числовых систем. Метод основан на использовании изоморфных гиперкомплексных числовых систем диагонального вида. При этом отпадает необходимость в решении высокоразмерных систем квадратичных уравнений.

In this article a method for constructing matrix representations of hypercomplex number systems is considered. The method is based on the use of hypercomplex number systems by isomorphic to diagonal form. This eliminates the need for solving quadratic equations of high dimension systems.

Вступление

Все более широкое применение гиперкомплексных числовых систем (ГЧС) требует разработки методов повышения эффективности как вычислительных процедур при моделировании с использованием гиперкомплексных чисел, так и синтеза самих математических моделей. С этой точки зрения широкие возможности предоставляет использование принципа изоморфизма гиперкомплексных числовых систем. Он позволяет получить достаточно весомые результаты в обоих выше обозначенных направлениях. Так, например, при использовании ГЧС для проектирования рекурсивных цифровых фильтров [1] использование изоморфизма позволяет снизить объем вычислений, необходимый для функционирования математической модели фильтра, что позволяет повысить тактовую частоту фильтра. В данной работе принцип изоморфизма применяется для упрощения построения матричных представлений в ГЧС, что повышает эффективность построения математических моделей.

Формы представления гиперкомплексных числовых систем

В теории гиперкомплексных числовых систем (ГЧС) рассматриваются две формы представления гиперкомплексных числовых систем: натуральная и матричная.

В первой из них гиперкомплексное число имеет вид:

$$A = \sum_{i=1}^{n} a_i e_i \qquad (1)$$

где $e = \{e_i\}_{i=1,\dots,n}$ – базис системы. При этом должна быть задана таблица умножения базисных элементов в форме:

$$e_i e_j = \sum_{k=1}^{n} \gamma_{ij}^k e_k ; i, j = 1,\dots, n , \qquad (2)$$

где γ_{ij}^k – структурные константы ГЧС ($\gamma_{ij}^k \in R$). При умножении двух гиперкомплексных чисел вида (1) необходимо произвести n^2 символьных операций обращения к таблице умножения (2).

При матричном представлении каждому базисному элементу e_i соответствует матрица $M(e_i)$ размерами $n \times n$, элементы которой принадлежат R, а гиперкомплексные числа (1) и произведения базисных элементов (2) также являются матрицами $n \times n$. И, таким образом, все действия с гиперкомплексными числами сводятся к действиям над матрицами.

Применение матричного представления ГЧС может быть целесообразным при больших размерностях систем и, особенно, в случае неканонических ГЧС.

Матричные представления ГЧС

Гиперкомплексная числовая система есть элемент конечномерной алгебры с фиксированным базисом [1,3], которая является кольцом, аддитивная группа которого – векторное пространство [2]. Поэтому на гиперкомплексные числовые системы распространяются все свойства колец. В частности, существует теорема о вложенности колец [2,6]: всякое кольцо изоморфно вкладывается в полное кольцо матриц. На этой теореме основано матричное представле-

ние гиперкомплексных числовых систем. Широко известны представления системы комплексных чисел C и кватернионов H.

Так для системы комплексных чисел C с базисом (e_1, e_2) матричные представления базисных элементов имеют вид:

$$M(e_1) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix};\ M(e_2) = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad (3)$$

Действительно:

$$M(e_2^2) = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} = -M(e_1).$$

Тогда и комплексное число $A = a_1 e_1 + a_2 e_2$ можно представить в матричном виде:

$$M(A) = a_1 \cdot \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + a_2 \cdot \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ -a_2 & a_1 \end{pmatrix}.$$

Общий случай определения матричных представлений

Рассмотрим общий случай определения матричного представления коммутативной гиперкомплексной числовой системы размерности n. Учитывая то, что каждому базисному элементу e_i соответствует матричное представление E_i размерами $n \times n$, то всего необходимо определить n^3 элементов n таких матриц. В случая наличия в базисе единичного элемента неизвестных элементов будет несколько меньше - $n^2(n-1)$.

Для определения неизвестных элементов необходимо сформировать и решить систему матричных уравнений, соответствующих таблице умножения рассматриваемой гиперкомплексной числовой системы (2):

$$M(e_i)M(e_j) = \sum_{k=1}^{n} \gamma_{ij}^k M(e_k);\ i, j = 1,..., n \quad (4)$$

которые соответствуют таблице умножения (2). Для коммутативной гиперкомплексной числовой системы размерности n в общем случае таких матричных уравнений будет $\frac{n(n+1)}{2}$, а в случае наличия в базисе единичного элемента n матричных уравнений (4) превратятся в тождества, и система матричных уравнений будет состоять из $\frac{n(n-1)}{2}$ уравнений.

Система матричных уравнений (4) расщепляется в систему вещественных квадратичных уравнений, число которых в первом случае – $\frac{n^2(n+1)}{2}$, а во втором – $\frac{n^2(n-1)}{2}$.

Существование вещественных решений системы (4) вытекает из вышеупомянутой теоремы о вложенности кольца в полное кольцо матриц.

Как видно из проделанного выше анализа, полученные системы квадратичных уравнений очень громоздки даже для не очень больших размерностей и их решение вызывает заметные трудности [4]. Рассмотрим в качестве примера процесс поиска матричных представлений базисных элементов для системы комплексных чисел C с базисом (e_1, e_2). Так как единичный элемент в C является базисным элементом (e_1), то его матричное представление $M(e_1)$ есть единичная матрица второго порядка. Для определения матричного представления второго базисного элемента $M(e_2)$ составляем одно матричное уравнение вида (4):

$$M(e_2)M(e_2) = -M(e_1).$$

Если учесть:

$$M(e_2) = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix},$$

то предыдущее матричное уравнение превращается в систему 4-х квадратичных уравнений с 4-мя неизвестными:

$$\left\{ \begin{array}{l} x_{11}^2 + x_{12}x_{21} = -1 \\ x_{11}x_{12} + x_{12}x_{22} = 0 \\ x_{11}x_{21} + x_{21}x_{22} = 0 \\ x_{22}^2 + x_{12}x_{21} = -1 \end{array} \right\}.$$

Система имеет несчетное множество решений:

$$x_{11} \in R, x_{22} = -x_{11}, x_{12} \in R \setminus 0, x_{21} = -\frac{1 + x_{11}^2}{x_{12}}.$$

Выберем значения произвольных такими, чтобы $M(e_2)$ имело простой вид. При $x_{11} = x_{22} = 0, x_{12} = 1, x_{21} = -1$ представление $M(e_2)$ имеет вид (3). Таким образом, квадратичная система с повышением размерности ГЧС становится все более громоздкой. Поэтому работы по повышению эффективности решения таких систем и получению матричных представлений представляют заметный интерес.

Построение матричного представления системы по матричному представлению другой системы и линейному преобразованию изоморфизма между этими системами

В некоторых практических задачах, например при проектировании рекурсивных фильтров с гиперкомплексными коэффициентами [1,5,7,8], применяются пары изоморфных ГЧС, одна из которых имеет сильно заполненную таблицу умножения, а вторая – слабо заполненную и имеющую диагональный вид. Последнее означает, что на главной диагонали таблицы умножения ГЧС размерности $n \geq 1$ стоит m заполненных подтаблиц размерностью n_m, так что $\sum_{i=1}^{m} n_m = n$, а все остальные элементы таблицы – нули. Например:

	e_1	e_2	e_3	e_4
e_1	e_1	0	0	0
e_2	0	e_2	0	0
e_3	0	0	e_3	e_4
e_4	0	0	e_4	$-e_3$

(5)

Как видно, здесь: $n = 4$; $m = 3$; $n_1 = n_2 = 1$; $n_3 = 2$.

Следует отметить, что линейное преобразование изоморфизма для базисных элементов пары изоморфных ГЧС сохраняется и для матричного представления базисных элементов ввиду транзитивности отношения изоморфизма. То есть, если L – линейный оператор изоморфизма систем с базисами $e = \{e_i\}_{i=1,\ldots,n}$, $f = \{f_i\}_{i=1,\ldots,n}$, и $e = L(f)$, то и для матричных представлений сохраняется то же соотношение: $E = L(F)$.

Поэтому по матричному представлению одной из систем и линейному преобразованию изоморфизма между этими системами можно легко построить матричное представление для другой системы. Для этого нужно подставить в уравнения линейного преобразования изомофизма матричные представления базисних элементов первой системы и проделать все матричне операции. При этом отпадает необходимость решения системы квадратичных уравнений (4).

Матричные представления ГЧС диагонального типа

Если ГЧС имеет диагональный вид, то для неё легко построить матричное представление. Действительно, ГЧС Γ диагонального вида является прямой суммой некоторого количества ГЧС меньших размерностей Γ_i:

$$\Gamma = \bigoplus_{i=1}^{m} \Gamma_i .$$

Размерность n системы Γ есть сумма размерностей всех входящих в неё подсистем:

$$n = \dim\ \Gamma = \sum_{i=1}^{m} \dim\ \Gamma_i$$

Тогда базис системы Γ состоит из всех базисних элементов его подсистем Γ_i. Если известно матричное представление базисного элемента из Γ_i, то его матричное представление в Γ образуется обрамлением представления в Γ_i соответствующим количеством нулей до матрицы $n \times n$.

Рассмотрим в качестве примера систему $R \oplus C$ – прямую сумму систем вещественных и комплексных чисел с таблицей умножения

	E_1	E_2	E_3
E_1	E_1	0	0
E_2	0	E_2	E_3
E_3	0	E_3	$-E_2$

(6)

Матричное представление системы вещественных чисел $M(R)$ – единичная матрица размерами (1×1). Так как E_1 стоит в правом верхнем углу таблицы (8), то дополнив справа и снизу двумя рядами и строками нулей, получим матричное представление базисного элемента E_1 в $R \oplus C$:

$$M(E_1) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \qquad (7)$$

Используя матричные представления базисных элементов для системы комплексных чисел (3), получим:

$$M(E_2) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}; \quad M(E_3) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix} \qquad (8)$$

Примеры построения матричных представлений

Рассмотрим примеры построения матричных представлений в гиперкомплексных числовых системах разной размерности по представлениям в изоморфных им системах.

1.Алгебра двойных чисел.

$$W = \begin{array}{c|cc} & e_1 & e_2 \\ \hline e_1 & e_1 & e_2 \\ e_2 & e_2 & e_1 \end{array} \qquad W_1 = \begin{array}{c|cc} & E_1 & E_2 \\ \hline E_1 & E_1 & 0 \\ E_2 & 0 & E_2 \end{array}$$

Изоморфизм между этими ГЧС устанавливается следующими взаимно обратными линейными преобразованиями:

$$\begin{aligned} e_1 &= E_1 + E_2 & E_1 &= (e_1 + e_2)/2 \\ e_2 &= E_1 - E_2 & E_2 &= (e_1 - e_2)/2 \end{aligned} \qquad (8)$$

Матричные представления базисных элементов E_1 и E_2 такие:

$$M(E_1) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, M(E_2) = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \qquad (9)$$

Матричные представления базисных элементов e_1 и e_2 такие могут быть получены из левой части линейных преобразований (8) с помощью операции сложения матриц:

$$M(e_1) = M(E_1) + M(E_2) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

$$M(e_2) = M(E_1) - M(E_2) = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Можно непосредственно убедиться в том, что матричные представления (8)-(9) удовлетворяют таблице умножения системы W.

Если гиперкомплексное число имеет форму (1) при $n = 2$, то его матричное представление имеет вид:

$$M(A) = a_1 \cdot \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + a_2 \cdot \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} =$$

$$= \begin{pmatrix} a_1 + a_2 & 0 \\ 0 & a_1 - a_2 \end{pmatrix}.$$

2. Системы квадриплексных чисел K и бикомплексных чисел $C \oplus C$.

Эти гиперкомплексные числовые системы изоморфны. Их таблицы умножения имеют вид:

Система K.

	e_1	e_2	e_3	e_4
e_1	e_1	e_2	e_3	e_4
e_2	e_2	$-e_1$	e_4	$-e_3$
e_3	e_3	e_4	$-e_1$	$-e_2$
e_4	e_4	$-e_3$	$-e_2$	e_1

(10)

Система $C \oplus C$.

	E_1	E_2	E_3	E_4
E_1	E_1	E_2	0	0
E_2	E_2	$-E_1$	0	0
E_3	0	0	E_3	E_4
E_4	0	0	E_4	$-E_3$

(11)

Прямое и обратное линейные преобразования, связывающие их базисы, такие:

$$\begin{cases} e_1 = E_1 + E_3 \\ e_2 = -E_2 + E_4 \\ e_3 = -E_2 - E_4 \\ e_4 = -E_1 + E_3 \end{cases} \qquad (12)$$

$$\begin{cases} E_1 = (e_1 - e_4)/2 \\ E_2 = (-e_2 - e_3)/2 \\ E_3 = (e_1 + e_4)/2 \\ E_4 = (e_2 - e_3) \end{cases} \qquad (13)$$

Матричные представления базисных элементов системы $C \oplus C$ в соответствии с представлениями (3) и (11) имеют вид:

$$M(E_1) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, M(E_2) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$M(E_3) = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, M(E_4) = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{pmatrix}.$$

Отсюда с помощью (12) получаем матричные представления базисных элементов системы K:

$$M(e_1) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, M(e_2) = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{pmatrix},$$

$$M(e_3) = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, M(e_4) = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Можно непосредственно убедиться в том, что эти матричные представления удовлетворяют таблице умножения системы K (10).

Если гиперкомплексное число A имеет вид (1) при $n = 4$, то его матричное представление будет:

$$M(A) = \sum_{i=1}^{4} a_i M(e_i) =$$

$$= \begin{pmatrix} a_1 - a_4 & -a_2 - a_3 & 0 & 0 \\ a_2 + a_3 & a_1 - a_4 & 0 & 0 \\ 0 & 0 & a_1 + a_4 & a_2 - a_3 \\ 0 & 0 & -a_2 + a_3 & a_1 + a_4 \end{pmatrix}.$$

3. Пример ГЧС 8-й размерности.

Рассмотрим гиперкомплексную систему 8-й размерности Γ_1, полученную удвоением ГЧС квадриплексных чисел K системой W размерности 2. Это будет ГЧС размерности $n = 8$ со следующей таблицей умножения:

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
e_1	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
e_2	e_2	$-e_1$	e_4	$-e_3$	e_6	$-e_5$	e_8	$-e_7$
e_3	e_3	e_4	$-e_1$	$-e_2$	e_7	e_8	$-e_5$	$-e_6$
e_4	e_4	$-e_3$	$-e_2$	e_1	e_8	$-e_7$	$-e_6$	e_5
e_5	e_5	e_6	e_7	e_8	e_1	e_2	e_3	e_4
e_6	e_6	$-e_5$	e_8	$-e_7$	e_2	$-e_1$	e_4	$-e_3$
e_7	e_7	e_8	$-e_5$	$-e_6$	e_3	e_4	$-e_1$	$-e_2$
e_8	e_8	$-e_7$	$-e_6$	e_5	e_4	$-e_3$	$-e_2$	e_1

(14)

Ей изоморфна ГЧС Γ_2, полученная удвоением ГЧС квадриплексных чисел K системой W размерности 2, со слабозаполненной таблицей умножения диагонального вида:

	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
f_1	f_1	f_2	0	0	0	0	0	0
f_2	f_2	$-f_1$	0	0	0	0	0	0
f_3	0	0	f_3	f_4	0	0	0	0
f_4	0	0	f_4	$-f_3$	0	0	0	0
f_5	0	0	0	0	f_5	f_6	0	0
f_6	0	0	0	0	f_6	$-f_5$	0	0
f_7	0	0	0	0	0	0	f_7	f_8
f_8	0	0	0	0	0	0	f_8	$-f_7$

(15)

Их изоморфизм определяется следующим линейным преобразованием элементов базисов:

$$\begin{aligned} e_1 &= f_1 + f_3 + f_5 + f_7 & e_5 &= f_1 + f_3 - f_5 - f_7 \\ e_2 &= -f_2 + f_4 - f_6 + f_8 & e_6 &= -f_2 + f_4 + f_6 - f_8 \\ e_3 &= -f_2 - f_4 - f_6 - f_8 & e_7 &= -f_2 - f_4 + f_6 + f_8 \\ e_4 &= -f_1 + f_3 - f_5 + f_7 & e_8 &= -f_1 + f_3 + f_5 - f_7 \end{aligned} \quad (16)$$

Оно имеет обратное линейное преобразование, так как его детерминант отличен от нуля. Конкретный вид детерминанта не будем приводить, так как оно громоздко, а для дальнейших расчетов нам не понадобится.

Матричные представления ГЧС Γ_2 построить очень легко: $M(f_k)$ представляет собой матрицу размером (8×8).

Для четных k:

$M_{2i-1,2i-1} = 1;\ M_{2i,2i} = 1;\ i = 1,...,4$.

Для нечетных k:

$M_{2i-1,2i} = 1;\ M_{2i,2i-1} = -1;\ i = 1,...,4$.

Все остальные элементы матрицы равны нулю, как это видно из следующих схем:

$M(f_{2i-1})$

		f_{2i-1}	f_{2i}	
	0	0	0	0
f_{2i-1}	0	1	0	0
f_{2i}	0	0	1	0
	0	0	0	0

$M(f_{2i})$

		f_{2i-1}	f_{2i}	
	0	0	0	0
f_{2i-1}	0	0	1	0
f_{2i}	0	−1	0	0
	0	0	0	0

Матричные представления базисных элементов системы Γ_1 строятся из матричных представлений системы Γ_2 в соответствии с линейным преобразованием (16). Для $i = 1,4,5.6$ они будут представлять собой матрицы размером (8×8), все элементы которой, кроме диагональных, - нули. На диагонали будут стоять единицы со знаками в соответствии с (16):

$$\begin{aligned} diagM\ (e_1) &= +1,+1,+1,+1,+1,+1,+1,+1 \\ diagM\ (e_4) &= -1,-1,+1,+1,-1,-1,+1,+1 \\ diagM\ (e_5) &= +1,+1,+1,+1,-1,-1,-1,-1 \\ diagM\ (e_8) &= -1,-1,+1,+1,+1,+1,-1,-1 \end{aligned} \quad (17)$$

Остальные матричные представления следующие:

$$M(e_2) = \begin{pmatrix} 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{pmatrix}$$

$$M(e_3) = \begin{pmatrix} 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Матричное представление гиперкомплексного числа A вида (1) при $n=8$ также будут представлять собой матрицы размером (8×8), все элементы которой, кроме указанных ниже - нули.

Как видно из вышеприведенного, для данной ГЧС (14) при выполнении операции умножения двух гиперкомплексных чисел количество умножений вещественных чисел сохраняется, но отпадает необходимость выполнения символьных операций по умножению базисных элементов при работе с натуральной формой представления гиперкомплексных чисел

$$M(e_6)=\begin{pmatrix} 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0\\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1\\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$M(e_7)=\begin{pmatrix} 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0\\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{pmatrix}$$

$$M_{11}=a_1+a_4+a_5+a_8 \qquad M_{55}=a_1-a_4-a_5+a_8$$
$$M_{12}=-a_2-a_3-a_6-a_7 \qquad M_{56}=a_2+a_3+a_6+a_7$$
$$M_{21}=a_2+a_3+a_6+a_7 \qquad M_{65}=-a_2-a_3+a_6+a_7$$
$$M_{22}=a_1-a_4+a_5-a_8 \qquad M_{66}=a_1-a_4-a_5+a_8$$
$$M_{33}=a_1+a_4+a_5+a_8 \qquad M_{77}=a_1+a_4-a_5-a_8$$
$$M_{34}=a_2-a_3+a_6-a_7 \qquad M_{78}=a_2-a_3-a_6+a_7$$
$$M_{43}=-a_2+a_3-a_6+a_7 \qquad M_{87}=-a_2+a_3+a_6-a_7$$
$$M_{44}=a_1+a_4+a_5+a_8 \qquad M_{88}=a_1+a_4-a_5-a_8$$

Выводы

1. При использовании предлагаемого авторами метода построения матричных представлений ГЧС для большой группы ГЧС, важных для практических применений, отпадает необходимость в решении высокоразмерных систем квадратичных уравнений.

2. Использование матричных представлений ГЧС может значительно сократить объемы вычислений при математическом моделировании.

Список литературы

1. Синьков М.В. Конечномерные гиперкомплексные числовые системы. Основы теории. Применения. / М.В. Синьков, Ю.Е. Бояринова, Я.А. Калиновский. – К.: Инфодрук, 2010.- 388с.
2. Курош А.Г. Лекции по общей алгебре/ А.Г.Курош – М.:Наука, 1973. – 400с.
3. Синьков М.В. Непозиционные представления в многомерных числовых системах / М.В. Синьков, Н.М. Губарени. – К. : Наук. думка, 1979. – 140 с.
4. Калиновский Я.А. Исследование свойств изоморфизма квадриплексных и бикомплексных числовых систем / Я.А. Калиновский // Реєстрація, зберігання і обробка. даних. — 2003. — Т. 5, № 1. — С. 69–73.
5. Дослідження та використання гіперкомплексних числових систем в задачах динаміки, кінематіки та кодування інформації застосування / М.В. Синьков, Каліновський Я.О., Ю.Є. Боярінова, О.В.Федоренко, Т.Г. Постнікова, Т.В. Синькова // Пріоритети наукової співпраці ДФФД і БРФФД, Бібліотека Держфонду фундаментальних досліджень. К.: — 2007. — С.21—34.
6. Noether E. Hypercomplex Grossen und Darstellungstheorie / Noether E. // Mathematische Zeitschrift. — 1929. — B. 30 P. 641-692.
7. Синьков М.В. Повышение эффективности цифровых фильтров с помощью гиперкомплексного представления информации / Синьков М.В., Калиновский Я.А., Синькова Т.В. // Сб. науч. тр. 8-й Международной научной конференции «Теория и техника передачи, приема и обработки информации» ИИИСТ-2002. — Харьков, 2002. — С. 503–504.
8. Toyoshima H. Computationally Efficient Implementation of Hypercomplex Digital Filters / Toyoshima H. // IEICE Trans. Fundamentals. — Aug. 2002. — E85-A, 8. — P.1870-1876.

УДК 004.056.53

САИДРЕЗА МАХМАЛИ

РЕАЛИЗАЦИЯ СХЕМЫ ИДЕНТИФИКАЦИИ FFSIS НА ОСНОВЕ УМНОЖЕНИЯ БЕЗ ПЕРЕНОСОВ

В статье представлен новый вариант реализации основанной на концепции нулевых знаний схемы идентификации Фейге-Фиата-Шамира (FFSIS), которая ориентирована на идентификацию абонентов многопользовательских систем или терминальный устройств. Предлагаемая модификация FFSIS состоит в использовании математической операции умножения без переносов на полях Галуа вместо модулярного умножения. Это позволяет повысить скорость выполнения процедуры идентификации как при программной, так и при аппаратной реализации. Изложена технология использования умножения без переносов для реализации FFSIS. Приведен численный пример реализации FFSIS с использованием умножения на полях Галуа. Выполнено аналитическое сравнение времени выполнения FFSIS для предложенного варианта реализации и известного, которое показало достигаемые преимущества.

In article the new variant of zero-knowledge FFSIS (Feige Fiat Shamir Identification Scheme) implementation fitted to identification of remote abonents of multiuser systems or tample-resistant devices is presented. The proposed FFSIS modification consist of using of mathematical operation of multiplication without carry on Galois fields instead of modular multiplication. It allows to speed up of identification process for software and hardware implementation. The technology of multiplication without carry using for FFSIS implementation is set forth clearly. A numerical example for FFSIS procedure based on Galois field multiplication is given. An analytical comparison of the FFSIS processing time of both the proposed and known variants is presented, that demonstrates the improvements attained.

Введение

В современных условиях, когда информационная интеграция становится одним из решающих факторов прогресса во всех областях человеческой деятельности, проблема быстрой и надежной идентификации удаленных абонентов многопользовательских систем приобретает большое практическое значение.

Эффективное решение этой проблемы позволяет разрешить противоречие между возможностью получения информации и ограничением доступа к ней. Действительно, с развитием компьютерных технологий расширяется доступ к информации, что позволяет заметно повысить эффективность принятия решения во всех сферах человеческой деятельности. С другой стороны, расширение возможностей доступа к информации не должно сказываться на соблюдении базовых прав охраны закрытых данных государства, личности и организаций. Значительная часть данных, по своей сути, является продуктом, на получение которого затрачены ресурсы и, соответственно, эти данные имеют определенную стоимость, что обуславливает необходимость ограничения доступа к ним.

В современных условиях быстрого развития информационного общества, происходит актуализация проблемы надежной и быстрой идентификации. Это обусловлено тем, что с расширением использования интегрированных систем хранения и обработки данных, возрастает ценность потенциально доступной информации, расширяется арсенал средств незаконного доступа к ней, в том числе, за счет побочного влияния на работу подсистемы идентификации. Расширение использования беспроводных каналов передачи данных создает предпосылки для прослушивания обмена идентификационными посылками и подмены легальных абонентов во время сеанса. Все эти факторы диктуют необходимость адекватного повышения надежности идентификации, что практически всегда связано с усложнением криптографических процедур, лежащих в ее основе. С другой стороны, рост числа абонентов, опережающий темпы увеличения производительности вычислительных средств, требует повышения скорости идентификации. Выход может быть найден только за счет создания новых методов и средств идентификации, позволяющих эффективно разрешать противоречие между надежностью и производительностью.

Часто в качестве терминальных устройств абонента выступают малоразрядные портативные микроконтроллеры (смарт-карты), для которых проблема вычислительной сложности процедур идентификации стоит особенно остро.

Таким образом, в современных условиях быстрого роста информационной интеграции, задача разработки новых и совершенствования известных методов идентификации является актуальной и важной для практики.

Идентификация на основе концепции нулевых знаний. Алгоритм FFSIS

В современных условиях объективно происходит расширение возможностей для несанкционированного доступа к закрытым информационным ресурсам интегрированных систем за счет нарушения процедур идентификации.

Как уже отмечалось выше, расширение использования беспроводных технологий передачи данных позволяет злоумышленнику активнее вмешиваться в процесс идентификации. В частности, в беспроводных линиях облегчается перехват злоумышленником пароля легального абонента, а также его подмена после проведения сеанса идентификации. Классическим средством противодействия подмене является периодическое повторное проведение сеансов идентификации в процессе взаимодействия системы с абонентом. Для этого процедура идентификации должна выполняться быстро.

Еще одним путем нарушения процессов идентификации является побочное направленное вмешательство в работу системы легальными пользователями, а также посредством вирусов или недобросовестного персонала. Для широкого класса коммерческих многоабонентных систем важным является исключение возможности имитации системой обращения пользователя.

Исходя из указанных обстоятельств, современные средства идентификации абонентов должны удовлетворять следующим требованиям [1]:

1) Идентифицирующая информационная посылка (пароль) должна меняться при каждом обращении к системе, при этом используемые пароли должны быть статистически независимыми;

2) Длина пароля должна полностью исключать возможность его подбора перебором;

3) Информация, хранящаяся в системе не должна быть достаточной для воспроизведения пароля абонента;

4) Процедура идентификации должна выполняться достаточно быстро.

В литературе [2] методы идентификации, удовлетворяющие первым трем из приведенных требований называют “строгими”, в противовес остальным, которые называются “слабыми”. К классу последних относится, например, процедура идентификации, используемая в операционной системе UNIX [2]. Эта процедура предусматривает сохранение в системе только хеш-образов паролей пользователей, что, при использовании необратимых хеш-преобразований, исключает возможность воспроизведение пароля системой; однако сами пароли не меняются, что позволяет достаточно просто их перехватить.

К классу “строгих” процедур относятся, большей частью, методы идентификации, в основе которых лежит концепция “нулевых знаний”. Наиболее известным из них является метод FFSIS (Feige Fiat Shamir Identification Scheme) [2]. FESIS представляет собой относительно простую и вместе с тем достаточно эффективную схему идентификации абонентов многопользовательских систем, на основе которой создано ряд имеющих практическое значение модификаций [3]. В плане практического использования основным недостатком FFSIS считается необходимость в большом числе обменов данными в процессе идентификации, что заметно нагружает линии передач.

Другие схемы идентификации, реализующие концепцию “нулевых знаний”, предложенные в [4,5] требуют существенно меньшего объема пересылок, но предусмотренные ими процедуры имеют большую вычислительную сложность, поскольку вместо операции возведения в квадрат, в них используются операции модулярного экспоненцирования.

Сущность FESIS состоит в следующем.

Посредник (абонент) выбирает два простых числа p и q, вычисляет модуль $m=p\cdot q$. Для генерации открытого и закрытого ключей посредник(абонент) выбирает число v, являющееся квадратичным вычетом по модулю m. Другими словами, выбирается v, для которого существует x такое, что $d^2 \bmod m = v$ и существует v^{-1} такое, что $v\cdot v^{-1} \bmod m = 1$. Затем вычисляется наименьшее s для которого выполняется: $s^2 \bmod m = v^{-1}$. Число v является открытым, а число s – закрытым ключом.

При регистрации абонент посылает в систему свой открытый ключ – число v.

В цикле идентификации абонент выбирает случайное число r и вычисляет значение $x = r^2 \bmod m$, вычисленное значение x отсылает в систему. Система инициирует выполнение t циклов аккредитации. В каждом цикле аккредитации выполняются следующие действия:

1) Система посылает абоненту случайный бит b.

2) Если b=0, то абонент посылает в систему число r, в противном случае, если b=1, то абонент вычисляет с использованием закрытого ключа s значение $y = r \cdot s \bmod m$ и отсылает его системе.

3) Если b=0, то система проверяет $x = r^2 \bmod m$, в противном случае, если b=1, проверяется, что $x = y^2 \cdot v \bmod m$, убеждаясь, что абонент знает $s = \sqrt{v^{-1}}$, поскольку,

$$y^2 \cdot v \bmod m = (r^2 \cdot (\sqrt{v^{-1}})^2 \cdot v) \bmod m =$$

$$= (r^2 \cdot v^{-1} \cdot v) \bmod m = r^2 \bmod m = x$$

Если злоумышленник знает открытый ключ v легального абонента, то он может подобрать любое g и вычислить $g^2 \cdot v \bmod m = \xi$; послать системе ξ в качестве x. Если посланный в ответ системой случайный бит b=1, то злоумышленник вместо y пересылает системе код g. Соответственно, система вычисляет $g^2 \cdot v \bmod m$, сравнивает с ξ, получая при этом положительный результат сравнения. Однако, при b=0 злоумышленник должен отправить системе код $g^2 \bmod m \neq \xi$, то есть выявить попытку подделки. Если злоумышленник пошлет в качестве x код $g^2 \bmod m$, то пройдет тест при b=0, но не пройдет при b=1.

Очевидно, что идентификация с положительным результатом достижима только в случае, если злоумышленник подобрал закрытый ключ s. Решение этой задачи эквивалентно отысканию значения v^{-1} по известному v.

Наиболее значительными недостатками описанного варианта схемы идентификации FFSIS являются: необходимость в нескольких циклах аккредитации и низкое быстродействие, обусловленное тем, что на каждом цикле аккредитации необходимо выполнять три операции модулярного умножения над многоразрядными числами. Особенно ощутимым этот недостаток становится, если в качестве терминальных устройств абонентов выступают портативные контроллеры (смарт-карты), в которых выполнение операции модулярного умножения занимают много времени.

Целью исследований является созданий модификации алгоритма FFSIS, обладающей меньшей вычислительной сложностью и позволяющей повысить скорость идентификации при программной и аппаратной реализации.

Реализация схемы идентификации FFSIS На основе алгебры полей Галуа

Существенное упрощение вычислений, связанных с реализацией схемы идентификации FFSIS может быть достигнуто за счет ее реализации в алгебре без межразрядных переносов и заемов. К числу последних относится алгебра полей Галуа. В последние годы эта алгебра широко используется в кодировании и защите информации. Одним из важных ее достоинств является то, что реализация ее базовых вычислительных операций: умножения, сложения и модулярной редукции выполняется существенно быстрее по сравнению с классическими операциями умножения, сложения и нахождением остатка.

В алгебре полей Галуа операция суммирования фактически соответствует логическому сложению (XOR). Операция умножения выполняется без межразрядных переносов и ниже обозначается знаком $\otimes$.

Произведение без межразрядных переносов $D=U \otimes V=\{d_{2 \cdot r},\ldots,d_3,d_2,d_1\}=d_1+2 \cdot d_2+4 \cdot d_3+\ldots+2^{2 \cdot r} \cdot d_{2 \cdot r}$, $\forall l \in \{1,..,2 \cdot r\}$: $d_l \in \{0,1\}$ двух r-разрядных чисел $U=\{u_r,\ldots,u_2,u_1\}=u_1+2 \cdot u_2+\ldots+2^r \cdot u_r$ и $V=\{v_r,\ldots,v_2,v_1\}=v_1+2 \cdot v_2+\ldots+2^r \cdot v_r$, $\forall i \in \{1,..,r\}$: $v_i,u_i \in \{0,1\}$ вычисляется следующим образом:

$$D = U \cdot v_1 \oplus (U \cdot v_2) << 1 \oplus \ldots \oplus (U \cdot v_r) << (r-1), \quad (1)$$

где $\cdot$ – операция логического умножения, $p<<q$ – операция логического сдвига числа p влево на q разрядов.

Сущность предлагаемой реализации схемы идентификации FFSIS в алгебре полей Галуа состоит в следующем.

На этапе регистрации абонентом (посредником) выполняется генерация открытого и закрытого ключей выбранной разрядности – n. Для этого выбирается простое в рассматриваемой алгебре (n+1)-разрядное число m. Например, если в порядке иллюстрации положить малую разрядность n=4, то можно выбрать в качестве модуля 5-разрядное $m=11001_2=25$.

Далее, абонент произвольно выбирает целое (n-1)-разрядное число η, после чего выполняется разложение (2·n-1)-разрядного числа $\eta \otimes m \oplus 1$ на n-разрядные множители d и s так, что $d \otimes s = \eta \otimes m \oplus 1$. Например, если η=4 и m=25, то $\eta \otimes m \oplus 1=101=9 \otimes 13$. Соответственно, d=9 и s=13. После этого вычисляются $v=d^2 \bmod m$ и $v^{-1} = s^2 \bmod m$; очевидно, что $v \cdot v^{-1} \bmod m = 1$. В рамках рас-

сматриваемого примера v=(9⊗9) mod m =14 и v^{-1} = (13⊗13) mod m = 7;

Число v является открытым, а число s – закрытым ключом.

При регистрации абонент посылает в систему свой открытый ключ – число v. v=14

При идентификации абонент случайным образом генерирует r, например, r=10; вычисляет d = (r^2) mod m и посылает вычисленное значение d в систему. Для примера d=10⊗10 mod 25 = 11. После этого вычисляет y=(r ⊗s) mod m и значение y также отсылает в систему. y=10⊗13 mod 25 = 15.

По получении указанных кодов, система, в соответствии со схемой FFSIS, инициирует выполнение t циклов аккредитации. В каждом цикле аккредитации выполняются следующие действия:

1) Система посылает абоненту случайный бит b.

2) Если b=0, то абонент посылает в систему число r, в противном случае, если b=1, то абонент вычисляет с использованием закрытого ключа s значение $y = r⊗s$ mod m и отсылает его системе.

3) Если b=0, то система проверяет справедливость $d = r^2$ mod m, в противном случае, если b=1, проверяется, что $d = y^2⊗v$ mod m, убеждаясь, что абонент знает $s = \sqrt{v^{-1}}$.

Например, пусть b=1 и абонент вычисляет значение y=(r ⊗s) mod m =10⊗13 mod 25 = 15 и значение y отсылает в систему. Система, получив, y = 15, вычисляет с использованием закрытого ключа v=14 z = 15⊗15⊗14 mod 25 =11. Поскольку z=11=d, то тест считается пройденным.

Если b=0, то абонент посылает в систему число r=10; $z = r^2$ mod m = 10⊗10 mod 25 = 11 и сравнивает его с ранее полученным значением d=11.

Таким образом, показано, что схема идентификации FFSIS работает при ее реализации в алгебре полей Галуа.

Оценка эффективности реализации FFSIS на полях Галуа

При программной реализации модулярного умножения n-разрядных чисел на w-разрядном процессоре, числа разбиваются на s фрагментов (s=n/w). Каждый фрагмент множителя умножается на каждый фрагмент множимого с формированием 2·w-разрядного произведения. Полученные произведения суммируются. Таким образом, при умножении n-разрядных чисел на w-разрядном процессоре общее число операций умножения w-разрядных чисел составляет s^2, а операций арифметического суммирования – $2 \cdot s^2$. Поскольку каждая из операций арифметического суммирования с вероятностью 0.5 сопровождается возникновением переноса, для учета которого необходимо повторно выполнять операцию инкрементирования (суммирования), то общее число операций сложения составляет $3 \cdot s^2$.

Нахождение остатка от деления n-разрядных чисел на w-разрядном процессоре выполняется весьма неэффективно: фактически n раз выполняется операция сравнения (вычитания) из произведения сдвигаемого модуля. При этом, в каждом из n циклов выполняется, в среднем, 1.5·s операций арифметического вычитания и s операций сдвига.

Таким образом, при программной реализации модулярного умножения на современных процессорах, время модулярного умножения $T_{мм}$ определяется следующим выражением:

$$T_{mm} = s^2 \cdot t_m + 1.5 \cdot s \cdot (2 \cdot s + n) \cdot t_a + n \cdot s \cdot t_S \qquad (2)$$

где t_m – время выполнения команды умножения, t_a – время выполнения команды арифметического сложения (вычитания), t_s – время выполнения команды сдвига. Согласно [6], команда умножения в современных процессорах занимает 10 тактов, арифметического сложения – 3 такта, сдвига – 1 такт, так, что формула (2) в оценочном плане может быть преобразована к виду:

$$T_{mm} \approx (19 \cdot s^2 + 5.5 \cdot s \cdot n) \cdot \tau \,, \qquad (3)$$

где τ – длительность такта.

При реализации операции умножения без переносов n-разрядных чисел на w-разрядном процессоре, числа также разбиваются на s фрагментов (s=n/w). Организуется поразрядная обработка n разрядов множителя, причем в каждом из n циклов выполняется логический сдвиг s фрагментов множимого и с половинной вероятностью осуществляется s операций логического суммирования. Нахождение остатка при полиномиальном делении полученного 2·n-произведения на образующий полином требует цикла последовательной обработки n старших разрядов полученного произведения. На каждом цикле выполняется операция сдвига кода образующего полинома и, с половинной вероятностью – s операций логического суммирования. Таким образом, время выполнения умно-

жения без переноса равно времени нахождения остатка (редуцирования), так, что суммарное время T_{mG} программной реализации умножения на поле Галуа над n-разрядных чисел на w-разрядном процессоре определяется формулой:

$$T_{mG} = 2 \cdot n \cdot s \cdot (0.5 \cdot t_X + t_S), \qquad (4)$$

где t_x – время выполнения команды логического сложения (XOR), которое согласно данным [6] занимает один такт, так, что в оценочном плане формула (4) может быть представлена в виде:

$$T_{mG} \approx 3 \cdot n \cdot s \cdot \tau. \qquad (5)$$

Сравнение выражений (4) и (5) свидетельствует о том, что время программной реализации операции умножения на поле Галуа, по меньшей мере, в 5.5/3 = 1.83 меньше времени модулярного умножения при одинаковой разрядности модуля и степени образующего полинома поля Галуа. Например, при n=1024 w=32 s=32 $T_{mm} = (19 \cdot 32^2 + 5.5 \cdot 32^3) \cdot \tau = 199680 \cdot \tau$, а $T_{\text{mG}} = 3 \cdot 32^3 \cdot \tau = 98304$, так, что соотношение времен программной реализации модулярного умножения и умножения на поле Галуа составляет:

$$h = \frac{T_{mm}}{T_{mG}} = \frac{199680}{98304} \approx 2 \qquad (6)$$

Ускорение программной реализации достигается за счет двух факторов: увеличении двое числа операций процессорного сложения из-за необходимости учета возникающих переносов; увеличения вдвое числа операций вычитания (сравнения).

При использовании в качестве терминальных устройств абонентов 8-разрядных смарт-карт, сокращение времени умножения при переходе к алгебре на полях Галуа получается еще более значительным. Так, при n=1024 w=8 s=128 $T_{mm} = (19 \cdot 128^2 + 5.5 \cdot 1024 \cdot 128) \cdot \tau = 1032192 \cdot \tau$, а $T_{\text{mG}} = 3 \cdot 1024 \cdot 128 \cdot \tau = 311296$, так, что соотношение времен программной реализации модулярного умножения и умножения на поле Галуа для смарт-карт составляет:

$$h = \frac{T_{mm}}{T_{mG}} = \frac{1032192}{311296} \approx 2.65. \qquad (7)$$

Практически, поскольку, при проведении цикла аккредитации выполняется две операции модулярного умножения, то переход к алгебре на полях Галуа позволяет примерно в 4 раза ускорить процесс идентификации. Проведенные экспериментальные исследования, в основном, подтвердили приведенные теоретические оценки. При аппаратной реализации, переход от модулярного умножения в традиционной алгебре к умножению на полях Галуа сопряжен с существенно большим выигрышем как в плане быстродействия, так в плане сложности схемы.

Базовой операцией модулярного умножения является арифметическое сложение, выполняемое над n-разрядными числами. Учет возникающего при сложении переноса выполняется путем выполнения еще одной операции сложения над n-разрядными числами, так, что каждое сложение реально требует выполнения, в среднем, 1.5 операций. При последовательном выполнении переноса время суммирования T_{as}, в оценочном плане, равно $1.5 \cdot t_3$, где t_3 – время срабатывания схемы 3-х входового сумматора; поскольку это время равно $t_3 = 3 \cdot t$, где t – время срабатывания двухвходового логического элемента, то, можно полагать, в первом приближении, что время суммирования равно $T_{as} = 4.5 \cdot n \cdot t$. Сложность (количество двухвходовых логических элементов) схемы последовательного арифметического суммирования S_{as} равна $S_{as} = n \cdot s_3$, где $s_3 = 6$ – число логических элементов, требующихся для реализации 3-х входового сумматора, соответственно, $S_{as} = 6 \cdot n$. При использовании схем ускоренного переноса, время арифметического суммирования T_{ap} определяется в виде: $T_{ap} = 1.5 \cdot (t_3 + t_c)$, где t_c – максимальное время формирования сигнала переноса, которое составляет $\log_2 n \cdot t$. таким образом, $T_{ap} \approx 1.5 \cdot \log_2 n$. При этом сложность схемы формирования переносов, в оценочном плане, определяется как $6 \cdot n^3$. Время T_L выполнения базовой операции умножения на полях Галуа – логического сложения равно t, а сложность S_L соответствующей схемы – n.

Сравнительные оценки времени выполнения и сложности аппаратной реализации базовой операции сложения в традиционной алгебре и на полях Галуа приведены в таблице 1.

Табл. 1. Соотношение времени выполнение и сложности схемы при аппаратной реализации арифметического и логического сложения.

	$\frac{T_a}{T_L}$	$\frac{S_a}{S_L}$
Последовательное формирование переноса	$4.5 \cdot n$ при n=1024 : 4608	6 при n=1024 : 6
Ускоренное формирование переноса	$1.5 \cdot \log_2 n$ при n=1024 : 15	$6 \cdot n^2$ при n=1024 : 6144

Выводы

Предложен способ реализации широко известной процедуры FFSIS строгой идентификации удаленных абонентов многопользовательских систем в рамках концепции “нулевых знаний” на основе операций без переносов на полях Галуа. Доказано, что использование таких операций не влияет на уровень защищенности, но позволяет существенно упростить и ускорить вычислительные процедуры FFSIS. Приведенная технология использования операций умножения на полях Галуа иллюстрирована примером. Доказано, что при программной реализации применение предложенного подхода позволяет, при практически используемых разрядностях чисел, примерно в 4 раза ускорить идентификации FFSIS.

Проведенными исследованиями показано, что при аппаратной реализации предложенный подход позволяет существенно ускорить процесс идентификации и заметно упростить схему. Так, если сравнивать предложенный вариант со схемой выполнения арифметических операций с последовательным переносом, предложенный подход позволяет ускорить процедуру идентификации в соответствии с FFSIS на три порядка. При сравнении со схемой с ускоренным переносом, предложенный вариант обеспечивает ускорение в 15 раз при том, что сложность схемы упрощается на 3 порядка.

Предложенный способ реализации процедуры FFSIS может быть использован для повышения эффективности идентификации как удаленных абонентов компьютерных сетей, так и для идентификации терминальный мобильных устройств (смарт-карт).

Список литературы

1. Bardis N., Doukas N. and Markovskyi O., Fast subscriber identification based on the zero knowledge principle for multimedia content distribution // International Journal of Multimedia Intelligence and Security.- 2010 - Vol.1.- № 4. - P. 363 - 377.
2. Feige U., Fiat A., Shamir. Zero Knowledge Profs of Identity // Journal of Cryptography.- 1988.- v.1- № 2.- P.77-94.
3. Micali S., Shamir A. An Improvement on the Feige-Fiat-Shamir Identification and Signature Schemes // Advances of Cryptology -Crypto-88. Proceeding.- Springer-Verlag.-190.- P. 244-247.
4. Guillou L.C., Quisquater J.-J. A Paradoxical Identity-Based Signature Schemes Resulting from Zero Knowledge // Advances of Cryptology -Crypto-88. Proceeding.- Springer-Verlag.-1990.- P. 216-231.
5. Cocks C. An identity based encryption schemes based on quadratic residues // Proceeding of the 6-th International Conference on Cryptography and Coding. - IMA.Press.-2001.- P.26-28.
6. Брэй Б. Микропроцессоры Intel. Архитектура, программирование и интерфейсы. - СПб.: Изд-во “БХВ-Петрербург”.- 2005.- 1328 С.

УДК 629.735.017

НЕЧИПОРЕНКО О. М.

КРИТЕРІЙ ПАРАМЕТРИЧНОЇ НАДІЙНОСТІ РОБАСТНО СТІЙКИХ СИСТЕМ АВТОМАТИЧНОГО КЕРУВАННЯ

Пропонується нормувати час переходу САК з початку її функціонування до моменту втрати параметричної (робастної) стійкості *робастним критерієм її відмови*, який обмежує допустиме значення призначеного ресурсу САК при її проектуванні. Такий критерій дозволяє враховувати процеси фізичної деградації САК під час її експлуатації при нормуванні часу наступного підрегулювання (тривалості межремонтного періоду).

Time of transition of the automatic control system (ACS) from the beginning of its functioning to the moment of loss of self-reactance (robast) stability it is suggested to ration the robust criterion of its refuse which limits the assumed value of the appointed resource of ACS at its planning. Such criterion allows to take into account the processes of physical degradation of ACS during its exploitation at setting of norms of time of the subsequent subadjusting (durations of the TBO period).

1. Вступ

При аналізі надійності об’єкту або системи існують проблеми визначення моменту виникнення порушення (втрати) його працездатності – наробітку до відмови [1, 2, 3, 4] для нормування, наприклад, допустимого значення призначеного ресурсу САК при її проектуванні. Якщо процеси втрати працездатності САК (за рахунок зношування, старіння, утоми, зміни зовнішніх умов тощо) можна визначити функціональною залежністю її параметрів від часу, розглядають *параметричні* відмови (*параметричну надійність* САК).

Існують ймовірнісно-фізичні моделі наближення об’єкта до параметричної відмови [5]. Визначення розподілу наробітку до першої відмови об’єктів у такому випадку зводиться до вирішування завдання першого досягнення процесом граничного рівня і пов’язано з розв’язанням рівняння дифузії (Фоккера-Планка-Колмогорова), яке є диференціальним рівнянням у частинних похідних та під час розв’язування потребує встановлення *граничних умов* залежно від вигляду реалізацій процесу. У випадку втрати стійкості САК граничною умовою можна вважати момент досягнення межі її стійкості.

Новизна роботи полягає в поєднанні класичних показників безвідмовності технічних об’єктів з критерієм парамертичної стійкості САК: запропоновано використовувати критерій стійкості робастно стійкої системи як ***робастний критерій її відмови***. Такий критерій також дозволить враховувати процеси фізичної деградації САК під час її експлуатації при нормуванні часу наступного підрегулювання (тривалості межремонтного періоду).

В роботі поширено умову робастної стійкості на нестаціонарні системи і використано час виходу САК на межу стійкості як наробіток до її відмови.

2. Аналіз проблеми

Відомо, що при зміні визначального параметра [2] у часі і досягненні ним межі допуску виникає відмова об’єкта [2, 4]. Крім того, втрату стійкості системи можна розглядати як її відмову: нестійка система, яка знаходиться на межі (аперіодичної або коливальної) стійкості, не є працездатною: будь-яке незначне відхилення призводить до її відмови.

Якщо ситема параметрично нестійка, виникає проблема визначення меж її стійкості в залежності від значення визначального параметра (ВП). Задачу визначення меж параметричної стійкості в теорії автоматичного регулювання вирішує *критерій робастної стійкості системи*. Але цей критерій розглянуто тільки для стаціонарних систем [6]. З точки зору надійності, при знаходженні меж робастної стійкості системи необхідно врахо вувати ***час*** досягнення цієї межі визначальним параметром системи.

3. Постановка задачі

Параметри стаціонарних *робастно стійких* систем із часом у силу старіння або інших причин можуть мінятися, тобто такі системи можна розглядати як *квазі стаціонарні* або *неста-*

ціонарні. У подібних випадках виникає задача побудови систем керування таким чином, щоб вона була стійка не при одних фіксованих значеннях параметрів, а при всіх можливих їх значеннях.

Мета статті – представити параметри стаціонарної системи як функції часу і розглянути змінювання у часі *визначальних* параметрів САК. У даному випадку (гіпер)паралелепіпед Харитонова [6] буде мати вершини, що змінюються у часі. Момент часу, при якому умова робастної стійкості (всі поліноми Харитонова стійкі) не стане виконуватися, буде визначати момент виникнення відмови САК (наробіток системи до відмови), який пропонується розглядати, як критерій робастної стійкості системи.

4. Визначення параметричної надійності САК

Якщо відмова розглядається як вихід за припустимі межі значення параметра $A(t)$ об'єкта, що відбувається через зміни цього параметра у часі t (у загальному випадку у функції будь-якої монотонно зростаючої величини – наробітку), то ці відмови називають ***параметричними*** [2]. Надійність об'єкта у разі параметричних відмов визначається функціоналом деякого випадкового процесу $\{t\}$, який характеризує зміну параметрів об'єкта у часі. Об'єкт лишається працездатним, поки величина $A(t)$, що змінюється у часі, не досягає межі припустимої робочої області Ω.

Основний технічний параметр, який характеризує працездатність об'єкта і визначає його міру якості, називають ***визначальним параметром (ВП)***.

У загальному випадку ВП може бути вектором, тобто мати кілька складових. Граничні значення, які установлюються на кожен ВП об'єкта, є ***припустимими*** значеннями ВП, котрі обмежують ***робочу область (поле допуску)***, яку задають у НТД.

Поки значення векторного ВП об'єкта перебувають усередині багатовимірної робочої області, об'єкт вважають працездатним. Однак із часом під впливом факторів, пов'язаних зі старінням, зношуванням або розрегулюванням, кінець вектора $A(t)$ може досягти межі робочої області Ω. При цьому об'єкт втрачає працездатність (відбувається відмова).

У практиці експлуатації об'єкта більш важливо знати не щільність розподілу часу до відмови, а конкретний ***час збереження працездатності*** $t_{çá}$**,** протягом якого ВП досягає межі робочої області.

Для вирішення цієї задачі пропонується розглядати час втрати стійкості робастно стійкої САК як час досягнення ВП меж робочої області.

5. Робастна стійкість нестаціонарних САК

У разі старіння або інших необоротних причин у робастно стійких системах виникає необхідність побудови системи керування так, щоб вона була стійка не при одних фіксованих значеннях параметрів, а при всіх можливих їх значеннях, які функціонально змінюються у часі. У останньому випадку можна вважати систему *нестаціонарною*, а при плавних повільних змінах ВП у часі в межах робастної стійкості – стійкою.

Робастна стійкість САК визначається таким чином [6]. Характеристичний поліном САК $D(\lambda) = a_0\lambda^n + a_1\lambda^{n-1} + ... + a_n$ називається *стійким поліномом* або *поліномом Гурвіця,* якщо всі його нулі є лівими.

Якщо позначити $(n+1)$-вектор $a = (a_0,\ a_1,\ ...,\ a_n)$ і в $(n+1)$-мірному просторі коефіцієнтів розглянути множину A ($A \subset R^{n+1}$), то поліном $D(\lambda)$ буде *робастно стійким* або *робастно стійким в множині A,* якщо він є стійким при будь-яких значеннях коефіцієнтів a_i $(i = 1,\ 2,\ ...,\ n)$ з множини A ($a \in A$).

Система буде *робастно стійкою* або *робастно стійкою на множині A,* якщо її характеристичний поліном є робастно стійким поліномом в A.

Для аналізу робастно стійких систем введено поліноми Харитонова [6].

Нехай множина A є (гіпер) паралелепіпедом:

$$A = \{\underline{a}:\ \underline{a}_i \le a_i \le \overline{a}_i,\ i = 0,\ 1,\ ...,\ n\}, \qquad (1)$$

де $\underline{a}_i$ і $\overline{a}_i$ – мінімальне і максимальне значення коефіцієнта a_i $(i = 1,\ 2,\ ...,\ n)$.

Підставимо в характеристичний поліном $\lambda = j\omega$ і виділимо дійсну і уявну частини:

$$Q(j\omega) = a_0(j\omega)^n + a_1(j\omega)^{n-1} + ... + a_n = u(\omega) + j\upsilon(\omega),$$

$$u(\omega) = a_n - a_{n-2}\omega^2 + a_{n-4}\omega^4 - a_{n-6}\omega^6 + ..., \qquad (2a)$$

$$\upsilon(\omega) = a_{n-1}\omega - a_{n-3}\omega^3 + a_{n-5}\omega^5 - a_{n-7}\omega^7 + \ldots \qquad (2б)$$

При фіксованому ω, коли вектор a пробігає всі значення з множини (1), характеристичний вектор описує прямокутник (рис. 1).

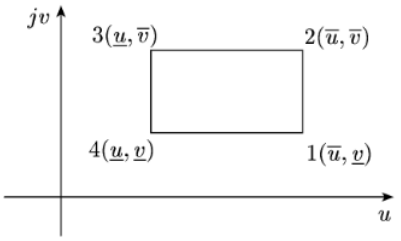


Рис. 1. Графік, що пояснює визначення поліномів Харитонова

Очевидно, що на вершинах прямокутника $u(\omega)$ і $\upsilon(\omega)$, як функції від a, приймають мінімальні або максимальні значення. На рис. 1 позначено мінімуми $u(\omega)$ і $\upsilon(\omega)$ – через $\underline{u}(\omega)$ і $\underline{\upsilon}(\omega)$, а максимуми – через $\overline{u}(\omega)$ і $\overline{\upsilon}(\omega)$ відповідно:

$$\underline{u} = \underline{u}(\omega) = \min_{a\in A} u(\omega), \quad \underline{\upsilon} = \underline{\upsilon}(\omega) = \min_{a\in A} \upsilon(\omega),$$

$$\overline{u} = \overline{u}(\omega) = \max_{a\in A} u(\omega), \quad \overline{\upsilon} = \overline{\upsilon}(\omega) = \max_{a\in A} \upsilon(\omega).$$

Функції $u(\omega)$ і $\upsilon(\omega)$ приймуть мінімальні значення, коли в (2а) і (2б) доданки з додатним знаком приймають мінімальні значення, а доданки з від'ємним знаком – максимальні значення. І навпаки, $u(\omega)$ і $\upsilon(\omega)$ приймуть максимальні значення, коли доданки з додатним знаком приймають максимальні значення, а доданки з від'ємним знаком – мінімальні. Тому з (2а) і (2б) маємо

$$\underline{u}(\omega) = \underline{a}_n - \overline{a}_{n-2}\omega^2 + \underline{a}_{n-4}\omega^4 - \overline{a}_{n-6}\omega^6 + \ldots, \qquad (3а)$$

$$\underline{\upsilon}(\omega) = \underline{a}_{n-1}\omega - \overline{a}_{n-3}\omega^3 + \underline{a}_{n-5}\omega^5 - \overline{a}_{n-7}\omega^7 + \ldots. \qquad (3б)$$

$$\overline{u}(\omega) = \overline{a}_n - \underline{a}_{n-2}\omega^2 + \overline{a}_{n-4}\omega^4 - \underline{a}_{n-6}\omega^6 + \ldots, \qquad (3в)$$

$$\overline{\upsilon}(\omega) = \overline{a}_{n-1}\omega - \underline{a}_{n-3}\omega^3 + \overline{a}_{n-5}\omega^5 - \underline{a}_{n-7}\omega^7 + \ldots. \qquad (3г)$$

Як випливає з рис. 1, вершинам прямокутника 1, 2, 3 і 4 відповідають характеристичні вектори

$$D_1(j\omega) = \overline{u}(\omega) + j\underline{\upsilon}(\omega), \quad D_2(j\omega) = \overline{u}(\omega) + j\overline{\upsilon}(\omega),$$

$$D_3(j\omega) = \underline{u}(\omega) + j\overline{\upsilon}(\omega), \quad D_4(j\omega) = \underline{u}(\omega) + j\underline{\upsilon}(\omega).$$

Підставивши у формулу для $D_1(j\omega)$ вирази для $\overline{u}(\omega)$ з (3в) і для $\underline{\upsilon}(\omega)$ з (3б), отримаємо

$$D_1(j\omega) = \overline{a}_n - \underline{a}_{n-2}\omega^2 + \overline{a}_{n-4}\omega^4 - \ldots$$

$$\ldots + j(\underline{a}_{n-1}\omega - \overline{a}_{n-3}\omega^3 + \underline{a}_{n-5}\omega^5 - \ldots) = \overline{a}_n + \underline{a}_{n-1}(j\omega) +$$

$$\underline{a}_{n-2}(j\omega)^2 + \overline{a}_{n-3}(j\omega)^3 + \overline{a}_{n-4}(j\omega)^4 + \underline{a}_{n-5}\omega^5 + \ldots.$$

Звідси, поклавши $j\omega = \lambda$, отримаємо характеристичний поліном $D_1(\lambda)$. Аналогічно можна отримати характеристичні поліноми, що відповідають решті вершин прямокутника.

Випишемо коефіцієнти при λ в порядку зростання степеня λ всіх чотирьох поліномів:

$$D_1(\lambda):\ \overline{a}_n,\ \underline{a}_{n-1},\ \underline{a}_{n-2},\ \overline{a}_{n-3},\ \overline{a}_{n-4},\ \underline{a}_{n-5},\ \ldots, \qquad (4а)$$

$$D_2(\lambda):\ \overline{a}_n,\ \overline{a}_{n-1},\ \underline{a}_{n-2},\ \underline{a}_{n-3},\ \overline{a}_{n-4},\ \overline{a}_{n-5},\ \ldots, \qquad (4б)$$

$$D_3(\lambda):\ \underline{a}_n,\ \overline{a}_{n-1},\ \overline{a}_{n-2},\ \underline{a}_{n-3},\ \underline{a}_{n-4},\ \overline{a}_{n-5},\ \ldots, \qquad (4в)$$

$$D_4(\lambda):\ \underline{a}_n,\ \underline{a}_{n-1},\ \overline{a}_{n-2},\ \overline{a}_{n-3},\ \underline{a}_{n-4},\ \underline{a}_{n-5},\ \ldots. \qquad (4г)$$

Поліноми $D_1(\lambda)$ $D_2(\lambda)$ $D_3(\lambda)$ і $D_4(\lambda)$ - поліноми Харитонова.

Необхідна умова робастної стійкості. Оскільки при робастній стійкості в паралелепіпеді (2) повинні бути стійкими характеристичні поліноми при всіх значеннях коефіцієнтів з цього паралелепіпеда, необхідно, щоб був стійким характеристичний поліном при $a_i = \underline{a}_i$ $(i = 1,\ 2,\ \ldots,\ n)$. Тому для робастної стійкості в паралелепіпеді (2) необхідно, щоб при $\underline{a}_0 > 0$ виконувалися умови

$$\underline{a}_1 > 0,\ \underline{a}_2 > 0,\ \ldots,\ \underline{a}_n > 0. \qquad (5)$$

Згідно теореми Харитонова, для того, щоб система з характеристичним поліномом $D(\lambda) = a_0\lambda^n + a_1\lambda^{n-1} + \ldots + a_n$ була робастно стійкою в паралелепіпеді

$$A = \left\{\underline{a}:\ \underline{a}_i \le a_i \le \overline{a}_i,\ i = 0,\ 1,\ \ldots,\ n\right\},$$

необхідно і достатньо, щоб всі поліноми Харитонова були стійкими.

Оскільки за визначенням робастної стійкості характеристичний поліном повинен бути стійким при всіх значеннях $a \in A$, то повинні бути стійкими і поліноми Харитонова як характеристичні поліноми, відповідні чотирьом різним значенням a з множини A. За критерієм Михайлова для робастної стійкості при $a_0 > 0$ достатньо, щоб годограф характеристичного вектора при будь-яких $a \in A$, почавшись на додатній дійсній піввісі, послідовно охоплював n квадрантів. Інакше кажучи, прямокутник на рис. 1 повинен послідовно охоплювати n квадрантів. Дійсна і уявна частини характеристичного вектора $D'(j\omega) = u'(\omega) + j\upsilon'(\omega)$, відповідного довільному $a' \in A$, задовольняє нерівностям

$$\underline{u}(\omega) \le u'(\omega) \le \overline{u}(\omega),\ \ \underline{\upsilon}(\omega) \le \upsilon'(\omega) \le \overline{\upsilon}(\omega).$$

Тому, якщо вершини прямокутника послідовно охоплюють n квадрантів, то і всі точки прямокутника послідовно охоплюватимуть n квадрантів.

Для того, щоб система з характеристичним поліномом $D(\lambda) = a_0\lambda^n + a_1\lambda^{n-1} + ... + a_n$ була робастно стійка в паралелепіпеді при виконанні необхідної умови робастної стійкості (5), необхідно і достатньо, щоб були стійкими:

а) у разі $n = 3$ поліном Харитонова $D_1(\lambda)$;

б) у разі $n = 4$ поліноми Харитонова $D_1(\lambda)$ і $D_2(\lambda)$;

в) у разі $n = 5$ поліноми Харитонова $D_1(\lambda)$, $D_2(\lambda)$ і $D_3(\lambda)$.

6. Робастний критерій відмови САК

Якщо при знаходженні меж робастної стійкості системи замість деякого значення ВП a_i множини параметрів A (див. вираз (1)) використовувати залежність цього параметра від часу $a_i(t)$, а потім розрахувати *час* досягнення цієї межі визначальним параметром САК, то таким чином можна отримати час $t_{\text{äð}}$, що визначає наробіток системи до відмови.

Тривалість досягнення межі робастної стійкості при змінюванні ВП САК у часі можна розглядати як ***робастний критерій відмови системи*** $t_{\text{äð}}$. Тоді (гіпер)паралелепіпед Харитонова [6] буде мати вершини, що змінюються у часі. Момент часу, при якому умови робастної стійкості (всі поліноми Харитонова стійкі) не стануть виконуватися, буде визначати момент виникнення відмови САК, який можна розглядати як критерій відмови.

Цей критерій можна використовувати для визначення впливу зміни одного з параметрів САК (наприклад, визначального) на її стійкість. Граничне значення ВП $a_{i\text{äð}}$, після якого система стає нестійкою, і буде тим значенням досліджуваного параметра, при якому визначається значення часу $t_{\text{äð}}$ (рис. 2).

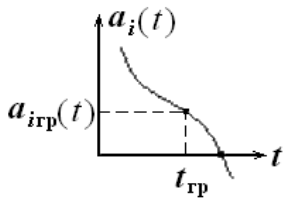


Рис. 2. Визначення граничного значення параметра $a_{i\text{äð}}$, яке знаходиться на межі стійкості системи

7. Параметрична надійність САК для випадкових змін ВП

Вище розглянуто випадок, коли змінювання ВП a_i у часі *детерміновані*. У загальній постановці завдання час досягнення ВП меж робочої області можна розглядати як систему *випадкових* величин або *векторний випадковий процес.*

Розглянемо характер випадкового процесу наближення до відмови на прикладі САК, працездатність якої визначається скалярним ВП A (однією координатою векторного ВП). При цьому простір ВП A буде одновимірним, а робоча область Ω обмежена відрізком прямої (граничне значення ВП $a_{\text{гр}}$). Нехай є множина $j = \overline{1, n}$ однакових САК, одночасно включених у роботу (за $t = 0$), і ВП кожної САК вимірюється у ті самі моменти часу t_i ($i = \overline{1, k}$).

Зміну ВП однакових САК у процесі експлуатації будемо розглядати як випадкову функцію часу $A(t)$. Для кожної j-ої САК ($j = \overline{1, n}$) зміна ВП є реалізацією (складовою) $A_j(t)$ випадкової функції $A(t)$. Точки перетину реалізацій $A_j(t)$ випадкового процесу із межею $A_{\text{äð}}$ робочої області (поля допуску) відповідають моментам часу відмов j-их САК. Тому випадковий характер виникнення поступових відмов у процесі експлуатації однакових систем описується щільністю розподілу $f\{X(t)\}$ часу перетину ВП межі $A_{\text{äð}}$, тобто щільністю розподілу часу до відмови.

Якщо з моменту включення в роботу (за $t = 0$), вимірюючи з однаковою $\Delta t = t_{i+1} - t_i = t_i - t_{i-1}$ або різною періодичністю (інтервалом) Δt, контролювати значення ВП кожної j-ої САК, то можна прогнозувати (екстраполювати) подальші зміни ВП, а отже, передбачити момент настання відмови. Це дасть можливість організувати технічне обслуговування таких САК, тобто забезпечити попереджувальне виведення їх на поточний або капітальний ремонт, або відправлення на регулювання. Інтервал часу від початку експлуатації $t = 0$ до моменту, коли вихід окремих реалізацій $A_j(t)$ випадкового процесу $A(t)$ за межі $A_{\text{äð}}$ робочої області стає частим явищем, називають ***часом збереження працездатності*** $t_{\text{çá}}$. Правий кінець інтервалу $t_{\text{çá}}$ визначається

абсцисою характерної точки кривої ЩРВ $f\{X(t)\}$, починаючи з якої спостерігається різке зростання кривої.

Маючи інформацію про реальне значення часу досягнення ВП граничного значення $t_r < t_{çá}$ на етапі проектування, можна аналітично розрахувати час збереження працездатності ситеми, тобто зробити обґрунтований прогноз про працездатність у майбутньому. Це дасть змогу вчасно попередити відмови, а також керувати станом складних САК, замінюючи їх елементи резервними, проводячи підрегулювання або змінюючи робочі режими САК.

Нестаціонарний випадковий процес A(t) характеризує довгострокові необоротні зміни параметрів у результаті зношування, старіння або розрегулювання. Процес $A(t)$ – основна причина відмов, його можна назвати ***процесом зношування.***

Для випадкових процесів зношування типовими є досить жорсткі зв'язки між значеннями параметра у послідовні моменти часу. Великий вплив на вид реалізації процесу $A(t)$ справляє фізико-хімічна структура матеріалу і технологія виготовлення об'єкта. Однотипні об'єкти дають близькі за формою криві зношування, але з різними значеннями швидкості зношування. Тому моделі процесів зношування повинні бути функціонально залежними від часу, а їх випадковий характер обумовлюється випадковими параметрами, що не залежать від часу. Подібні випадкові процеси іноді називають ***детермінізованими*** або ***напіввипадковими.***

Випадковий процес $A(t)$ зношування можна розглядати як залежність

$$A(t) = A_0 + \int_0^t B(\tau)d\tau, \quad \tau \in [0,\ t],$$

де A_0 – початкове (заводське, фабричне, промислове, виготовлене, настроєне тощо) значення ВП; $B(t)$ – напіввипадковий процес зміни *швидкості* зношування. Початкове значення A_0 ВП є випадковою величиною, що іноді має усічений (через допуск підприємства-виробника) розподіл, але не залежить від часу *t*.

Як відомо, основою випадкових процесів зміни ВП є необоротні випадкові зміни ВП, викликані старінням, зношуванням або розрегулюванням. Вони певною мірою залежні від часу, їх можна розглядати (з деякою мірою ймовірності) як *поступові*. При цьому *випадковий* характер таких змін обумовлений *випадковими* параметрами, що *не залежать від часу*. Отже, моделі реальної зміни ВП об'єкта мають бути випадкововими функціями, аргументами яких є постійні у часі випадкові величини й сам час.

Розглянемо найпоширеніші моделі (класи моделей) нестаціонарних випадкових процесів наближення до відмов.

Лінійні випадкові функції. Під час лінеаризації реального процесу зношування об'єкта кожна реалізація $A_j(t)$ процесу замінюється прямою, тобто реальний процес зміни ВП $A(t)$ апроксимується випадковою функцією вигляду

$$A = A_0 \pm Vt, \tag{6}$$

де A_0 – випадкове початкове значення ВП (за $t = 0$), що має математичне сподівання (МС) $m_{A0} = M\{A_0\}$ та середнє квадратичне відхилення (СКВ) $S_{A0} = \sqrt{D_{A0}}$, $A_0 = A(t = 0)$; V – випадкова нормально розподілена швидкість зміни ВП у часі, що має МС $m_V = M\{V\}$ та СКВ $S_V = \sqrt{D_V}$, $V = \{\nu\}$.

Нелінійні випадкові функції. Для багатьох об'єктів є типовою деяка постійна відносна швидкість зміни ВП $\dfrac{dA(t)/dt}{A(t)} = V^*$, що відповідає нелінійному випадковому процесу $A(t)$, який апроксимується випадковою функцією вигляду

$$A = A_0 \exp(\pm V^* t), \tag{7}$$

де V^* – випадкова, нормально розподілена швидкість зміни натурального логарифма ВП $V^* = \dfrac{d \ln A}{dt}$, що має МС $m_{V*} = M\{V^*\}$ та СКВ $S_{V*} = \sqrt{D_{V*}}$.

У моделях обох класів (6) та (7) знаки «+» і «–» використовуються для апроксимації відповідно зростаючих й спадних у часі процесів. Випадкова величина X_0 у моделях (6) та (7) є постійною в часі, як і випадкова величина швидкості *V* зміни ВП у моделі (6). У моделі (7) постійною в часі є швидкість зміни *логарифма* ВП, сам же ВП має змінювану у часі швидкість зміни.

Для зручності подальшого розгляду моделей тільки в лінійному варіанті модель (7) лога-

рифмуванням перетворимо у лінійну модель зміни логарифма ВП:

$$\ln A(t) = \ln A_0 \pm V^* t. \qquad (8)$$

Якщо позначити натуральний логарифм ВП випадковою функцією $Y(t)$, тобто $Y(t) = \ln A(t)$; $Y_0 = \ln A_0$, то вираз (8) можна представити як

$$Y(t) = Y_0 \pm V^* t, \qquad (9)$$

подібний моделі (6). Тобто можна розглядати різні модифікації випадкових процесів тільки для $A(t)$, а результати аналізу використовувати сумісно, як для ВП $A(t)$, так і для $\ln A(t)$, тому що моделі для $A(t)$ та $\ln A(t)$ будуть подібні.

Розглянуті лінійні моделі зручні для апроксимації випадкових процесів зміни ВП тим, що дають можливість характеризувати ці процеси обмеженою кількістю аргументів моделі, для визначення яких потрібен мінімальний обсяг експериментальних даних.

Крім того отримані вирази (23), (24) та інші, з яких можна визначити час t, дозволяють знайти час збереження працездатності $t_{çá}$, і таким чином визначити критерій відмови робастно стійкої системи.

8. Висновки

1. Досягнення САК межі параметричної (робастної) стійкості можна розглядати як перехід системи в стан непрацездатності (в стан відмови). Час переходу САК з початку її функціонування до моменту втрати параметричної стійкості можна нормувати ***робастним критерієм її відмови***, який обмежує допустиме значення призначеного ресурсу САК при її проектуванні. Такий критерій також дозволить враховувати процеси фізичної деградації САК під час її експлуатації при нормуванні часу наступного підрегулювання (тривалості межремонтного періоду).

2. *Робастний критерій відмови* визначається для квазі стаціонарних і нестаціонарних систем згідно умов робастної стійкості стаціонарних систем [6], якщо використати замість фіксованого значення визначального параметра САК його функціональну залежність від часу.

Список літератури

1. Канарчук В. Є. Надійність машин: підруч. [для студ. вищ. навч. закл.] / В. Є. Канарчук, С. К. Полянський, М. М. Дмитрієв. – К.: Либідь, 2003. – 424 с.
2. Нечипоренко О. М. Основи надійності літальних апаратів: навчальний посібник [для студ. вищ. навч. закл.] / Олена Миколаївна Нечипоренко. – К.: НТУУ «КПІ», 2010. – 240 с.
3. Черкесов Г. Н. Надежность аппаратно-программных комплексов: учеб. пособие [для студ. высш. учебн. завед.] / Г. Н. Черкесов. – СПб.: Питер, 2005. – 479 с.
4. Половко А. М. Основы теории надежности / А. М. Половко, С. В. Гуров. – СПб.: БХВ-Петербург, 2006. – 704 с.
5. Надійність техніки. Моделі відмов. Основні положення: ДСТУ 3433-96. – [чинний від 1996--]. – К.: Держстандарт України, 1998. – 42 с. – (Національний стандарт України).
6. Ким Д. П. Теория автоматического управления. Т1. Линейные системы / Д. П. Ким. – М.: ФИЗМАТЛИТ, 2003. – 288 с.

УДК 519.854.2

ПАВЛОВ А.А.,
ЛИСЕЦКИЙ Т.Н.

НЕСТАЦИОНАРНЫЙ МЕТОД АНАЛИЗА ИЕРАРХИЙ В ЗАДАЧАХ ИЕРАРХИЧЕСКОГО ПЛАНИРОВАНИЯ И ПРИНЯТИИ РЕШЕНИЙ

В статье показано, что в предложенном в [1] гл. 9.5 дереве иерархий, предназначенного для выбора плана, которому соответствует минимальный риск существенной потери планируемой прибыли, веса вкладов критериев нижнего уровня иерархии в критерии следующего уровня по убыванию номера иерархии зависят от реализуемой альтернативы. Приводится модификация метода анализа иерархий для этого случая, предлагается информационная технология, реализующая модифицированный метод анализа иерархий в трех уровневой модели планирования и принятия решений.

The article describes, that in suggested in [1] hierarchical tree, used for selecting plan with minimal risk of significant loss of planning profit, weights of impact of low level criterion on the next level criterion on decreasing hierarchy number depends on selected alternative. The article gives a modification of Analytic Hierarchy Process Saaty for this case, it is proposed an information technology that implements the Modified Analytic Hierarchy Process in three-level model of planning and making decisions.

Введение

При решении задач планирования возникает необходимость в принятии решений по выбору оптимального плана выполнения работ с точки зрения большого количества критериев оптимальности. Для решения реальных задач многокритериального выбора в сложной обстановке с иерархическими структурами, включающими как осязаемые, так и неосязаемые факторы, широко применяется метод анализа иерархий (маи) саати. Но, так как в реальных задачах количество альтернатив может быть достаточно большим, применение классического маи будет не корректным. Такие ситуации возникают, например, когда множество альтернатив генерируется искусственно. В данной статье рассматривается модифицированный метод анализа иерархий (ммаи) [1] который может применяться на случай большого количества альтернатив.

Постановка задачи

Имеем следующую задачу принятия решений (Рис. 1), представленную в иерархической форме: m альтернатив $A_1 ... A_m$ и s уровней критериев $E_j^i, i = \overline{1, s},\ j = \overline{1, m_i}$.

Необходимо из достаточно большого множества m альтернатив $A_1 ... A_m$ выбрать такую альтернативу, для которой риск от существенного уменьшения планируемой прибыли будет минимальным. При этом считается, что планируемая прибыль от реализации любой рассматриваемой альтернативы будет допустимой. Так как число альтернатив достаточно большое, предполагается решение этой задачи модифицированным методом анализа иерархий [1]. Однако в общем виде дерево иерархий можно представить на рис. 1.

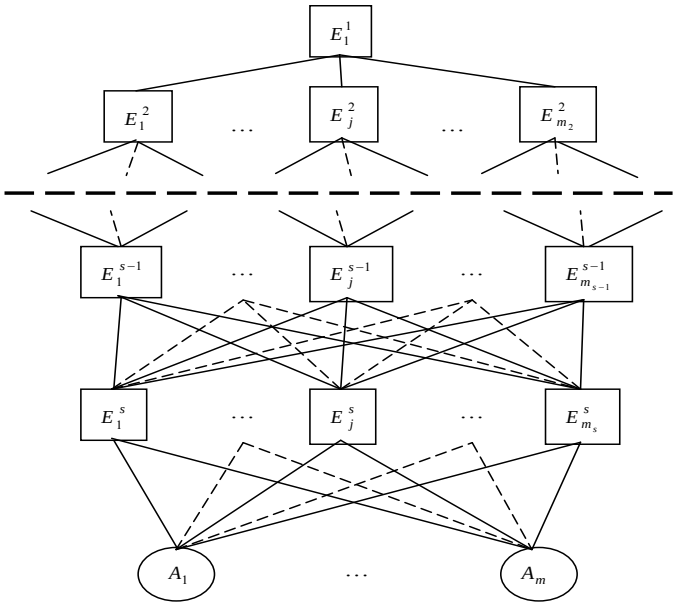


Рис. 1. Пример иерархического представления задачи принятия решений

В представленном дереве (рис. 1):

E_1^1 – глобальная цель – найти допустимый рабочий план с минимальным риском существенного уменьшения планируемой прибыли.

E_j^i – риск существенного уменьшения планируемой прибыли от влияния определенного

фактора, зависящего от конкретной задачи, $j=\overline{1,m_i}$, $i=\overline{1,s}$.

$A_1, \ldots, A_m$ – набор альтернатив.

Модификация метода анализа иерархий

Пусть $\omega_{E_j^i}^{E_e^{i+1}}$ – вклад (вес) критерия E_e^{i+1} в критерий E_j^i. Классическое использование метода анализа иерархий предполагает, что значения $\omega_{E_j^i}^{E_e^{i+1}}$ для всех j, e, t не зависят от альтернатив $A_1, \ldots, A_m$. Во многих случаях это не так [1] (п. 9.5).

Веса $\omega_{E_j^i}^{E_e^{i+1}}$ всех критериев вычисляются в соответствии с общепризнанными требованиями к методу анализа иерархий [2] по матрицам попарных сравнений.

На нижнем уровне иерархии $\gamma_{ij}, i, j=\overline{1,m}\,(i\neq j)$ – это элементы матриц парных сравнений. Они показывают во сколько раз степень риска существенного изменения прибыли у допустимого плана выполнения работ A_i выше, чем у допустимого плана выполнения работ A_j как следствие возможной реализации условий, определяющих критерий $E_l^s, l=\overline{1,m}$.

Рассмотрим промежуточный уровень иерархии, $j=\overline{1,s}$. Пусть найдены веса $\omega_{E_j^l}^{i}$, $i=\overline{1,m}$, $j=\overline{1,m_l}$ *l-го* уровня дерева иерархий. Тогда находится вклад (вес) критерия E_e^{l-j+1} $(e=\overline{1,m_{l-j+1}})$ в критерий E_t^{s-j} $(t=\overline{1,m_{l-j}})$ в предположении, что реализуется альтернатива A_i, т. е. находятся по матрицам парных сравнений веса $\omega_{E_t^{l-j}}^{E_e^{l-j+1}}(A_i)$. Соответствующие нормировки весов каждого набора являются одинаковыми. Таким образом, вес критерия вычисляется по следующей формуле:

$$E_t^{l-j}(A_i)=\sum_{e=1}^{m_{l-j+1}}\omega_{E_t^{l-j}}^{E_e^{l-j+1}}(A_i)E_e^{l-j+1}(A_i) \qquad (1)$$

Для верхнего уровня дерева иерархий по каждому набору весов $\left\{\omega_{E_t^1}^{E_e^2}(A_i),\forall e,t\right\}$, $i=\overline{1,m}$, находятся результирующие веса альтернатив $E_1^1(A_i)=\omega_i(A_i)=\sum_{e=1}^{m_2}\omega_{E_1^1}^{E_e^2}(A_i)E_e^2(A_i)$.

На основании формулы (1) можно утверждать, что критерий каждого уровня зависит от критерия следующего уровня по убыванию номера иерархии ($l-j$ уровень зависит от $l-j+1$). Таким образом, веса вкладов критериев нижнего уровня иерархии в критерии следующего уровня по убыванию номера иерархии зависят от реализуемой альтернативы.

Наилучшей альтернативе соответствует минимальный вес $\min\limits_i \omega(A_i)$, то есть, наименьший риск существенного уменьшения планируемой прибыли имеют те планы выполнения работ, у которых результирующий вес $E_1^1(A_i)$ является наименьшим.

ММАИ для четырехуровневой структуры

Очевидно, что каждой конкретной системе планирования соответствует свое собственное дерево иерархий. Рассмотрим инвариантное дерево иерархий (рис. 2), представленное в [1] п. 9.5.

E_1^1 – глобальная цель – найти допустимый рабочий план с минимальным риском существенного уменьшения планируемой прибыли.

E_1^2 – риск существенного уменьшения планируемой прибыли для плана в случае его успешного выполнения.

E_2^2 – риск существенного уменьшения планируемой прибыли в случае срыва заданных ограничений по срокам.

E_1^3 – риск существенного уменьшения планируемой прибыли в силу ухудшения рыночной конъюнктуры, как следствие возможного локального (глобального) финан сового кризиса.

E_2^3 – риск существенного уменьшения планируемой прибыли в силу возможного появления в планируемом периоде на рынке конкурентно эффективных альтернатив.

E_3^3 – риск существенного уменьшения планируемой прибыли в силу резкого увеличения затрат на выполнение допустимого плана, зависящих от возможной реализации ряда (определенных типом системы планирования) факторов, оценка которых не поддается детерминированному прогнозу.

E_4^3 – риск существенного уменьшения планируемой прибыли от возможного изменения в худшую сторону финансовой стабильности заказчика (заказчиков).

E_5^3 – риск существенного уменьшения планируемой прибыли от неправильной оценки

ресурсных, трудовых, наукоемких затрат для выполнения допустимого плана. Фактор является существенным, когда допустимый план включает в себя выпуск качественно новой продукции.

E_6^3 – риск существенного уменьшения планируемой прибыли от возможного несоответствия качества исходной продукции (полуфабрикатов, комплектующих изделий и т. д.) стандартам, влияющих на значение показателей качества выпускаемой продукции, что может привести к увеличению сроков выпуска результирующей продукции.

E_7^3 – риск существенного уменьшения планируемой прибыли от возможного срыва запланированных сроков поставщиками.

E_8^3 – риск существенного уменьшения планируемой прибыли от возможного ухудшения на плановом периоде функционирования системы ее финансового состояния, что может привести к срыву допустимых сроков плана выполнения работ.

Локальные вклады критериев $Å_1^3 \ldots Å_4^3$ в критерий $Å_2^2$ равны нулю.

Локальные вклады критериев $Å_5^3 \ldots Å_8^3$ в критерий $Å_1^2$ равны нулю.

$A_1, \ldots, A_m$ – набор альтернатив (планов).

Допустимые планы выполнения работ, реализуются на основе результатов прогноза рыночной конъюнктуры, сформированного портфеля заказов, и каждый из них может быть принят к реализации в плановом периоде.

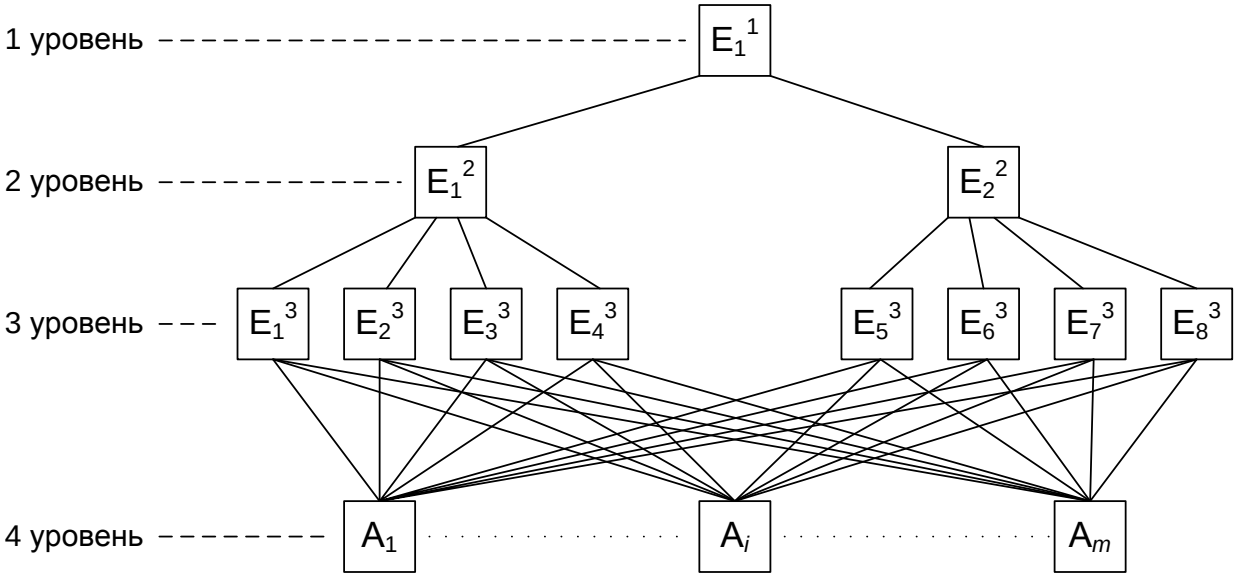


Рис. 2. Дерево иерархий

Элементы матриц парных сравнений на четвертом уровне иерархии $\gamma_{ij}, i, j = \overline{1,m}\,(i \neq j)$ интерпретируются следующим образом: во сколько раз степень риска существенного изменения прибыли у допустимого плана выполнения работ A_i выше, чем у допустимого плана выполнения работ A_j как следствие возможной реализации условий, определяющих критерий $E_e^3, e = \overline{1,8}$. По матрицам попарных сравнений вычисляются веса:

$E_e^3(A_i) = \omega_{E_e^3}^i$, $i = \overline{1,m}$, $e = \overline{1,8}$.

Элементы матриц парных сравнений на третьем уровне иерархий $\gamma_{ij}, i, j = \overline{1,4}\,(i \neq j)$ интерпретируются следующим образом: во сколько раз вклад критерия E_i^3 весомей вклада критерия E_j^3 в критерий E_1^2 (либо в критерий E_2^2) без учета требования о существенном уменьшении планируемой прибыли.

Найдем значение критерия E_1^2 и E_2^2 на альтернативе A_i:

$$E_1^2(A_i) = \sum_{l=1}^{4} \omega_{E_1^2}^{E_l^3}(A_i) E_l^3(A_i) = \sum_{l=1}^{4} \omega_{E_1^2}^{E_l^3}(A_i) \omega_{E_l^3}^{i};$$

$$E_2^2(A_i) = \sum_{l=5}^{8} \omega_{E_1^2}^{E_l^3}(A_i) E_l^3(A_i) = \sum_{l=5}^{8} \omega_{E_1^2}^{E_l^3}(A_i) \omega_{E_l^3}^{i}.$$

Нормированные локальные веса $\omega_1(A_i), \omega_2(A_i)$ при значениях $E_1^2(A_i)$ и $E_2^2(A_i)$ окончательно реализуют интегральный вес альтернативы A_i:

$$E_1^1(A_i) = \omega_1(A_i) E_1^2(A_i) + \omega_2(A_i) E_2^2(A_i) =$$

$$= \omega_1(A_i)\sum_{l=1}^{4} \omega^{E_l^3}_{E_1^2}(A_i)\omega^{i}_{E_l^3} + \omega_2(A_i)\sum_{l=5}^{8} \omega^{E_l^3}_{E_1^2}(A_i)\omega^{i}_{E_l^3},$$

$i = \overline{1,m}$, веса $\omega_1(A_i), \omega_2(A_i) \geq 0$ непосредственно определяются экспертным путем и интерпретируются как оценка возможности выполнения или невыполнения плана (альтернативы A_i). Можно лишь указать очевидную тенденцию: чем стабильнее финансовое состояние самой системы и экономики в целом, тем больше $\omega_1(A_i)$ по сравнению с $\omega_2(A_i)$.

Наименьший риск существенного уменьшения планируемой прибыли имеют те планы выполнения работ, у которых результирующий вес $E_1^1(A_i)$ является наименьшим.

Информационная технология, реализующая модифицированный метод анализа иерархий

Рассмотренный модифицированный метод анализа иерархий используется в иерархической системе планирования в блоке принятия решений, который состоит из двух этапов (рис. 3) [1]. На первом этапе по различным критериям оптимальности формируется целая серия возможных допустимых планов, отличающихся конкретным видом критерия, директивными сроками, весовыми коэффициентами, технологией реализации. На втором этапе в результате применения модифицированного метода анализа иерархий выбирается тот, который минимизирует риск существенного уменьшения планируемой прибыли и будет реализован в течение планового периода.

Данный алгоритм реализован в рамках отдельного класса MAHP (Modified Analytic Hierarchy Process), предоставляющего программный интерфейс для взаимодействия с ним, и находящегося в модуле принятия решений. Информационная система трехуровневой модели планирования и принятия решений вызывает этот модуль после второго уровня, когда получен согласованный план выполнения агрегированных работ. В соответствии с алгоритмом программное обеспечение строит отдельное дерево иерархий для каждой альтернативы.

Программное обеспечение, реализующее блок соответствующий модифицированному методу анализа иерархий, дает возможность экспертам задавать структуру дерева иерархий, указывать элементы матриц попарных сравнений всех альтернатив и критериев и, после вычислений результирующих весов для всех планов, предоставляет возможность выбора плана для реализации.

Процедура вычисления результирующих весов планов (альтернатив) начинается с определения весов альтернатив по отношению к всем критериям нижнего уровня. Они вычисляются на основании матриц попарного сравнения в соответствии с общепризнанными требованиями к методу анализа иерархий. Далее для каждого плана строится свое дерево иерархий. Вклад критериев каждого уровня в критерии следующего уровня (по убыванию номера иерархии) зависят от реализуемой альтернативы. Эта процедура повторяется для каждого уровня критериев, пока не будет получен результирующий вес планов. В результате будут получены оценки весов всех альтернатив.

Наилучшей альтернативе соответствует минимальный вес, то есть, наименьший риск существенного уменьшения планируемой прибыли имеют те планы выполнения работ, у которых результирующий вес является наименьшим.

Если план с наименьшим весом имеет максимальную прибыль, то он выбирается для выполнения. В других случаях, найденные веса для всех планов помогают лицу принимающему решение выбрать оптимальный план с точки зрения, как максимизации прибыли, так и минимизации возможности понести существенные финансовые потери.

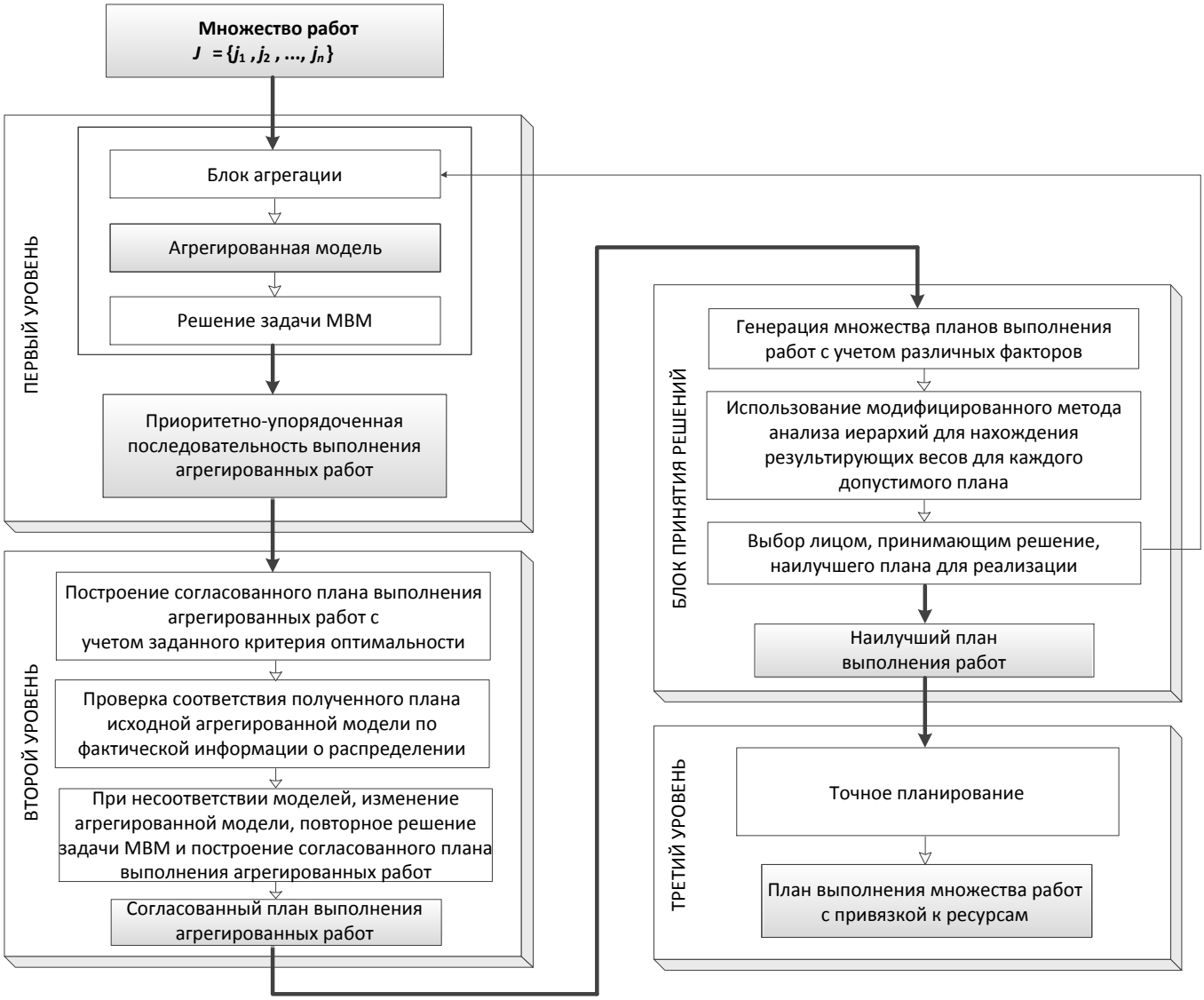


Рис. 3. Функциональная схема иерархической модели планирования и управления сложными системами

Выводы

На примере конкретного дерева иерархий в статье показано, что веса вкладов критериев нижнего уровня иерархии в критерии следующего уровня по убыванию номера иерархии зависят от реализуемой альтернативы.

Алгоритм модифицированного метода анализа иерархий построен таким образом, что для каждой альтернативы строится свое дерево иерархий Саати.

В результате выполнения алгоритма для каждой альтернативы (допустимого плана) вычисляется результирующий вес, отражающий риск существенного уменьшения прибыли при реализации соответствующей альтернативы.

Список литературы

1. Згуровский М.З., Павлов А.А. Принятие решений в сетевых системах с ограниченными ресурсами: Монография.– К.: Наукова думка, – 2010.– 573 с.
2. Саати Т. Принятие решений. Метод анализа иерархий: Tomas Saaty. The Analytic Hierarchy Process. – Пер. с англ. Р.Г. Вачнадзе. – М.: Радио и связь, 1993. – 315 с.

УДК 519.854.2

ПАВЛОВ А.А.

ПРИКЛАДНЫЕ АЛГОРИТМЫ ИДЕНТИФИКАЦИИ НЕЛИНЕЙНЫХ СИСТЕМ УПРАВЛЕНИЯ

В статье на основе алгоритмов фильтрации и восстановления истинной закономерности по ограниченному набору данных [1] приводятся прикладные алгоритмы идентификации [2] (определения) математической модели нелинейных систем управления в классе обычных нелинейных дифференциальных уравнений в форме Коши. Предполагается, что измерения данных, необходимых для идентификации, проводятся с аддитивной ошибкой, которая задается случайной величиной с нулевым математическим ожиданием, конечной дисперсией и произвольным распределением.

The article describes applied identification algorithms [2] (determination) of mathematical model of non-linear control system based on filtering algorithms and algorithms of restoration of valid regularity by limited data set [1] in class of regular linear differential equation in Cauchy form. It is supposed, that required for identification data changes are making with additive error, which sets by random variable with zero mathematical expectation, finite variance and arbitrary distribution.

Постановка задачи

Пусть динамическая система управления оценивается следующей моделью, представленной в векторной форме

$$\dot{\bar{X}}(t) = X(t, \bar{x}(t), \bar{u}(t)), \qquad (1)$$

где $\bar{u}(t)$ – известна вектор-функция от времени. Предполагается, что все условия существования, непрерывности и дифференцируемости решения системы (1) выполнены.

На отрезке $[t_0, t_k]$ измеряется с аддитивной помехой ε (случайная величина $M\varepsilon = 0$, $D\varepsilon = \sigma^2 < \infty$, распределение произвольно) либо $x_i(t)$ $i = \overline{1,n}$, либо $\frac{dx_i(t)}{dt}$.

По данным эксперимента необходимо восстановить функции:

$$X(t, \bar{x}(t), \bar{u}(t)) = \left(X_i(t, \bar{x}(t), \bar{u}(t)), i = \overline{1,n}\right)^T.$$

Дополнительной информацией является принадлежность функции $X_i(t, \bar{x}(t), \bar{u}(t))$ избыточному описанию:

$$X_i(t, \bar{x}(t), \bar{u}(t)) \in \sum_{j=1}^{L_i} a_{ij} \psi_j^i(t, \bar{x}(t), \bar{u}(t)), \qquad (2)$$

$i = \overline{1,n}$, где $\psi_j^i(t, \bar{x}(t), \bar{u}(t))$ – известная функция, a_{ij} – неизвестные коэффициенты, большинство из которых при фиксированном i равно нулю.

Алгоритм решения задачи

1) Для определенности считаем, что измеряется с ошибкой $x_i(t)$, т.е. $x_i(t) + \varepsilon$, $i = \overline{1,n}$, $M\varepsilon = 0$, $D\varepsilon = \sigma^2 < \infty$ с помощью очевидной модификации алгоритма, изложенного в [1] (глава 6, 6.1.1), находится оценка $\hat{x}_i(t)$ на отрезке $[t_0, t_k]$.

Примечание. Для удобства измерения на интервале $[t_0, t_k]$ можно масштабированием перенести на любой заданный отрезок, например, $[1,2]$ для того, чтобы аппроксимирующий полином восстанавливался необходимой степени.

2) Численными методами с усреднением по $\hat{x}_i(t)$, $i = \overline{1,n}$ восстанавливается $\frac{d\hat{x}_i(t)}{dt}$, $i = \overline{1,n}$ для моментов времени $t \in \overline{t_0, t_k}$.

3) По имеющейся информации $\left.\frac{d\hat{x}_i(t)}{dt}\right|_{t=\overline{t_0,t_k}} = X_i(t_k, \bar{x}(t_k), \bar{u}(t_k))\Big|_{t=\overline{t_0,t_k}}$, $i = \overline{1,n}$ методом, изложенным в [1] (глава 6, 6.3.4) находится истинное значение функций $X_i(t, \bar{x}(t), \bar{u}(t))$ в соответствии с введенным в [1] (глава 6, 6.3) критерием истинности о минимальном описании истинной законномерности (минимум ненулевых коэффициентов a_{ij} при фиксированном индексе i).

Примечания

1) Очевидным образом алгоритм модифицируется для случая, когда измеряется $\frac{dx_i(t)}{dt} + \varepsilon$ для любого $t \in \overline{t_0, t_k}$.

2) Очевидным образом алгоритм модифицируется, когда объект управления описывается в виде:

$$\frac{d^n \tilde{x}(t)}{dt^n} = X\left(t, \frac{dx^{n-i}(t)}{dt^n}, i = \overline{1, n}, u(t)\right).$$

3) Теоретически обосновать, что приведенный алгоритм всегда решает задачу идентификации очевидно нельзя. Однако в силу ограничений (2) (принадлежность истинного описания к избыточному классу) вероятность точного решения достаточна велика. А если описание модели найдено правильно, то установить этот факт с помощью дополнительных экспериментов не вызывает затруднений.

Список литературы

1. Згуровский М.З., Павлов А.А. Принятие решений в сетевых системах с ограниченными ресурсами: Монография.– К.: Наукова думка, – 2010.– 573 с.
2. Справочник по теории автоматического управления./ Под редакцией А. А. Красовского. – М.: Наука. Гл. ред. физ.-мат. 1987, 712 стр.

УДК 004.94

ПЛАКСА О.С.

ФОРМУВАННЯ ПЛАНУ ДІЯЛЬНОСТІ ОРГАНІВ ДКРС

В статті наведені критерії відбору об'єктів контролю, визначені критерії корисності, наведено алгоритм формування плану діяльності органу ДКРС, розглянута задача визначення залежності тривалості ревізії від суми фінансування, яка виділялася об'єкту контролю, представлена математична модель побудови плану робіт для органів ДКРС з максимізацією корисності плану.

The article contains the criteria for selection of control objects and criteria of usefulness. The author describes an algorithm for forming the annual activity plan for SCRS. He solves the problem of determination of the time dependence for the controlling activity from the amounts of funding and gives a mathematical model for the developing work plan for the SCRS with maximizing usefulness of the plan.

Вступ

Сучасні системи підтримки прийняття рішення (СППР) є засобами вирішення задач керування і допомагають особам, що приймають рішення (ОПР). За допомогою СППР може виконуватися вибір рішень деяких неструктурованих або слабоструктурованих задач. Такі системи є результатом міждисциплінарних досліджень комп'ютерних наук [1].

Аналізуючи розвиток напрямку прикладного програмного забезпечення зрозуміло, що область СППР буде залишатись однією з найважливіших для бізнесу, адже вони покликані зробити процес прийняття рішення більш свідомим та поінформованим.

Так, ефективність вирішення задач планування діяльності робіт органів ДКРС та аналізу результатів проведення контрольних заходів залежить від послідовності подій, що виникають уже після ухвалення рішення. Тому системи підтримки прийняття рішень реалізують функцію прогнозу. Проблема полягає в тому, щоб отриманий прогноз як найбільше задовольняв потребам ОПР.

Вирішення цієї задачі іноді потребує обробки даних великої розмірності, а також проведення обчислень на їх базі. Крім того, системи для забезпечення наукових та інженерних розрахунків характеризуються високою індивідуалізацією. Це спричинено тим, що створюються вони невеликими дослідницькими групами, що займаються вузькоспеціальними задачами. Здобуті ними досягнення рідко стають відомі широкому колу фахівців, ще рідше можна говорити про повторне використання створеного програмного забезпечення. В результаті актуальними проблемами є достовірність отриманих результатів, а також ефективність роботи СППР.

Частково вирішення зазначених проблем досягається за рахунок використання декількох методів, повторного «прогону» певного алгоритму з різними вхідними параметрами.

Оцінка сучасного стану проблеми

Специфіка СППР насамперед виявляється під час порівняння цілей різних видів ІС. Традиційні інформаційно-звітні системи узагальнюють і регулярно надають поточну регламентовану інформацію про основні функції ділової діяльності (маркетинг, виробництво, фінанси). Отримання звітів на робочих місцях менеджерів відбувається за графіком або за запитом [1].

СППР створюються для неструктурованих та напівструктурованих проблем. Генеруючи нетипові специфічні рішення, менеджери самі формують інформацію в інтерактивному режимі. Для планування і контролю на тактичному і стратегічному рівні менеджерам потрібна додаткова, унікальна, разова інформація [2].

Порівняння технології формування інформації в традиційних звітних ІС та в СППР розкриває основну особливість СППР. Регламентовані звіти, які менеджери отримують готовими від ІС, сформовані на основі чітко визначеної технології, описаної в проектній документації до ІС і контрольованої її інженерно-технічним персоналом.

Формування інформації засобами СППР також передбачає використання певних технологій .

Технологія підтримки прийняття рішень не виконується повністю автоматично, оскільки здійснюється під управлінням менеджера.

СППР – це така людино-машинна система, де процеси формування і використання інформації не розділяються [3,4].

Основними компонентами СППР є:

- база даних;
- підсистема управління базою даних;
- інтерфейс користувача;
- база моделей;
- система управління базою моделей.

Характеристики та переваги сучасних СППР:

- надає особі, яка приймає рішення (ОПР), допомогу в процесі прийняття рішення і забезпечує підтримку в усьому діапазоні структурованих, напівструктурованих і неструктурованих задач;
- не замінює і не скасовує судження та оцінки ОПР, а лише підтримує їх;
- підвищує ефективність генерування альтернативних рішень;
- здійснює інтеграцію моделей та аналітичних методів зі стандартним доступом до них;
- проста в роботі і придатна для використання менеджерами, які не мають значного досвіду роботи з ЕОМ;
- побудована за принципом інтерактивного розв'язання задач;
- орієнтована на гнучкість та адаптивність у пристосуванні до змін середовища або підходів до розв'язання задач;
- не нав'язує певний процес прийняття рішення. Користувач має можливість вибору альтернатив, використовуючи їх відповідно до свого пізнавального стилю. СППР класифікуються за рівнем, призначенням, галуззю та функціональною приналежністю [4].

Ринок СППР

На ринку СППР компанії пропонують наступні види послуг зі створення систем підтримки прийняття рішень:

- реалізація пілот-проектів по СППР-системам, з метою демонстрації керівництву замовника якісного потенціалу аналітичних додатків;
- створення спільно з замовником повнофункціональних СППР, включаючи сховище даних та засоби Business Intelligence;
- проектування архітектури сховища даних, включаючи структури зберігання та процеси керування;
- створення «вітрин даних» для виділеної предметної області;
- встановлення і налагодження засобів OLAP і Business Intelligence, їх адаптація до вимог замовника;
- аналіз інструментів статистичного аналізу і «видобутку даних» для вибору програмних продуктів під архітектуру і потреби замовника;
- інтеграція систем СППР в корпоративні інтранет-мережі замовника, автоматизація електронного обміну аналітичними документами між користувачами сховища;
- розробка Інформаційних Систем Керівника (EIS) під необхідну функціональність;
- послуги з інтеграції баз даних в єдине середовище зберігання інформації
- навчання фахівців замовника технологіям сховищ даних та аналітичних систем, а також роботі з необхідними програмними продуктами;
- надання консалтингових послуг замовнику на всіх стадіях проектування і експлуатації сховищ даних та аналітичних систем;
- комплексні проекти створення чи модернізації обчислювальної інфраструктури, що забезпечує функціонування СППР: рішення будь-якого масштабу, від локальних систем до систем масштабу підприємства / концерну / галузі [5].

Формулювання цілей статті

Метою данного дослідження є підвищення ефективності та якості інформаційної технології інтелектуальної обробки даних у системах підтримки прийняття рішень для органів ДКРС. Відповідно до зазначеної мети поставлено таку задачу – розробити методику розв'язання задачі формування плану діяльності органів ДКРС:

- визначити множину критеріїв, які впливають на вибір об'єктів контролю, що мають входити до плану;
- розробити алгоритм визначення тривалості ревізії;

– розробити математичну модель формування плану.

Розділи плану основних напрямків діяльності

Метою планування контрольних заходів органами Державної контрольно-ревізійної служби є [6]:

– забезпечення оптимальної концентрації трудових, фінансових та матеріальних ресурсів суб'єктів державного фінансового контролю при реалізації визначених напрямів контролю;
– усунення паралелізму та дублювання в роботі суб'єктів державного фінансового контролю шляхом забезпечення належного виконання вимог законодавства щодо державної регуляторної політики.

План основних напрямків (ПОН) має чотири розділи:

– контроль за використанням коштів державного бюджету, станом збереженням державного майна у міністерствах, інших центральних органах виконавчої влади та на підприємствах, в установах і організаціях;
– контроль за виконанням бюджетів у регіонах;
– тематичні перевірки з питань дотримання фінансово-бюджетної дисципліни;
– стан контрольно-ревізійної роботи у міністерствах, інших центральних органах виконавчої влади.

По кожному пункту ПОН повинен включати:

– тему контрольного заходу;
– назву суб'єкта господарювання (об'єкту контролю), який підлягає контролю, та місце його розташування;
– термін закінчення контрольного заходу;
– період, який підлягає контролю;
– перелік суб'єктів державного фінансового контролю, які будуть спільно проводити контрольні заходи (за їх погодженням).

Критерії відбору об'єктів

Першим етапом при створенні ПОНу є відбір об'єктів контролю відповідно до таких параметрів:

– адміністративно-територіальна одиниця (АТО), до якої належить об'єкт контролю – дозволяє виконати перевірку за виконанням бюджетів у регіонах;
– найменування органу влади, який є власником (органом управління) об'єкту контролю – дозволяє здійснити фінансову перевірку діяльності окремого органу влади;
– організаційно-правова форма управління об'єкту контролю;
– форма власності об'єкту контролю;
– виділена з державного бюджету сума фінансування;
– код програмної класифікації (КПК) виділеної суми;
– код функціональної класифікації (КФК) виділеної суми.

Визначення множини цих показників дозволяє звузити сферу проведення майбутніх контрольних заходів.

Вагові критерії

Для визначення доцільності включення об'єкту контролю до ПОНу має бути розрахована **корисність** об'єкту. Для цього експертом визначається вага кожного з наступних критеріїв:

– доручення Президента України, в т. ч. що містяться у виданих ним актах;
– звернення або запит народних депутатів;
– доручення Кабінету Міністрів України, в т. ч. що містяться у виданих ним актах;
– доручення Міністра фінансів України, в т. ч. що містяться в наказах Мінфіну;
– доручення правоохоронних органів;
– звернення міністерства, іншого центрального органу виконавчої влади, контролюючого органу;
– звернення громадян;
– звернення підприємства, установи чи організації;
– прохання іншого регіонального КРУ чи територіального КРВ;
– доручення Верховної ради України, в т. ч. що містяться в Постановах;
– звернення місцевого органу виконавчої влади;
– звернення місцевого органу самоврядування;
– звернення або запитом депутатів місцевих рад;

- доручення Координаційного комітету по боротьбі з корупцією і організованою злочинністю при Президентові України;
- доручення Ради національної безпеки і оборони України;
- доручення судових органів;
- звернення Координаційного комітету при місцевому органі виконавчої влади;
- доручення органів ДПА;
- час з проведення попереднього контрольного заходу;
- зміна керівництва;
- зміна головного бухгалтера;
- наявність кредитів, виданих під гарантію уряду;
- наявність державного майна;
- обсяг виділених коштів державного бюджету;
- обсяг виділених коштів місцевих бюджетів;
- ризик не включення об'єкту до ПОНу.

Алгоритм формування плану діяльності

Для формування плану діяльності органу ДКРС необхідно виконати наступні етапи:

Етап 1. Визначення ваг (цінностей) критеріїв відбору $k_1, k_2,\ldots, k_n$ та стратегічних напрямків діяльності $K_1, K_2,\ldots, K_7$ експертом.

Етап 2. Відбір підконтрольних об'єктів $O_j \in O$, які відповідають визначеним стратегічним напрямкам $\sum_{m=1}^{7} O_j(K_m) \geq 1$.

Етап 3. Розрахунок тривалості проведення ревізії t_j для виділених підконтрольних об'єктів. Для цього визначити кластер, до якого належить об'єкт та використовуючи відповідну залежність отримати значення t_j [7].

Етап 4. Розв'язання багатокритеріальної задачі для визначення «цінності» f_j проведення контрольного заходу у підконтрольному об'єкту з використанням визначених ваг $k_1, k_2,\ldots, k_n$, їх значень по кожному об'єкту $k_{1j}, k_{2j},\ldots, k_{nj}$, та інтегральної оцінки $k_{0j} = \sum_{m=1}^{7} O_j(K_m)$ [8].

Етап 5. Розв'язання задачі формування плану діяльності з отриманими f_j та t_j.

Аналіз даних

Тривалість проведення ревізії на підконтрольному об'єкті залежить від різних бюджетних сум, цільове використання яких перевіряється підчас контрольного заходу. Тоді задача прогнозу тривалості ревізії може бути зведена до розв'язання задачі регресійного аналізу, де в якості цільової змінної буде виступати тривалість ревізії.

Таким чином для кожного підконтрольного об'єкту відома виділена сума фінансування, що має такі складові:

- фінансування на утримання X_1;
- виділення грошей X_2;
- акумулювання пільг оподаткування X_3;
- кредити, отримані під гарантію уряду X_4;
- акумулювання інших державних цільових грошей X_5;
- гроші державних цільових фондів X_6;
- позабюджетні гроші X_7.

Нехай t – тривалість проведення ревізії, що обчислюється як кількість людино-днів, витрачених на її проведення.

Необхідно визначити залежність t від векторної змінної X, де $X = \{ X_1, X_2, X_3, X_4, X_5, X_6, X_7 \}$, тобто визначити коефіцієнти полінома Колмогорова-Габора [7]:

$$t = f(X_1, X_2, ..., X_7) = a_0 + \sum_{i=1}^{7} a_i X_i + \sum_{i=1}^{7}\sum_{j=i}^{7} a_{ij} X_i X_j + \sum_{i=1}^{7}\sum_{j=i}^{7}\sum_{k=j}^{7} a_{ijk} X_i X_j X_k + \ldots \quad (1)$$

Розподіл трудових ресурсів органів ДКРС

Побудуємо математичну модель побудови плану робіт для органів ДКРС таким чином, щоб корисність плану була максимальною:

$$\sum_{i=1}^{n} f_i x_i \to \max \quad (2)$$

$$\sum_{i=1}^{n} t_i x_i \leq T \quad (3)$$

$$x_i = \{0,1\} \quad (4)$$

x_i може приймати значення 1 – у разі, якщо підприємство включено до плану і 0, якщо – не включено;

f_i – коефіцієнт корисності включення і-го підприємства до плану;

t_i – час проведення контрольного заходу на і-тому підприємстві (чол/дн) ;

T – загальний час планування (чол/дн).

Задачу (2) – (4) доцільно розв'язувати за допомогою методу динамічного програмування [9].

Висновки

Визначені критерії, що дозволяють провести початкове розбиття множини підконтрольних об'єктів відповідно до стратегічних планів розвитку. Розроблено алгоритм формування плану діяльності органів ДКРС.

Показано, що задачу формування плану діяльності органів ДКРС можна звести до задачі про одновимірне пакування, що може бути вирішена за допомогою методу динамічного програмування.

Список літератури

1. Батюк А.Є. Інформаційні системи в менеджменті. Навч. пос. //– Львів: НУ "Львівська політехніка", 2004. – 520 с.
2. Кречетов Н.А. Продукты для интеллектуального анализа данных. // – М.: Рынок программных средств, № 14–15, 1997. – С. 32–39.
3. Иоффин А.И. Системы поддержки принятия решений // Мир ПК. – 1993. – №5. – С. 47–57.
4. Boulding K. E. General Systems Theory // – Management Science, 2, 2000. – 230 с.
5. Гужва В. М. Інформаційні системи і технології на підприємствах: Навч. пос. // – К.: КНЕУ, 2001. – 400 с.
6. Офіційний вісник України // Режим доступу: http://www.gdo.kiev.ua/
7. Богушевская Н.В., Плакса А.С., Ягодкина Е.В. Использование интеллектуального анализа данных для прогнозирования длительности ревизии для органов ГКРС // Адаптивні системи автоматичного управління. Межвідомчий науково-технічний збірник. – Дніпропетровськ: Системні технології, – 2009. – Вип. 15(35). – С.10 – 15.
8. Рыков А.А. Модели и методы многокритериальной оценки качества и выбора решений при риске // – М.: Открытое образование, 2006 – 164 с.
9. Левитин А.В. Метод грубой силы: Задача о рюкзаке //– М.: «Вильямс», 2006. – С. 160–163.

УДК 621.313

ПОЛТОРАК В.П.,
ВІТІЩЕНКО Н.С.

ВПЛИВ ОСНОВИ КОДУ НА ЕФЕКТИВНІСТЬ НАДЛИШКОВОГО КОДУВАННЯ ДАНИХ

Досліджений вплив основи надлишкових кодів на ефективність кодування у системах передавання даних. Обґрунтована верхня межа для основи коду, за якої досягається максимальна ефективність кодування. Обґрунтовано перспективи дослідження та використання кодів з короткою довжиною.

A code base impact on encoding efficiency was investigated for data transmission systems. Upper edge for code base is validated for encoding efficiency maximization. A short codes usage was grounded.

Відомості про навколишній світ передаються різноманітними каналами у формі дискретних символів – даних. Під час передавання у певному середовищі (металеве, оптичне, радіоефір, тощо) сигнали, що переносять дані, можуть бути спотворені завадами різної природи. Відповідно, дані на приймальному боці можуть не співпадати із тими, що були передані – виникають помилки. Виправлення помилок виконують за допомогою надлишкових (завадостійких) кодів.

Природна надлишковість є невід'ємною властивістю висхідних повідомлень (текстів, файлів, тощо). Будемо вважати, що джерело повідомлень містить лише безнадлишкові дані, як результат кодування висхідних повідомлень оптимальним нерівномірним кодом (ОНК), що до певної міри знімає їх природну надлишковість. Саме такі дані є незахищеними від спотворень у каналі передавання.

Захист їх надлишковим кодом передбачає цілеспрямоване, «дозоване» введення (додавання) надлишковості до повідомлень від джерела: наприклад, рівно на стільки, скільки потрібно для «виявлення до $\mathbf{t_в}$ можливих помилок», або ж – «виправлення до $\mathbf{t_{вп}}$ можливих помилок», тощо, що припадає на певну кількість **n** символів кодованих повідомлень. Звичайно, таке надлишкове кодування має сенс у разі, коли «дозована», обґрунтована кодом надлишковість менша за природну надлишковість висхідних повідомлень. У даній статті предмет дослідження обмежимо блоковими (**n, k**)-кодами, де **n** – довжина закодованого блоку даних, **k** – кількість інформаційних символів від джерела, які треба захистити від спотворень. Надлишковість зазвичай виступає у формі певної кількості **r** надлишкових символів коду, причому: $\mathbf{n = k + r}$.

Мірою втрат за умови надлишкового кодування (надалі – коду, кодування) є відносна надлишковість (надалі - надлишковість) $\mathbf{D = r/n}$. Її віддзеркаленням є відносна швидкість коду $\mathbf{R = k/n}$ (надалі – швидкість коду) – своєрідний технічний ефект від кодування. Вочевидь, що $\mathbf{D + R = 1}$, а самі **D** і **R** мають смисл питомих величин.

Мірою технічного ефекту, що досягається є, також, кількість помилок $\mathbf{t_в}$, або $\mathbf{t_{вп}}$. Доцільніше обрати за міру технічного ефекту питому величину, наприклад, $\mathbf{x = t_{вп}/n}$.

Поєднавши дві, визначені таким чином, характеристики, **R** і **x** оберемо відомий критерій оцінки ефективності кодів $\mathbf{R = f(x)}$ – межу для мінімальної кодової відстані, для оцінки ефективності кодів з різним значенням основи **q** , або – потужності алфавіту коду [1, 2]. Зазвичай, межу для мінімальної кодової відстані визначають при великих значеннях **n**, а сам критерій задається параметрично, через параметри кодів. Його зображують графічно на площині (**R;x**) і він є геометричним місцем точок **K(R;x)** з відповідними координатами. Вигляд **x** можна трохи модифікувати, врахувавши значення мінімальної кодової відстані **d** за Хемінгом: $\mathbf{d \geq 2\, t_{вп} + 1}$, до форми $\mathbf{x = d/2n}$, за умови, що **n** – велике. Різниця між дозволеними кодовими комбінаціями (надалі – КК) визначається мінімальною кодовою відстанню **d** коду, при тому, що вся множина КК розбита на дві підмножини – дозволених та заборонених комбінацій однієї довжини **n** [2].

Коди мають велику кількість параметрів, що їх характеризують, тому критерій ефективності кодів можна сформулювати одним із трьох способів [1]:

1. серед кодів з однаковими **n** і **d** кращим є код, який має більшу **k (**за однакових $\mathbf{x = d/2n)}$;

2. серед кодів з однаковими **n** і **k** кращим є код, який має більшу **d (**за однакових **R);**

3. серед кодів з однаковими **d** і **k** кращим є код, який має меншу **n (**менше **r).**

Ці критерії органічно відповідають аналізу ефективності кодів на площині **R = f(x).**

В даній статті проведемо дослідження впливу основи коду **q** на ефективність надлишкового кодування даних, при $2 \le \mathbf{q} < \infty$ за критерієм **(R;x).**

На рис.1 зображена класична структура системи передачі даних, що складається з джерела повідомлень, двох відносно самостійних пристроїв: Кодеку та Модему, каналу зв'язку та отримувача повідомлень. В рамках даної статті обговорюємо вхід Кодера та вихід Декодера.

За великого обсягу алфавіту сигналів, кожен

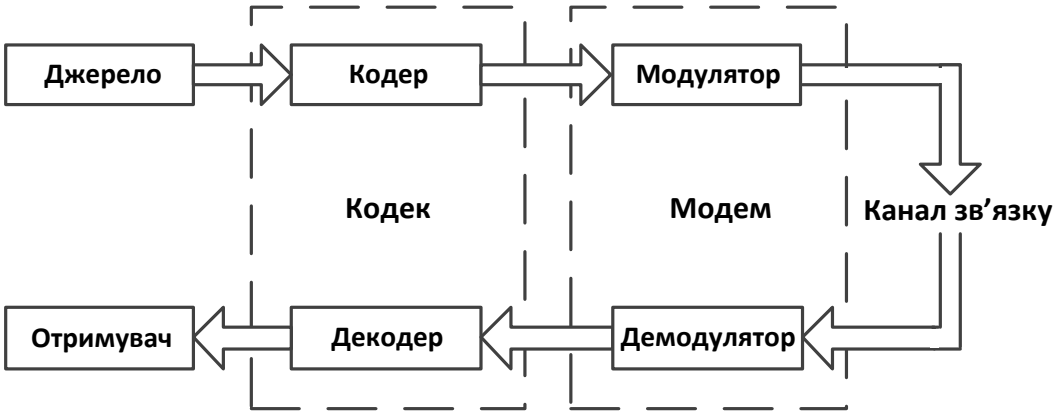


Рис. 1. Модель системи передачі даних

символ може переносити набагато більше інформації ніж за **q = 2**. Зокрема, алфавіт, що містить **q** символів дозволяє передавати $\mathbf{I=\log_2 q}$ двійкових одиниць інформації на кожен символ.

При фіксованих значеннях **n** та **k**, можна отримати верхню та нижню межі для найбільшої **d**. Розглянемо ці межі.

Відповідно до

$$\frac{t_в}{n} = \frac{d-1}{2n} \approx \frac{d}{2n}, \qquad (1)$$

найкращі коди мають велику коректувальну здатність $\mathbf{t_в}$. Ці граничні залежності і є межами (верхніми або нижніми) для **d**.

Верхня межа Плоткіна є точною для низькошвидкісних кодів. У [1,2] вона записана у вигляді:

$$k \le n - \frac{qd-1}{q-1} + 1 - \log_q d. \qquad (2)$$

Поділивши (2) на n та спрямувавши n до $n \to \infty$, отримаємо нерівність [3]:

$$R \le 1 - \frac{2q}{q-1}\frac{d}{2n}, \qquad (3)$$

в якій відкинуто доданки вищих ступенів малості.

Верхні межі Хемінга і Елайеса, а також нижня межа Варшамова – Гілберта пов'язані з нерівністю Чернова [1], де використовується функція вигляду

$$\varphi(x) = x\log_q(q-1) - x\log_q x - (1-x)\log_q(1-x). \qquad (4)$$

Верхня межа Хемінга визначає максимальну можливу кількість дозволених кодових комбінацій $\mathbf{q^k}$ будь-якого коду при заданих значеннях **n** та **d.** В [2] межа Хемінга має наступний вигляд:

$$R \le 1 - \varphi\left(\frac{t_в}{n}\right). \qquad (5)$$

Врахувавши (1) та $n \to \infty$, дістанемо

$$R \le 1 - \varphi\left(\frac{d}{2n}\right) = 1 - \varphi(x), \qquad (6)$$

де $x = d/2n$.

Включивши до (6) вираз (4), матимемо рівняння межі Хемінга:

$$R = 1 - x\log_q(q-1) - x\log_q x - (1-x)\log_q(1-x). \qquad (7)$$

Область значень для **x** визначається виразом (7), з якого випливає межа Хемінга. Якщо збільшувати **q**, то лінія межі Хемінга піднімається вище над початковою межею, при **q = 2**, за рахунок збільшення **R**. При $q = \infty$, отримаємо асимптотичний вираз межі при великих значеннях основи **q**, тобто

$$R \le 1 - x. \qquad (8)$$

Верхня межа Елайеса, для середньо швидкісних кодів з **q** символами, визначається виразом [2]

$$\frac{d}{n} < \delta(R)\left[2 - \frac{\delta(R)q}{q-1}\right] + \varepsilon, \qquad (9)$$

де $\delta(R)$ – розв'язок рівняння $\varphi(x) = 1 - R$, утвореного з (5), а $\varphi(x)$ – функція (4); $\varepsilon > 0$ – як завгодно мала складова.

За допомогою межі Хемінга (7) та позначивши $x = \delta(R)$ дістаємо парні значення $x = \delta(R)$ та **R**. Після цього для кожного **R** за відомим $x = \delta(R)$ розраховуємо значення $d/2n$ межі Елайеса, користуючись виразом (9), перетвореним до вигляду

$$\frac{d}{2n} < \frac{1}{2}\delta(R)\left[2 - \frac{q}{q-1}\delta(R)\right] + \varepsilon . \quad (10)$$

Відповідно виразу (10), будуємо лінії для межі Елайеса при $2 < q < \infty$.

Нижня межа Варшамова-Гілберта – для великих значень **n** встановлює нижню межу для числа **r** перевірних розрядів, необхідного для забезпечення заданої **d**. Нижня межа оцінює число **r** при заданих **k** та **d**.

Межа Варшамова-Гілберта в [2] визначається виразом

$$q^{n-k} \le \sum_{i=0}^{d-2} C_n^i (q-1)^i , \quad (11)$$

де C_n^i – біномні коефіцієнти. Виключимо їх із розгляду, скориставшись асимптотичною оцінкою цих коефіцієнтів у вигляді нерівності Чернова

$$\sum_{i=0}^{t_в} C_n^i (q-1)^i \le q^{n\varphi(x)}, \quad (12)$$

де $\varphi(x)$ – функція виду (4).

З урахуванням (12), прологарифмуємо обидві частини нерівності (11) і знайдемо

$$n - k \le n\varphi(\frac{d-2}{n}) ,$$

звідки $R = \frac{k}{n} \ge 1 - \varphi(\frac{d}{n} - \frac{2}{n})$.

Спрямувавши **n** до $n \to \infty$, дістаємо асимптотичну форму нижньої межі Варшамова – Гілберта

$$R \ge 1 - \varphi(\frac{d}{n}) . \quad (13)$$

Порівнявши (13) з межею Хемінга (6), можна остаточно записати перетворений вираз (7) для межі Варшамова – Гілберта

$$R \ge 1 + \log_q \left[(\frac{2x}{q-1})^{2x} (1-2x)^{(1-2x)} \right] . \quad (14)$$

Завдяки цим межам, можна визначити області можливих значень параметрів надлишкових кодів та отримати конкретні числові значення для кожного з досліджуваних кодів при різних значеннях $\mathbf{t_в}$.

Оцінимо вплив **q** на ефективність ряду відомих надлишкових кодів: Хемінга, БЧХ та Ріда–Соломона, використавши отримані вирази критеріїв для мінімальної кодової відстані.

Код **Хемінга** має мінімальну кодову відстань **d** = 3 та здатен виправити поодинокі помилки в межах переданого блоку [1]. Мінімальне співвідношення коректувальних та інформаційних розрядів, нижче якого код не зберігає свої коректувальні властивості, визначається як $\mathbf{q^r - 1 = n}$ [1, 3].

Для порівняльного аналізу, з основних параметрів коду Хемінга були обрані: основа коду **q**, **r**, **n** та розраховані **k**, **x**, **R** та **D.**

Нижче наведено приклад одного з варіантів розрахунку параметрів для коду Хемінга з великим **n** при **q** = 4:

Оберемо **r** = 6, тоді:

- $n = \frac{q^r - 1}{q - 1} = \frac{4^6 - 1}{4 - 1} = 1365$
- $k = n - r = 1365 - 6 = 1359$
- $x = \frac{tв}{n} = \frac{1}{1365} = 0.0007$
- $R = \frac{k}{n} = 0.996$
- $D = \frac{r}{n} = \frac{6}{1365} 100\ \% = 0.43\ \%$

У Таблиці 1 наведено параметри коду Хемінга, розраховані для різних **q**: 4, 8, 16, 32, 64, з метою побачити, як впливає зміна **q** на ефективність коду.

Табл.1. Параметри кодів Хемінга

$\mathbf{t_в = 1}$					
q	**4**	**8**	**16**	**32**	**64**
n	1365	4681	4369	1057	4161
x	0.0007	0.00021	0.00023	0.0009	0.00024
R	0.996	0.9989	0.9991	0.997	0.9992

Недвійкові коди **БЧХ** будуються за допомогою твірних поліномів P(x), які визначаються за заданими $\mathbf{d_{min}}$ та **n**. Найпоширенішим кодом БЧХ є код, для побудови якого застосовуються поле елементів GF(q) та розширене поле локаторів $GF(q^m)$. При побудові кодів БЧХ завжди забезпечується мінімально досяжна надмірність $r = 2mt_в$.

Для порівняльного аналізу, з основних параметрів коду БЧХ були обрані наступні: **q**, $\mathbf{t_в}$, **m**, $\mathbf{n|(q^m-1)}$, **x**, **r** , $k = n - r$, **D** та **R.**

Нижче наведено приклад одного з варіантів розрахунку параметрів для коду БЧХ, з виправленням 1 помилки при **q = 4**. Оберемо, наприклад, **m =** 5, тоді:

- $n : q^m - 1 = q^5 - 1 = 1023$
- $x = \frac{tв}{n} = \frac{1}{1023} = 0.0009$
- $r = 2mt_в = 10$
- $k = n - r = 1023 - 10 = 1013$
- $R = \frac{k}{n} = 0.99$
- $D = \frac{r}{n} = \frac{10}{1023} = 0.97\,\%$

У Таблиці 2 наведено параметри коду БЧХ, розраховані при різних **q**: 4, 8, 16, 32, 64 та кількості виправлених помилок $\mathbf{t_в}$= 1, 2 та 3, з метою побачити, як впливає зміна **q** на ефективність коду.

Табл.2. Параметри коду БЧХ

$\mathbf{t_в}$ = 1					
q	**4**	**8**	**16**	**32**	**64**
n	1023	4095	4095	1023	4095
x	0.0009	0.0002	0.0002	0.0009	0.00024
R	0.99	0.998	0.998	0.996	0.999
$\mathbf{t_в}$ = 2					
q	**4**	**8**	**16**	**32**	**64**
n	1023	4095	4095	1023	4095
x	0.0019	0.0005	0.0005	0.0019	0.0005
R	0.98	0.996	0.997	0.992	0.998
$\mathbf{t_в}$ = 3					
q	**4**	**8**	**16**	**32**	**64**
n	1023	4095	4095	1023	4095
x	0.0029	0.0007	0.0007	0.0029	0.0007
R	0.971	0.994	0.996	0.988	0.997

Коди **Ріда-Соломона** використовуються для передачі інформації по каналах з високою інтенсивністю завад, коли виникають помилки кратності два та більше, пачки помилок. Цей код є частковим випадком коду БЧХ і в ньому елементи та локатори знаходяться в одному скінченому полі Галуа.

Для порівняльного аналізу, з основних параметрів коду Ріда-Соломона були обрані наступні: **q, $\mathbf{t_в}$, n** = q - 1, $r = 2t_в$ та $k = n - r$, **x, R** та **D**.

Нижче наведено приклад одного з варіантів розрахунку параметрів для коду Ріда-Соломона, з виправленням 2 помилок при **q = 8**:

- $n = q - 1 = 7$
- $r = 2t_в = 4$
- $k = n - r = 3$
- $x = \frac{tв}{n} = \frac{2}{7} = 0.29$
- $R = \frac{k}{n} = 0.43$
- $D = \frac{r}{n} = \frac{4}{7} = 0{,}57\,\%$

У Таблиці 3 наведено параметри коду Ріда-Соломона, розраховані при різних **q**: 8, 16, 32, 64 та кількості виправлених помилок $\mathbf{t_в}$= 1, 2 та 3, з метою побачити вплив **q** на ефективність коду.

Табл.3. Параметри коду Ріда-Соломона

$\mathbf{t_в}$ = 1					
q	**4**	**8**	**16**	**32**	**64**
n	3	7	15	31	63
x	0.33	0.143	0.066	0.032	0.016
R	0.33	0.71	0.86	0.935	0.97
$\mathbf{t_в}$ = 2					
q	**4**	**8**	**16**	**32**	**64**
n		7	15	31	63
x		0.29	0.133	0.064	0.063
R		0.428	0.733	0.87	0.94
$\mathbf{t_в}$ = 3					
q	**4**	**8**	**16**	**32**	**64**
n		7	15	31	63
x		0.43	0.2	0.096	0.048
R		0.14	0.6	0.81	0.91

При подібних розрахунках, доводиться погодитися зі зміною довжини КК.

Отримані точки **K(R;x)** з відповідними координатами для кодів Хемінга, БЧХ та Ріда-Соломона з різною кількістю виправлених помилок нанесемо на графіки разом з теоретичними межами, отриманими вище.

На рис.2 зображені верхні межі Хемінга та нижні Варшамова-Гілберта при **q = 2** та **q = ∞.**

Межа Хемінга – одна з верхніх меж для **d**, зміст якої в тому, що не існують коди кращі за ті, що відображаються верхньою межею, тобто немає коду, координати якого перевищили б відповідні координати будь-якої точки верхньої межі.

Теоретична крива для межі Хемінга побудована за формулами (7) та (8). Теоретична крива для нижньої межі Варшамова–Гілберта на всіх наступних графіках побудована за виразом (14).

На графіку Рис. 2 видно, що для кодів, зі збільшенням відносної кількості помилок, що виправляються на довжину КК, швидкість коду значно зменшується.

Так, для коду Ріда-Соломона ($\mathbf{t_в}$ = 1)

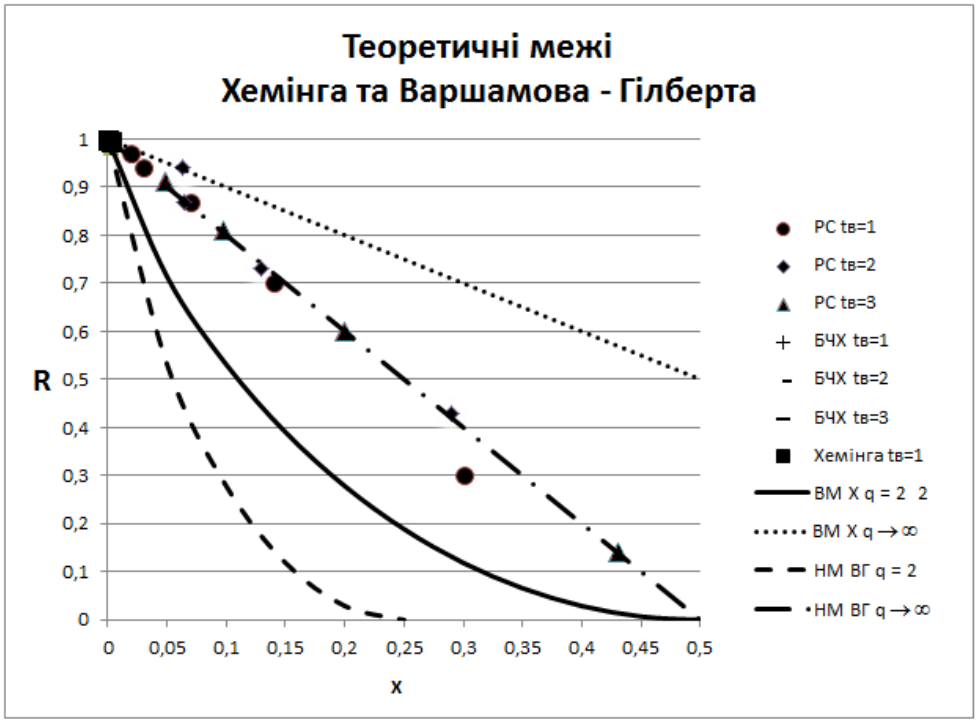


Рис. 2. Межі Хемінга та Варшамова-Гілберта з кодовими точками

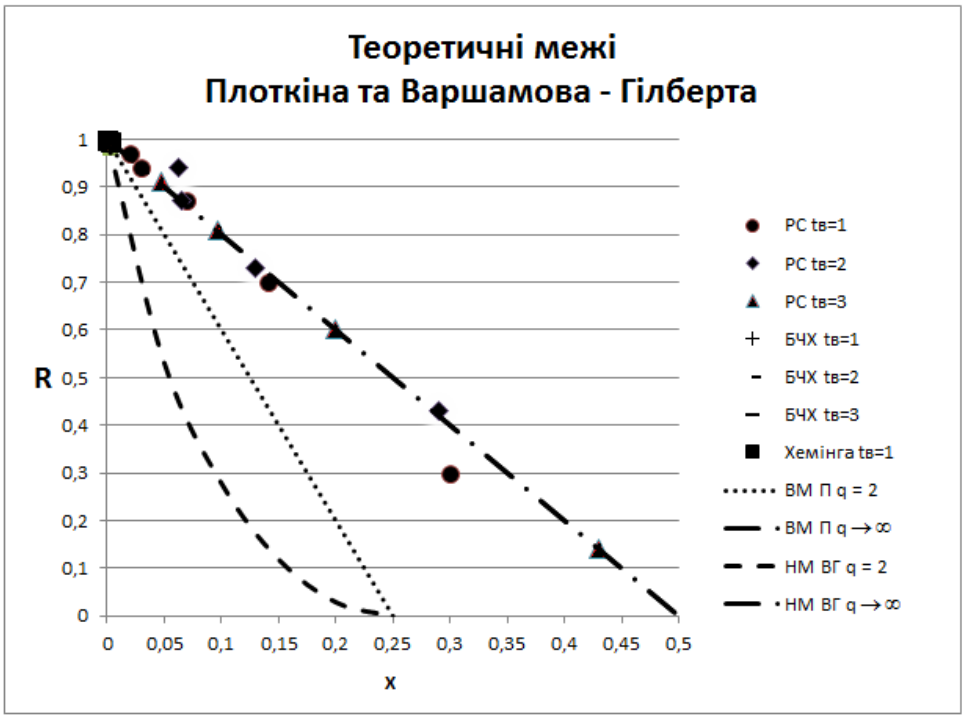


Рис. 3. Межі Плоткіна та Варшамова-Гілберта з кодовими точками

q = 4 **D** = r/n = 2/3 = 66,6%

q = 8 **D** = r/n = 2/7 = 28,5%

q = 16 **D** = r/n = 2/15 = 13,3%

q = 32 **D** = r/n = 2/31 = 6,45%

q = 64 **D** = r/n = 2/63 = 3,17%

В результаті бачимо, що при відносно постійній довжині кодової комбінації та зі збільшенням основи коду q зменшується надлишковість.

Збільшення основи коду **q** призводить до зміщення точок в сторону покращення характеристик ефективності коду.

На Рис. 3 зображені верхні межі Плоткіна та нижні Варшамова-Гілберта при **q = 2** та **q = ∞.**

Теоретична крива для верхньої межі Плоткіна побудована за формулою (3). З графіка випливає, що зі збільшенням **q** збільшується ефективність кодування. Так, при фіксованому

значенні **x**, збільшення **q** приводить до зростання швидкості **R** і зменшення надлишковості **D**.

Так, наприклад, для коду Хемінга

q = 4 **D** = r/n = 6/1365 = 0.43%

q = 8 **D** = r/n = 5/4681 = 0.12%

q = 16 **D** = r/n = 4/4369 = 0.09%

q = 32 **D** = r/n = 3/1057 = 1.1%

q = 64 **D** = r/n = 3/4161 = 0.072%

На рис.4 зображені верхні межі Елайєса та нижні Варшамова-Гілберта при **q = 2** та **q = ∞.**

Порівняння розташування точок для конкретних кодів із ходом кривих меж для мінімальної кодової відстані на всіх рисунках показує гарну відповідність точок межам при відповідних значеннях основи коду **q**.

Видно, що точки для конкретних кодів добре збігаються із розташуванням теоретичних кривих для однакових значень основи коду **q**.

Видно, що збільшення основи **q** призводить до суттєвого збільшення ефективності за **x** та **R**

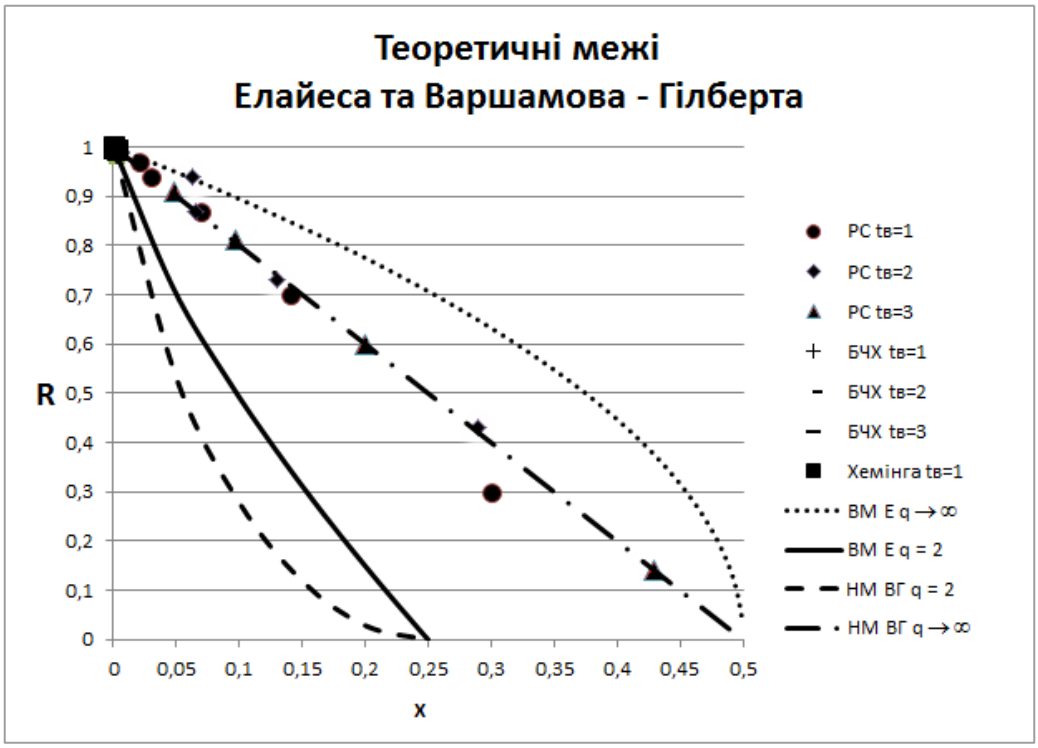


Рис. 4. Межі Елайеса та Варшамова-Гілберта з кодовими точками

Теоретична крива межі Елайеса побудована за виразами (7) та (10). Додатково на графіку нанесені точки, що відповідають ряду кодів із параметрами, розрахованими вище.

На рис.5 зображені верхні межі Хемінга та нижні Варшамова-Гілберта при **q = 2** та **q =** ∞ в більшому масштабі, щоб покращити подання точок високошвидкісних кодів.

Так, наприклад, для коду БЧХ ($\mathbf{t_в}$ = 1)

q = 4 **D** = r/n = 10/1023 = 0.97%

q = 8 **D** = r/n = 8/4095 = 0.19%

q = 16 **D** = r/n = 6/4095 = 0.15%

q = 32 **D** = r/n = 4/1023 = 0.39%

q = 64 **D** = r/n = 4/4095 = 0.097%

Зі збільшенням основи коду спостерігається значне зменшення надлишковості для усіх трьох кодів. Але, звернувши увагу на відносну швидкість коду, бачимо, що для коду БЧХ – вона найбільша, і зі збільшенням **q** суттєво не змінюється.

надлишкових кодів незалежно від конкретного коду чи типу межі для мінімальної **d**.

Більш докладний аналіз ефективності кодів показує, що прирости ефективності при переході від **q** = 2 до більших значень **q** швидко скорочуються і практично стають нерозрізненними при **q** ≥ 32 від випадку **q** = ∞.

Задля збільшення ефективності надлишкових кодів, немає сенсу збільшувати основу **q** до нескінченності, оскільки максимальні значення ефективності, що характерні для випадку **q** = ∞, досягаються практично вже при **q** ≥ 32. Знайдене тут обмеження для **q** може слугувати певним «маяком» при конструюванні кодеків систем передавання даних, оскільки спрямування $q \to \infty$ означає необмежене збільшення складності апаратури та програмного забезпечення її реалізації, що не можна вважати за бажане. І,

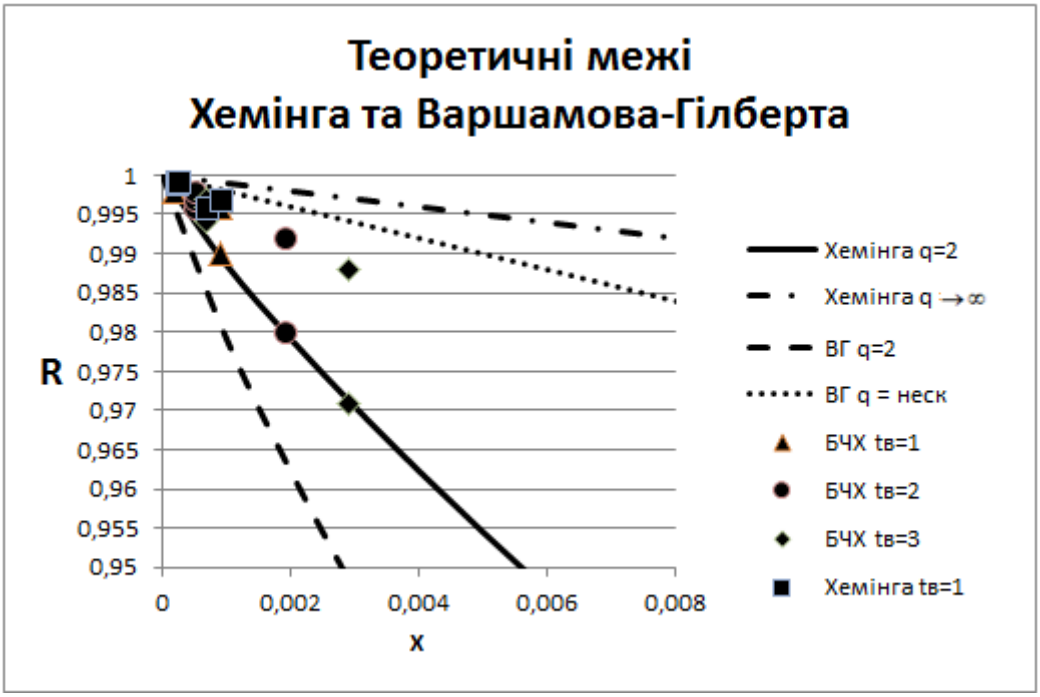


Рис. 5. Межі Хемінга та Варшамова-Гілберта з кодовими точками в збільшеному масштабі

навпаки, обґрунтоване обмеження основи коду **q** ≤ 32 дозволяє досягти максимальних значень ефективності кодів за умови менш витратних характеристик складності апаратури та програмного забезпечення кодеків систем передавання даних.

Детальний аналіз впливу **q** на характеристики надлишкових кодів дозволяє виявити ще одну неочевидну властивість. Мова йде про довжину коду **n**.

Дотепер ще існує відомий «штамп» у мисленні як математиків, так і інженерів «кодовиків» про те, що найбільш ефективними (за критерієм **D**) є «довгі» коди [1, 2] із великими **n**. Однак наше дослідження показує, що це є справедливим лише за умови фіксованого значення **q** = const. Так, до речі, і сформувалося свого часу наведене уявлення про «довгі» коди, як найбільш ефективні. Бо і до сього часу переважна більшість кодів, що використовуються на практиці є суто двійковими з **q** = 2.

Проведене в даній роботі дослідження показує, що високих показників ефективності надлишкового кодування можна досягти і при відносно невеликих довжинах **n** за рахунок збільшення основи коду **q**>2.

Це відкриває перспективи дослідження та використання відносно «коротких» кодів, для яких **n** є співставними з **q**.

Перелік літератури

1. Т. Касами, Н. Токура, Е. Ивадари, Я. Инагаки Теория кодирования. - М.: Мир, 1978. – 576с.
2. У. Питерсон , Э. Уэлдон Коды, исправляющие ошибкию - М.: Мир, 1976. – 590с.
3. Жураковський Ю.П., Полторак В.П. Теорія інформації та кодування: Підручник. – К.: Вища шк., 2001. – 255с.:іл.

УДК 004:519.2

КИРИЧЕНКО Л.О.

ИССЛЕДОВАНИЕ ВЫБОРОЧНЫХ ХАРАКТЕРИСТИК, ПОЛУЧЕННЫХ МЕТОДОМ МУЛЬТИФРАКТАЛЬНОГО ФЛУКТУАЦИОННОГО АНАЛИЗА

В работе проведено численное исследование метода мультифрактального детрендированного флуктуационного анализа на модельных реализациях самоподобных и мультифрактальных процессов. Показано, что для реализаций небольшой длины выборочные мультифрактальные характеристики обладают существенными погрешностями, что может приводить к ошибочной интерпретации численных результатов. Проведен мультифрактальный анализ температурных рядов по городу Киеву. Показано наличие временных интервалов с антиперсистентной и персистентностной зависимостью.

The study of applicability of the Multifractal Detrended Fluctuation Analysis in proper detecting of mono- and multifractal character of data is perform. Shown that the estimated multifractal characteristics obtained by time series of a short length have errors. This can lead to misinterpretation of the numerical results. The multifractal analysis of the temperature series of Kiev is performed.

1. Введение

В настоящее время стало общепризнанным, что многие информационные, биологические, физические, технологические процессы обладают сложной фрактальной структурой. Фрактальный анализ используется для моделирования, анализа и контроля сложных систем в различных областях науки и техники. В частности, в геологии – для прогнозирования сейсмической активности и цунами, определения возраста геологических пород; в биологии и медицине – для диагностики заболеваний и физиологического состояния по записям ЭКГ и ЭЭГ, при исследовании мутаций и изменений на генетическом уровне; в экономике – для прогнозирования кризисных ситуаций и оценивания риска по финансовым рядам; в физике – для исследования турбулентности и термодинамических процессов. Этот список далеко не полон. [1-4]

Процессы, обладающие фрактальными свойствами, можно разделить на две группы: самоподобные (монофрактальные) и мультифрактальные. Монофрактальные процессы являются однородными в том смысле, что их скейлинговые характеристики остаются неизменными на любом диапазоне масштабов и обладают одним показателем скейлинга. Мультифрактальные процессы допускают разложение на участки с различными локальными масштабными свойствами и характеризуются спектром скейлинговых показателей.

Одним из самых популярных инструментов мультифрактального анализа является метод детрендированного флуктуационного анализа, который базируется на идеологии одномерных случайных блужданий и широко используется при анализе временных рядов различной природы. Ориентированный на обработку рядов с трендовыми составляющими, детрендированный флуктуационный анализ является мощным инструментом статистического описания нестационарных процессов. [5,6]

В настоящее время проведен ряд исследований, направленных на изучение статистических свойств данного метода [5,7,8]. Однако в большинстве работ рассмотрены модельные реализации больших длин – более десятка тысяч значений. В этом случае выборочные характеристики достаточно близки к теоретическим. В тоже время, временные ряды, полученные на практике, имеют значительно меньший диапазон значений.

Целью представленной работы является численное исследование метода мультифрактального флуктуационного анализа на модельных реализациях самоподобных и мультифрактальных процессов небольшой длины и применение этого метода для анализа экспериментальных данных.

2. Характеристики фрактальных и мультифрактальных множеств

Самоподобие фрактальных объектов заключается в сохранении структуры объекта при изменении масштаба. Рассмотрим основные характеристики мультифрактального множества [9]. Пусть в общем случае мультифрактальный объект занимает некоторую ограниченную область в d -мерном Евклидовом про-

странстве и определяет собой множество из $N \to \infty$ точек. Разобьем всю область на кубические ячейки с ребром ε и объемом ε^d. Рассмотрим обобщенную статистическую сумму $Z(q,\varepsilon)$, характеризуемую показателем степени q, который может принимать любые значения в интервале $-\infty < q < +\infty$:

$$Z(q,\varepsilon) = \sum_{i=1}^{N(\varepsilon)} p_i^q(\varepsilon), \qquad (1)$$

где $p_i(\varepsilon) = \lim\limits_{N\to\infty} \dfrac{n_i(\varepsilon)}{N}$, $n_i(\varepsilon)$ – количество точек, попавшее в ячейку с номером i, $N(\varepsilon)$ – суммарное количество занятых ячеек, которое зависит от размера ячейки ε. Вероятности p_i характеризуют относительную заселенность ячеек.

В общем случае мультифрактальное множество характеризуется некоторой нелинейной функцией $\tau(q)$, определяющей поведение статистической суммы $Z(q,\varepsilon)$ при $\varepsilon \to 0$:

$$Z(q,\varepsilon) \propto \varepsilon^{\tau(q)}. \qquad (2)$$

Функция $\tau(q)$ обычно называется скейлинговой экспонентой и определяется как

$$\tau(q) = \lim_{\varepsilon\to 0} \frac{\ln Z(q,\varepsilon)}{\ln \varepsilon}. \qquad (3)$$

В случае однородного фрактального множества с фрактальной размерностью D во всех занятых ячейках содержится одинаковое количество точек, т.е., $p_i(\varepsilon) = p(\varepsilon) = 1/N(\varepsilon)$, и обобщенная статистическая сумма принимает вид:

$$Z(q,\varepsilon) = N^{1-q}(\varepsilon) = \varepsilon^{-D(1-q)},$$

а функция $\tau(q) = (q-1)D$ является линейной. Если распределение точек по ячейкам не одинаково, то фрактальное множество является неоднородным, т.е. мультифрактальным, и $\tau(q)$ является нелинейной функцией. При $q \to +\infty$ основной вклад в обобщенную статистическую сумму вносят ячейки, содержащие наибольшее число частиц n_i и, следовательно, характеризующиеся наибольшей вероятностью заполнения p_i. Наоборот, при $q \to -\infty$ основной вклад в сумму дают самые разреженные ячейки с малыми значениями p_i. Таким образом, функция $\tau(q)$ показывает, насколько неоднородным является исследуемое множество точек.

Наряду со скейлинговой экспонентой $\tau(q)$ для характеристики мультифрактального множества используется функция мультифрактального спектра (спектра сингулярностей) $f(\alpha)$. Зависимость вероятности от размера ячейки $p_i(\varepsilon)$ имеет степенной характер

$$p_i(\varepsilon) \propto \varepsilon^{\alpha_i}, \qquad (4)$$

где α_i – представляет собой некоторый показатель степени, вообще говоря разный, для разных ячеек (показатель сингулярности). Для однородного фрактала все показатели степени α_i одинаковы и равны фрактальной размерности D.

Функция мультифрактального спектра $f(\alpha)$ характеризует распределение вероятностей различных значений α_i. Если величина $n(\alpha)d\alpha$ является вероятностью того, что α_i находится в интервале $(\alpha, \alpha + d\alpha)$, т.е. представляет собой число ячеек i, обладающих одинаковой мерой $p_i(\varepsilon)$ с $\alpha_i \in (\alpha, \alpha + d\alpha)$, тогда

$$n(\alpha) \approx \varepsilon^{-f(\alpha)}. \qquad (5)$$

Таким образом, функция $f(\alpha)$ представляет собой фрактальную размерность некоего однородного фрактального подмножества ξ_α из исходного множества ξ, характеризуемого одинаковыми вероятностями заполнения ячеек $p_i(\varepsilon) \approx \varepsilon^{\alpha}$.

Учитывая выражения (1) и (5), обобщенную статистическую сумму $Z(q,\varepsilon)$ можно выразить через функцию мультифрактального спектра $f(\alpha)$ следующим образом:

$$Z(q,\varepsilon) = \sum_{i=1}^{N(\varepsilon)} p_i^q(\varepsilon) \approx \int d\alpha\, n(\alpha)\varepsilon^{q\alpha} \approx \int d\alpha\, \varepsilon^{q\alpha - f(\alpha)}.$$

Формально переход от переменных $\{q, \tau(q)\}$ к переменным $\{\alpha, f(\alpha)\}$ может быть осуществлен при помощи следующих преобразований Лежандра:

$$\begin{cases} \alpha = \dfrac{d\tau}{dq} \\ f(\alpha) = q\dfrac{d\tau}{dq} - \tau \end{cases} \quad \text{и} \quad \begin{cases} q = \dfrac{df}{d\alpha} \\ \tau(q) = \alpha\dfrac{df}{d\alpha} - f \end{cases} \qquad (6)$$

3. Характеристики самоподобных и мультифрактальных случайных процессов

Самоподобие случайных процессов заключается в сохранении статистических характеристик при изменении масштаба времени. Стохастический процесс $X(t)$ является самоподобным с параметром самоподобия H, если про-

цесс $a^{-H}X(at)$ описывается теми же конечномерными законами распределений, что и $X(t)$:

$$\mathrm{Law}\{X(t)\} = \mathrm{Law}\{a^{-H}X(at)\}, \vee a > 0,\ t > 0 \ . \quad (7)$$

Параметр H, $0 < H < 1$, называемый показателем Херста, представляет собой степень самоподобия. Наряду с этим свойством, показатель H характеризует меру долгосрочной зависимости стохастического процесса [1,2,10].

При значениях $0.5 < H < 1$ ряд демонстрирует персистентное (трендоустойчивое) поведение. Если персистентный ряд возрастал (убывал) в предыдущий период, то чем ближе показатель Херста к 1, тем с большей вероятностью будет сохраняться тенденция поведения этого ряда в течение такого же периода в будущем. Значение $H = 0.5$ указывает на независимость (отсутствие какой-либо памяти о прошлом) приращений временного ряда. Диапазон $0 < H < 0,5$ соответствует антиперсистентным рядам: если антиперсистентный ряд характеризовался ростом в предыдущем периоде, то чем ближе показатель Херста к 0, тем с большей вероятностью в следующем периоде начнется спад.

Легко показать, положив $a = 1/t$, что для самоподобного процесса выполняется следующее равенство:

$$\mathrm{Law}\{X(t)\} = \mathrm{Law}\left\{\left(\frac{1}{t}\right)^{-H} X(1)\right\} = \mathrm{Law}\{t^{H}X(1)\} \ . \quad (8)$$

Учитывая (8), начальные моменты самоподобного случайного процесса можно выразить как

$$\begin{aligned} \mathrm{M}\left[\left|X(t)\right|^{q}\right] &= \mathrm{M}\left[\left|t^{H}X(1)\right|^{q}\right] = \\ &= t^{qH}\ \mathrm{M}\left[\left|X(1)\right|^{q}\right] = C(q)\cdot t^{qH} \end{aligned}, \quad (9)$$

где величина $C(q) = \mathrm{M}\left[\left|X(1)\right|^{q}\right]$.

Мультифрактальные процессы проявляют более гибкие скейлинговые закономерности для моментных характеристик. В отличие от самоподобных процессов, где все моменты $\mathrm{M}\left[\left|X(t)\right|^{q}\right]$ показывают одинаковое масштабное поведение, для мультифрактальных процессов выполняется отношение

$$\mathrm{M}\left[\left|X(t)\right|^{q}\right] = C(q)\cdot t^{qh(q)}, \quad (10)$$

где $h(q)$ – обобщенный показатель Херста, являющийся нелинейной функций, для которой значение $h(q=2)$ совпадает со значением степени самоподобия H. Для временных рядов, которые отвечают монофрактальному процессу, обобщенный показатель Херста $h(q) = H$ и не зависит от параметра q.

Обобщенный показатель Херста связан с функцией $\tau(q)$ соотношением [4,5]:

$$\tau(q) = qh(q) - 1. \quad (11)$$

Определение функции мультифрактального спектра $f(\alpha)$ случайного процесса производится по формулам (6).

3. Метод мультифрактального детрендированного флуктуационного анализа

При оценивании параметра H для самоподобных временных рядов используется метод детрендированного флуктуационного анализа (ДФА) [6,10]. В этом случае для исходного временного ряда $x(t)$ строится кумулятивный ряд $y(t) = \sum_{i=1}^{t} x(t)$, который разбивается на N сегментов длиной s. Для каждого сегмента $y(t)$ вычисляется флуктуационная функция

$$F^{2}(s) = \frac{1}{s}\sum_{t=1}^{s}\left(y(t) - Y_{m}(t)\right)^{2}, \quad (12)$$

где $Ym(t)$ – локальный m-полиномиальный тренд в пределах данного сегмента.

Функция $F(s)$ усредняется по всему ряду $y(t)$. Такие вычисления повторяются для различных размеров сегментов, чтобы получить зависимость $F(s)$ в широком диапазоне значений параметра s. Для процессов с фрактальными свойствами с ростом s функция $F(s)$ также возрастает, и линейная зависимость $\log F(s)$ от $\log s$ свидетельствует о наличии свойства масштабной инвариантности:

$$F(s) \propto s^{H}. \quad (13)$$

При исследовании свойств мультифрактальных процессов применяется мультифрактальный флуктуационный анализ (МФДФА) [5-8]. При проведении МФДФА исследуется зависимость флуктуационной функции $F_{q}(s)$ от параметра q:

$$F_{q}(s) = \left\{\frac{1}{N}\sum_{i=1}^{N}\left[F^{2}(s)\right]^{\frac{q}{2}}\right\}^{\frac{1}{q}}, \quad (14)$$

полученной возведением выражения (12) в степень q и последующим усреднением по всем сегментам.

Изменяя временную шкалу s при фиксированном показателе q, находим зависимость $F_q(s)$, представляя её в двойных логарифмических координатах. Если исследуемый ряд сводится к мультифрактальному множеству, проявляющему долгосрочные зависимости, то флуктуационная функция $F_q(s)$ представляется степенной зависимостью

$$F_q(s) \propto s^{h(q)} \qquad (15)$$

с функцией обобщенного показателя Херста $h(q)$. Из определений (12) и (14) следует, что при $q = 2$ этот показатель сводится к обычному значению H. Для временных рядов, которые отвечают монофрактальному множеству, флуктуационная функция $F_q(s)$ одинакова для всех сегментов, и обобщенный показатель Херста $h(q) = H$ не зависит от параметра q. Для мультифрактальных рядов $h(q)$ является нелинейной функцией: при положительных q основной вклад в функцию $F_q(s)$ дают сегменты, проявляющие большие отклонения $F^2(s)$, а при отрицательных q доминируют сегменты с малыми дисперсиями $F^2(s)$. Таким образом, при отрицательных значениях q обобщенный показатель Херста $h(q)$ описывает сегменты, проявляющие малые флуктуации, а при положительных – большие.

4. Исследование характеристик модельных реализаций

В работе представлены результаты численного эксперимента, в ходе которого моделировались реализации трех типов фрактальных стохастических процессов: фрактального броуновского движения (монофрактальный процесс), α – устойчивого процесса (бифрактальный) и биномиального каскада (мультифрактальный). Длина реализаций была выбрана равной 256, 512, 1024 и 2048 значений. Для каждого сгенерированного временного ряда методом МФДФА рассчитывалась мультифрактальные характеристики $h(q)$, $\tau(q)$ и $f(\alpha)$, которые потом усреднялись по множеству реализаций. Значения параметра q изменялись в диапазоне $-5 \le q \le 5$.

Фрактальное броуновское движение. Одной из наиболее известных и простых моделей самоподобного процесса является фрактальное броуновское движение (ФБД) [1,2]. Гауссовский процесс $X(t)$ называется фрактальным броуновским движением с параметром H, $0 < H < 1$, если приращения случайного процесса $\Delta X(\tau) = X(t+\tau) - X(t)$ имеют распределение вида

$$P(\Delta X < x) = \frac{1}{\sqrt{2\pi}\sigma_0 \tau^H} \cdot \int_{-\infty}^{x} \mathrm{Exp}\left[-\frac{z^2}{2\sigma_0^{\,2}\tau^{2H}} \right] dz ,$$

где σ_0 – коэффициент диффузии.

ФБД с параметром $H = 0.5$ совпадает с классическим броуновским движением. Приращения ФБД называются фрактальным гауссовским шумом, дисперсия которого подчиняется соотношению $D[X(t+\tau) - X(t)] = \sigma_0^{\,2}\tau^{2H}$.

На рис. 1 показана реализация ФБД при значении параметра H=0.8 длиной 1000 значений и соответствующая ей реализация фрактального гауссовского шума.

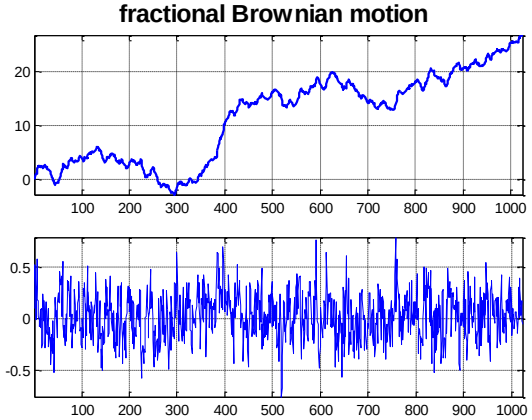


Рис. 1. Реализация ФБД (вверху) и ряд приращений (внизу)

ФБД является монофрактальным процессом, его скейлинговые характеристики полностью определяются значением показателя Херста H.

$$h(q) = H\ ;\ \ \tau(q) = qH - 1;$$

$$\begin{cases} \alpha = H \\ f(\alpha) = 1 \end{cases} \qquad (16)$$

На рис. 2 - 4 представлены функция обобщенного показателя Херста, скейлинговая экспонента и мультифрактальный спектр, полученные по реализациям ФБД разной длины с параметром $H = 0.8$. Пунктирной линией на графиках $h(q)$ и $\tau(q)$ показаны теоретические значения.

Функция $\tau(q)$ визуально очень близка к линейной. Для наглядности значения $\tau(q)$ приве-

дены в диапазоне $4 \le q \le 5$. Мультифрактальный спектр $f(\alpha)$ представляет собой точку $(0.8,1)$.

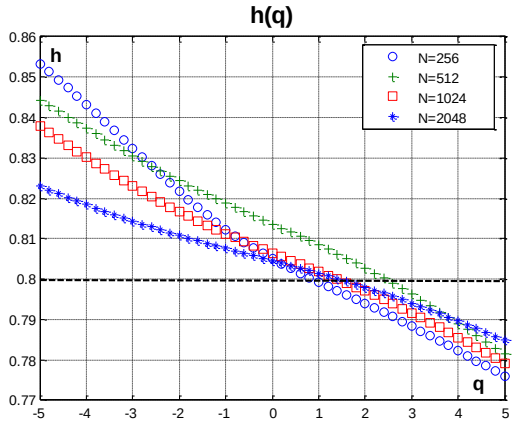


Рис. 2. Функция $h(q)$ ***для реализаций ФБД***

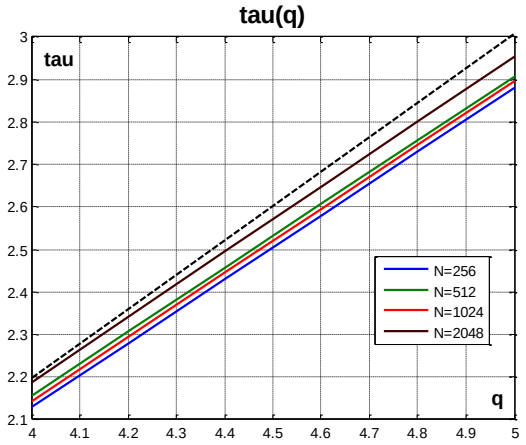


Рис. 3. Функция $\tau(q)$ ***для реализаций ФБД***

Очевидно, что с увеличением длины реализации выборочные характеристики стремятся к своим аналитическим значениям. Однако при небольшом числе значений во временном ряде его характеристики $h(q)$ и $f(\alpha)$ демонстрируют мультифрактальные свойства.

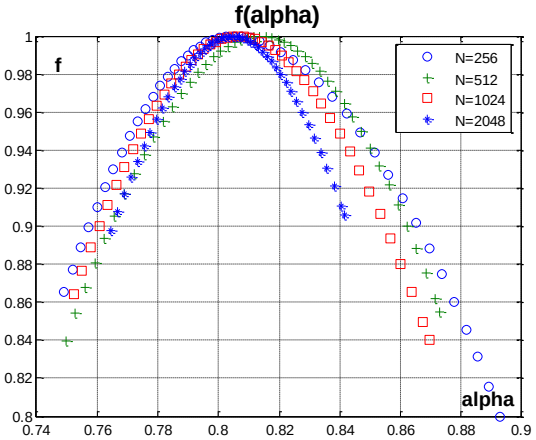


Рис. 4. Функция $f(\alpha)$ ***для реализаций ФБД***

В работах [11,12] отмечается, что одним из недостатков оценивания мультифрактального спектра является то, что оценивается верхняя огибающая истинного спектра. Это может приводить к неправильной интерпретации численных результатов, поскольку при истинном дискретном спектре $f(\alpha)$, полученная огибающая будет включать множество ложных точек. Поэтому в случае рядов небольшой длины возникает необходимость проведения дополнительных исследования для подтверждения наличия мультифрактальных свойств.

α – устойчивый процесс с независимыми приращениями. Случайные величины с устойчивыми законами распределений играют большую роль в моделировании экономических и информационных процессов. В общем случае, для α -устойчивых распределений в явном виде может быть записана только характеристическая функция $\varphi(t) = \mathrm{M}[e^{itX}]$. Характеристическая функция устойчивой случайной величины $X \sim S_\alpha(\sigma,\beta,\mu)$ с параметрами $0 < \alpha \le 2$, $\sigma \ge 0$, $-1 \le \beta \le 1$ и $\mu \in R$ имеет вид [13]:

$$\ln \varphi(t) = \begin{cases} -\sigma^\alpha |t|^\alpha \left(1 - i\beta \operatorname{sign}(t) \tan\frac{\pi\alpha}{2}\right) + i\mu t, \alpha \ne 1 \\ -\sigma |t| \left(1 + i\beta \operatorname{sign}(t) \frac{2}{\pi} \ln|t|\right) + i\mu t, \quad \alpha = 1 \end{cases}.$$

Параметр α называется индексом устойчивости и определяет, насколько выражен тяжелый хвост распределения. При $0 < \alpha < 2$ случайные величины имеют бесконечную дисперсию, а при $0 < \alpha \le 1$ обладают бесконечным средним. Параметр смещения β задает степень асимметричности распределения, величина σ выражает степень разброса значений относительно среднего значения, μ при $\alpha > 1$ равно математическому ожиданию величины X.

Случайный процесс является α – устойчивым процессом, если его конечномерные распределения являются устойчивыми случайными величинами.

На рис. 5 показан временной ряд, приращениями которого являются независимые случайные величины с параметром $\alpha = 1.2$: $X_i \sim S_{1.2}(1,0,0)$.

Для α – устойчивого процесса с независимыми приращениями выполняется равенство:

$$\mathrm{Law}\{X(at)\} = \mathrm{Law}\{a^{1/\alpha} X(t)\}, \vee a > 0,\ t > 0 \quad (17)$$

Сравнивая выражения (7) и (17), понятно, что α – устойчивого процессы обладают свойствами самоподобия. Показано [14], что такие процессы являются бифрактальными. Соответствующие характеристики имеют вид:

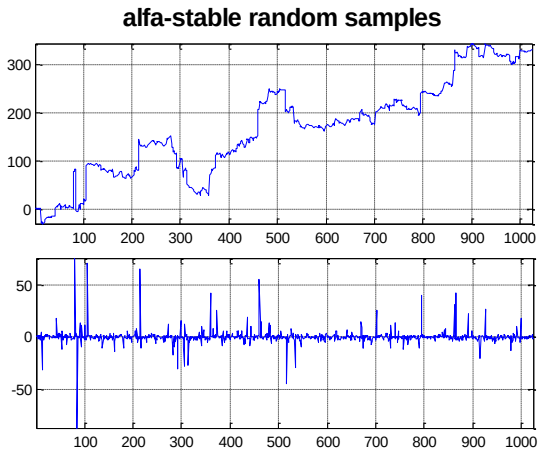


Рис. 5. Реализация устойчивого процесса (вверху) и ряд приращений (внизу)

$$h(q)=\begin{cases}1/\alpha_L & q\le\alpha_L\\ 1/q & q>\alpha_L\end{cases};\tau(q)=\begin{cases}q/\alpha_L-1 & q\le\alpha_L\\ 0 & q>\alpha_L\end{cases};$$

$$\begin{cases}\alpha=\begin{cases}1/\alpha_L & q\le\alpha_L\\ 0 & q>\alpha_L\end{cases}\\ f(\alpha)=\begin{cases}1 & q\le\alpha_L\\ 0 & q>\alpha_L\end{cases}\end{cases}, \qquad (18)$$

где величина α_L в данном случае обозначает индекс устойчивости.

На рис. 6 - 8 представлены мультифрактальные характеристики $h(q)$, $\tau(q)$ и $f(\alpha)$, полученные по реализациям устойчивого процесса разной длины с параметром $\alpha_L=1.2$. Реализации такого процесса имеют теоретическую бесконечную дисперсию и являются самоподобными с показателем Херста $H=\dfrac{1}{\alpha_L}=\dfrac{5}{6}$.

Пунктирной линией на графиках $h(q)$ и $\tau(q)$ показаны теоретические значения. Мультифрактальный спектр представляет собой две точки на плоскости $\{\alpha, f(\alpha)\}$: $(0,0)$ и $(5/6,1)$.

В спектре $f(\alpha)$, как и в случае оценивания характеристик для ряда броуновского движения, характерно наличие ложных точек, что может привести к неправильным выводам о мультифрактальных свойствах реализаций.

Биномиальный каскад. Простейшей моделью мультифрактального процесса с заданными свойствами является биномиальный каскад [1-4]. При его построении первоначальный единичный отрезок делится на два равных интервала, которым приписываются весовые коэффициенты p_1 и $p_2=1-p_1$ соответственно. Затем с каждым из интервалов проделывается аналогичная процедура. В результате на втором

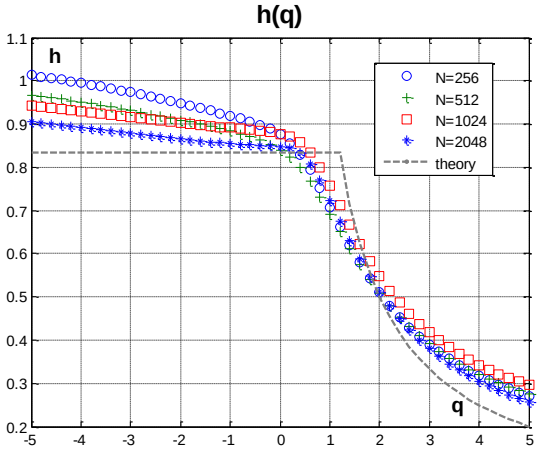


Рис. 6. Функция $h(q)$ ***для реализаций*** α ***– устойчивого процесса***

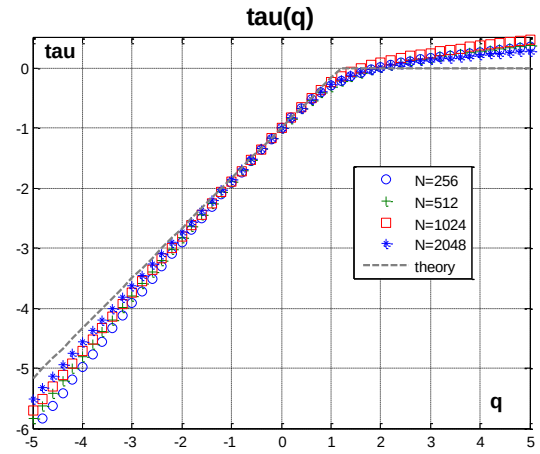


Рис. 7. Функция $\tau(q)$ ***для реализаций*** α ***– устойчивого процесса***

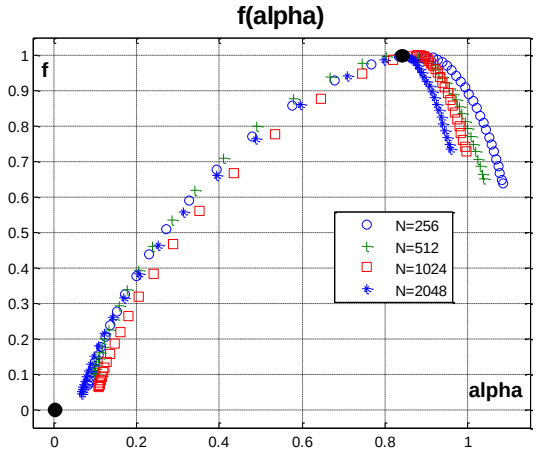


Рис.8. Функция $f(\alpha)$ ***для реализаций*** α ***– устойчивого процесса***

шаге имеется 4 интервала с весовыми коэффициентами p_1^2, p_1p_2, p_2p_1 и p_2^2. При числе шагов $n\to\infty$ и $p_1\ne p_2$ мы приходим к неоднородному фрактальному множеству. На рис. 9 показан временной ряд значений биномиального каскада при $p_1=0.7$ и $n=10$, т.е. длиной реализации равной 2^{10} значений и соответствующий кумулятивный ряд.

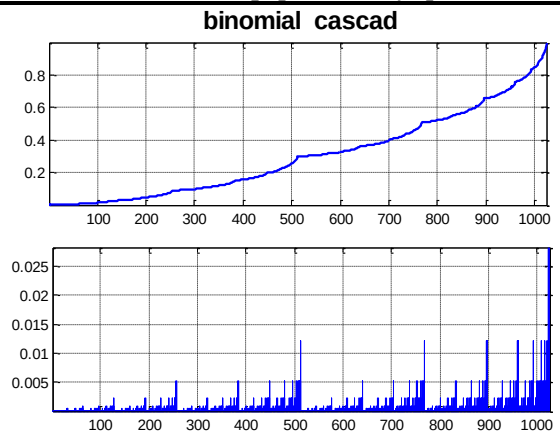


Рис. 9. Реализация биномиального каскада (вверху) и кумулятивный ряд (внизу)

Характеристики $h(q)$, $\tau(q)$ и $f(\alpha)$ для биномиального мультифрактального процесса зависит только от весовых коэффициентов p_1 и p_2 и определяются аналитически:

$$h(q) = \left(\frac{1}{q} - \frac{\ln(p_1^q + p_2^q)}{q \ln 2}\right); \tau(q) = \frac{-\ln(p_1^q + p_2^q)}{\ln 2} \quad (19)$$

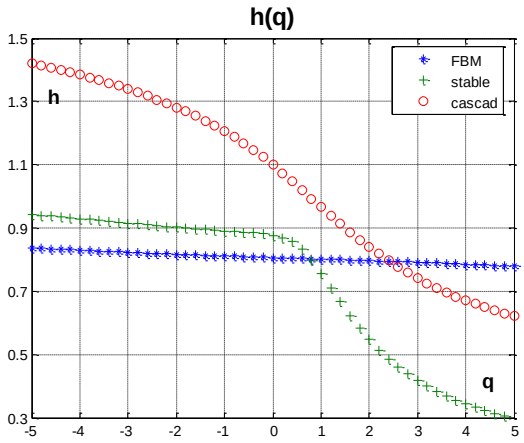
$$\begin{cases} \alpha = -\dfrac{p_1^q \ln(p_1) + p_2^q \ln(p_2)}{\ln 2\,(p_1^q + p_2^q)} \\ f(\alpha) = -\dfrac{q}{\ln 2}\dfrac{p_1^q \ln(p_1) + p_2^q \ln(p_2)}{(p_1^q + p_2^q)} + \dfrac{\ln(p_1^q + p_2^q)}{\ln 2} \end{cases}$$

На рис. 10 - 12 представлены мультифрактальные характеристики, полученные по реализациям биномиального каскада разной длины с коэффициентом $p_1 = 0.7$. Пунктирной линией на графиках показаны теоретические значения.

С увеличением длины реализации выборочные характеристики стремятся к своим аналитическим значениям. При небольшом числе значений во временном ряде его характеристики демонстрируют смещение диапазона значений.

Обобщая результаты проведенного мультифрактального анализа, возникает необходимость дальнейших исследований и построения доверительных интервалов для выборочных характеристик, что позволит принимать или отвергать гипотезы о наличии моно- и мультифрактальных свойств. На рис. 13 - 14 представлены выборочные функции $h(q)$ и $f(\alpha)$, рассмотренные выше полученные по реализациям длиной 1024 значения. Построение скейлинговой экспоненты $\tau(q)$ не дает наглядной информации о различиях фрактальных свойств.

На совместном графике различия между моно- и мультифрактальными характеристиками проявляются наглядно. Поэтому имеет смысл при анализе экспериментальных данных рассматривать полученные выборочные характеристики совместно с аналогичными выборочными характеристиками модельных процессов.



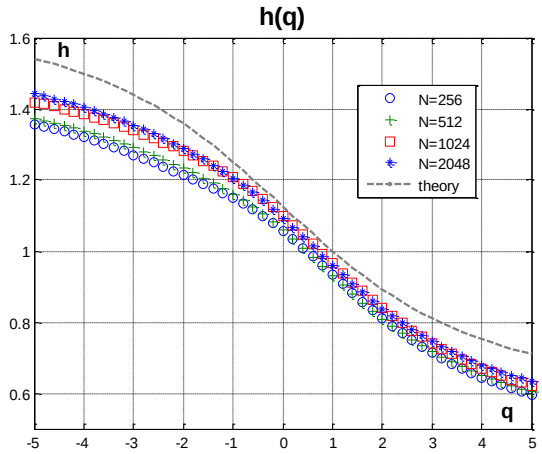


Рис.10. Функция h(q) для реализаций биномиального каскада

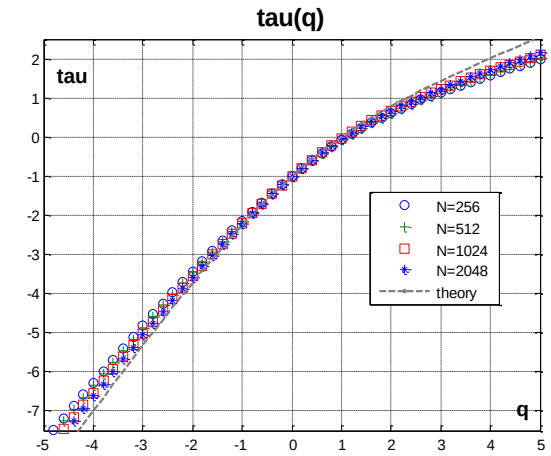


Рис. 11. Функция $\tau(q)$ для реализаций биномиального каскада

Хорошо известно, что многие временные ряды, отвечающие природным явлениям, имеют самоподобную структуру. К таким явлениям относятся разливы рек, количество выпавших

осадков, изменение температуры, увеличение диаметра колец деревьев. Почти все они являются персистентными. [1-3]

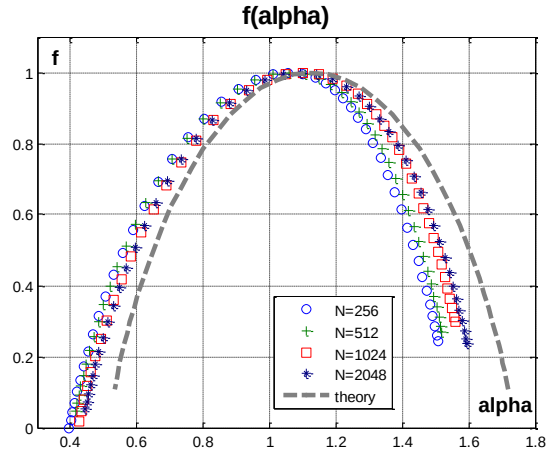


Рис. 12. Функция $f(\alpha)$ ***для реализаций биномиального каскада***

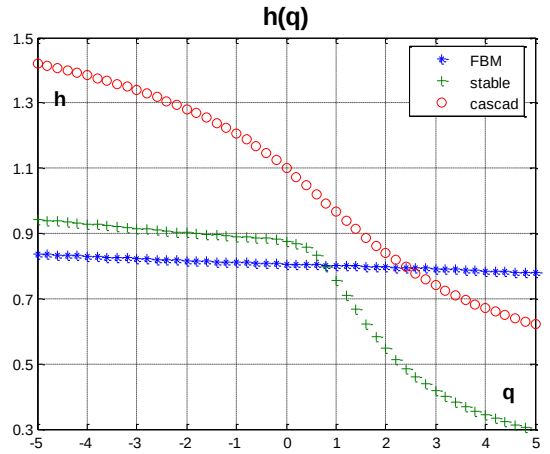


Рис. 13. Функции $h(q)$ ***для реализаций ФБД,*** α ***– устойчивого процесса. биномиального каскада***

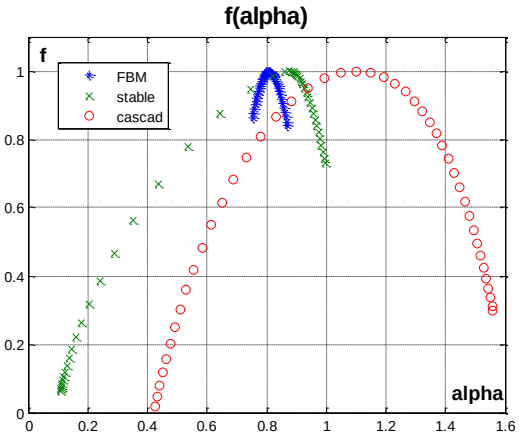


Рис. 14. Функции $f(\alpha)$ ***для реализаций ФБД,*** α ***– устойчивого процесса. биномиального каскада***

6. Исследование температурных рядов

В данной работе рассмотрены ряды ежедневной температуры (максимальное, минимальное и среднее значения) по городу Киеву за 50 лет с 1942 по 1992 годы. Данные взяты на специализированном сайте [15]. На рис.15 показан фрагмент температурного ряда за 1952-1956 гг. (вверху) и соответствующий ряд ежедневных приращений (внизу).

Перед проведением мультифрактального анализа, необходимо исследовать флуктуационную функцию $F(\tau)$, определяемую

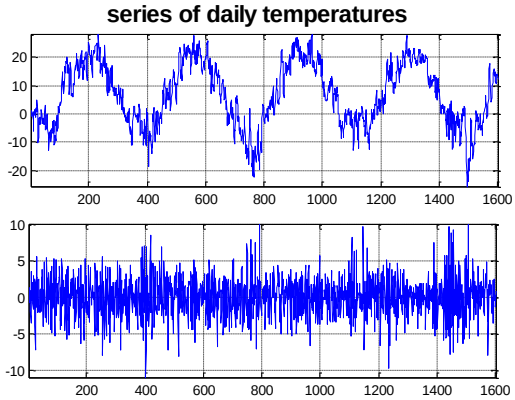


Рис. 15. Ряд ежедневных температур и ряд ежедневных приращений

формулой (2) . Самоподобному поведению отвечает наличие участка с линейной зависимостью. Если функция $F(\tau)$ имеет несколько линейных участков, это предполагает наличие нескольких скейлингов для различных временных интервалов [10]. На рис.16 приведен график $F_2(s)$, построенной на временном интервале от 2^5 (месяц) до $2^{9.5}$ (два года) дней. Визуально очевидно, что график имеет два близких к линейным участка с разными углами наклона. Стоит отметить сильный выброс на втором участке, который соответствует значению аргумента $2^{8.5}$ и практически равен одному году, что отвечает годовой сезонной составляющей процесса.

Мультифрактальный анализ был проведен отдельно для каждого участка. Для первого участка параметр Херста $H = 0.21$, для второго $H = 0.92$. На рис. 17 показаны функции мультифрактального спектра для обоих участков.

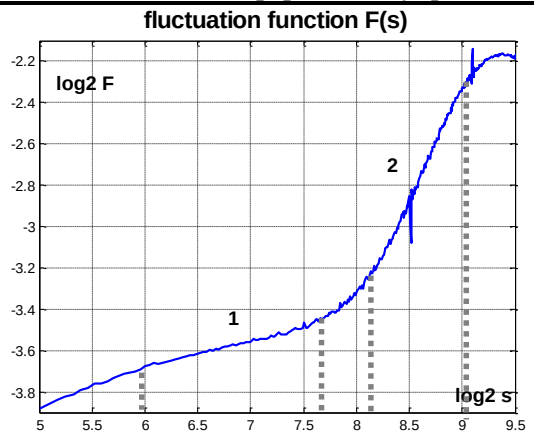


Рис.16. Флуктуационная функция $F_2(s)$ ***для температурного ряда***

Таким образом, можно сделать вывод, что на интервалах от двух месяцев до полугода (участок 1) ряд температурной зависимости является антиперсистентным, а при временных значениях от полугода до полутора лет (участок 2), ряд обладает сильной долгосрочной зависимостью.

Учитывая большое число значений временного ряда, можно предположить наличие мультифрактальных свойств.

9. Заключение

В работе рассмотрено исследование выборочных мультифрактальных характеристик, полученных методом мультифрактального флуктуационного анализа, по реализациям самоподобных и мультифрактальных процессов небольшой длины. Показано, что для реализаций с теоретическим дискретным мультифрактальным спектром в выборочном спектре появляются дополнительные ложные значения, что может приводить к ошибочной интерпретации результатов.

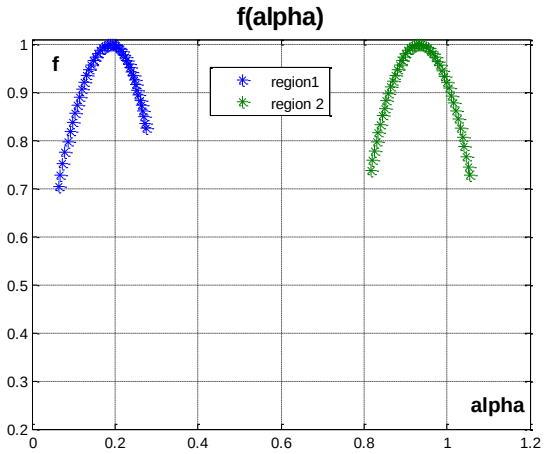


Рис. 17. Функции $f(\alpha)$ ***для участков антиперсистентной и персистентной зависимости***

Проведен фрактальный анализ температурных рядов по городу Киеву за 50 лет. Показано наличие временных интервалов с антиперсистентной и персистентностной зависимостью.

Автор выражает большую благодарность профессору Горбаню И.И. за проявленный интерес и полезные обсуждения свойств фрактальных случайных процессов.

Список литературы

1. Федер Е. Фракталы / Е. Федер – М.: Мир, 1991. – 254 с..
2. Мандельброт Б. Фрактальная геометрия природы / Б. Мандельброт – Москва–Ижевск : Институт компьютерных исследований, 2002. – 656 с..
3. Шредер М. Фракталы, хаос и степенные ряды. Миниатюры из бесконечного рая. / М.Шредер. – Ижевск: НИЦ «Регулярная и хаотичекая динамика», 2005. – 528 с.
4. Riedi R.H. Multifractal processes, in Doukhan P., Oppenheim G., Taqqu M.S. (Eds.), Long Range Dependence: Theory and Applications, p. 625–715, Birkhuser. 2002.
5. Kantelhardt J.W. Multifractal detrended fluctuation analysis of non-stationary time series / J.W. Kantelhardt, S.A. Zschiegner, A. Bunde, S. Havlin, E. Koscielny-Bunde, H.E. Stanley // Physica A. – 2002. – № 316. – P. 87-114.
6. Kantelhardt J.W. Fractal and Multifractal Time Series. – 2008 [Электронный ресурс]: http://arxiv.org/abs/0804.0747
7. Олемской А.И. Мультифрактальный анализ временных рядов / А.И. Олемской, В.Н. Борисюк, И.А. Шуда // Вісник СумДУ. Серія «Фізика, математика, механіка». – 2008. – №2.– С.70-81.
8. Oswiecimka P. Wavelet versus detrended fluctuation analysis of multifractal structures / P. Oswiecimka, J. Kwapin, S.Drozdz // Physical Review E: Statistical, Nonlinear, and Soft Matter Physics. - Vol. 74. -2006. -P. 161-203.
9. Божокин С.В. Фракталы и мультифракталы / С.В. Божокин, Д. А. Паршин. – Ижевск: НИЦ «Регулярпая и хаотическая динамика». – 2001. – 128 с.

10. Kantelhardt J.W. Detecting long-range correlations with detrended fluctuation analysis / J.W. Kantelhardt, E. Koscielny-Bunde, H.H.A. Rego, S. Havlin, A. Bunde // Physica A. – 2001. – № 295. – P. 441-454.
11. Павлов А.Н. Мультифрактальный анализ сигналов / А.Н. Павлов, В.С. Анищенко // Известия Саратовского университета. Серия «Физика». - 2007. -Т. 7. -Вып. 1.-С.3-25
12. Veneziano D. Multifractal analysis: pitfalls of standard procedures and alternatives / D. Veneziano, G.E. Moglen, R.L. Bras // Phys. Rev. E. -1995. -V.52. - P.1387−1398.
13. Borak S. Stable distributions / Borak, S., W. Hardle, R. Weron in Statistical tools for finance and insurance. Berlin: Springer. -2005. -P. 21-44
14. Nakao H Multi-scaling properties of truncated Levy flights / H. Nakao //Phys. Lett. A. – 2000. –V.266 –P. 282-289
15. Архив погоды по городам СНГ [Электронный ресурс]: http://thermo.karelia.ru/ weather/w_history.php

УДК 004.415.2.043

САЛАПАТОВ В.І.

ПІДГОТОВКА АДРЕСИ ОПЕРАНДІВ ПРИ СИНТЕЗІ КОДІВ ПРОГРАМ

У статті розглядається метод підготовки операндів команд під час синтезу машинного коду команд у компіляторах. Пропонується узагальнена модель даних для інформаційних ресурсів. Розроблена принципіально нова схема алгоритму синтезу оптимального машинного коду програм для цільових ЕОМ.

In the article a method of preparation operands at the synthesis of machine code commands in compilers is discussed. A generalization of the data model for information resources is proposed. It's developed a fundamentally new scheme of the synthesis of optimal machine code programs for the target computer.

Вступ

Формування кодів команд у процесі синтезу кодів програм вимагає підготовки адрес операндів. Для цього треба використовувати або вже існуючі данні з інформаційних ресурсів, які вже присутні у програмно доступних регістрах (ПДР), або спеціально формувати відповідні команди пересилок для підготовки необхідних для формування кодів команд адрес. Традиційні методи компіляції передбачають розміщення операндів у певних ПДР, що є однією з причин надмірності програмного коду. Адже для створення оптимальних за певними критеріями програм на рівні машинного коду треба використовувати усі наявні на кожен момент виконання програм данні. Обмеження стосовно використання ПДР при компіляції породжує низку зайвих команд пересилок даних, частка яких є досить значною. Тому дуже важливо використати у повній мірі як вже існуючі інформаційні ресурси (ІР), так і оптимальні за форматом та використанням ПДР команди підготовки адрес операндів для машинних команд. Така машинно-залежна оптимізація коду програм вимагає нового підходу щодо формування машинних кодів. На відміну від звичайної генерації машинного коду за жорсткими правилами маємо синтез кодів з максимальним використанням ІР та системи команд цільової ЕОМ (ЦЕОМ) разом з ПДР цільової ЕОМ. Саме так створюються стислі програми на мовах Асемблер. Але програмування на Асемблері досить трудомістке і вимагає значних зусиль та часу. Компілятори мов програмування полегшують створення програм, але все ще вносять значну надмірність коду програм. Тому проблема підвищення продуктивності комп'ютерних систем за рахунок підвищення якості машинних кодів програм є актуальною і являє собою як теоретичний, так і практичний інтерес.

Огляд особливостей внутрішнього подання програм

Для реалізації машинно-залежної оптимізації кодів програм з максимальним використанням наявних ІР та системи команд ЦЕОМ з одного боку треба мати опис системи команд та особливостей архітектури ЦЕОМ [1, 2], а з другого – зберігати та постійно оновлювати дані про ІР. Процес компіляції програм з мов програмування високого рівня із застосуванням методу машинно-залежної оптимізації передбачає наявність таких даних: початкова програма користувача, опис системи команд ЦЕОМ, опис ІР та вибір форми подання ІР. У процесі синтаксичного розбору програми вона перетворюється у низку внутрішніх уявлень. Кінцеве уявлення повинно складатися з низки операторів, які за своєю семантикою наближені до системи команд ЦЕОМ, але без прив'язки даних до конкретного типу адресації з посиланням тільки на імена змінних та констант. Треба зазначити, що компілятори використовують тільки частину машинних команд ЦЕОМ. Це перш за все арифметико-логічні команди, команди пересилок даних, команди зсуву та команди розгалуження. Решта машинних команд використовуються стандартними та спеціалізованими бібліотечними підпрограмами. Крім того, для внутрішнього подання треба, також, враховувати наявність розширених арифметичних команд. Так, для сучасних комп'ютерів Intel та інших це пов'язано з наявністю співпроцесора, де на рівні мікропрограм реалізовано арифметичні операції над дійсними даними та обчислення деяких функцій. Сигнальні процесори родини ADSP-21xx реалізують однією командою виконання обчислення A+B*C і можуть при цьому суміщати до двох пересилок даних. Це також треба враховувати для кінцевого внутрішнього подання програми. Таким чином, необхідно ви-

значити ті оператори внутрішнього подання, які за своєю семантикою присутні у системі команд ЦЕОМ. Далі, треба обрати форму подання даних про ІР. У подальшому необхідно обрати спосіб формування асемблерних операторів ЦОЕМ, які відображають фактично машинний код. Перетворення програм в їх асемблерний еквівалент використовується майже усіма сучасними компіляторами і є досить зручним як при реалізації, так і при аналізі.

Перетворення внутрішнього уявлення програм в асемблерний еквівалент

Сигнальні процесори родини ADSP-21xx мають нестандартні арифметичні команди, які дозволяють прискорити обчислення, перевірити умову їх виконання та сумістити ще до двох пересилок. Тобто, рівень машинних команд наближається до рівня мікропрограмування. З наявно існуючих груп команд ЦЕОМ треба обрати ті, що будуть відображатись внутрішніми операторами компілятора. З групи арифметико-логічних команд обираються лише ті, які не потребують додаткового обчислення. Наприклад, обчислення функції сінусу у команді співпроцесора для комп'ютерів Intel пов'язано з обмеженим діапазоном аргументів і вимагає перерахунку значення аргументу, тому такі команди використовуються тільки у бібліотечних підпрограмах. Таким чином, арифметикологічні оператори внутрішнього уявлення охоплюють семантики тільки тих машинних команд, які одразу дають остаточний результат ALU = $S(M_{al})$ без попереднього перерахунку. ALU – це семантики арифметико-логічних операто-рів внутрішнього уявлення, S функція взяття семантики, M_{al} – машинні команди, які дають кінцевий результат. Команди пересилок повинні охоплювати пересилки даних та адрес у форматах регістр-регістр, регістр-пам'ять, пам'ять-регістр, регістр – безпосередній операнд та пам'ять-пам'ять. Тобто MOV = $S(M_{rr}, M_{rm}, M_{mr}, M_{ri}, M_{mm})$. MOV – це семантики операторів пересилок внутрішнього уявлення, M_{rr}, M_{rm}, M_{mr}, M_{ri}, M_{mm} – команди пересилок усіх перелічених вище форматів. Ці команди використовуються при компіляції як допоміжні для підготовки операндів для інших команд. Команди зсуву часто використовуються у виразах, тому доцільно включити внутрішні оператори з семантикою зсуву у перелік операторів внутрішнього уявлення. До них відносяться арифметичні, логічні та циклічні оператори зсуву праворуч та ліворуч. Отже, SHIFT = $S(S_{ar}, S_{al}, S_{lr}, S_{ll}, S_{rr}, S_{rl})$, де S_{ar}, S_{al}, S_{lr}, S_{ll}, S_{rr}, S_{rl} – машинні команди правого та лівого зсувів арифметичного, логічного та циклічного відповідно, а SHIFT – семантики операторів зсуву внутрішнього уявлення. І, насамкінець, команди розгалуження, до яких відносяться команди безумовного та умовного переходів та зациклювання за шістьома умовами ($<$, $\leq$, $>$, $\geq$, $=$, $\neq$) з урахуванням знаку та без нього. Сюди ж треба віднести команди звернення до підпрограм та повернення з них, або CHANGE = S(Jmp, Jxx, Bxx, LOOP, Lxx, CALL, RET). CHANGE – це множина семантик операторів розгалуження внутрішнього уявлення. Jmp, Jxx, Bxx, Loop, Lxx, CALL, RET – це відповідні машинні команди ЦЕОМ розгалуження.

Стосовно подання опису машинних команд, то раніше вже було обрано спеціальну структуру бази даних (БД), за допомогою якої повністю описуються семантики кожної машинної команди [1, 4].

Залишилось обрати форму подання ІР. Для простоти звернення та наступної обробки ІР треба подати у вигляді реляційної БД. Ключовими елементами головного відношення мають бути змінні або елементи масивів певного типу. Решта атрибутів кожного запису таблиці визначає тип даних, його довжину та спосіб його розміщення через ПДР або стек. У відповідності до цього структура такої реляційної таблиці TABIP з ІР має наступний вигляд.

Ім'я змінної	Тип змінної	Довжина	Розміщення	Ідентифікатор
NAME	TYPE	LEN	ADDR	ID

Рис. 1. Структура відношення опису ІР

Якщо змінна являє собою елемент масиву, то для вказівки на сукупність індексів масиву до основного відношення необхідно додати ще відношення, яке вказує на елемент масиву за сукупністю його індексів. Структура такого відношення INDEX подана на рисунку 2 нижче. Атрибути ID та NDX є ключовими і пов'язують елемент масиву за сукупністю значень усіх його індексів. Розмірність масиву визначає максимальну кількість записів у цій таблиці. Кількість записів для одного елементу відповідає розмірності

масиву. Зв'язок з основним відношенням – один до багатьох.

Ідентифікатор	Номер індексу	Значення індексу	MAX індексу
ID	NDX	NUM	MAXNDX

Рис. 2. Структура відношення для даних типу масив

Для вказівки способу розміщення операнда як IP необхідно ще одне відношення, яке б вказувало на спосіб адресації до наявного IP. Атрибути ID та ADDR є ключовими і пов'язують елемент масиву за способом його розміщення (ПДР, пам'ять, вид адресації у пам'ять). Атрибут TREG вказує на тип ПДР (базовий, індексний, тощо). Ім'я ПДР визначає конкретний ПДР. Структура відношення LOCATE, яке визначає розміщення IP показана на Рис. 3. Атрибути ID та ADDR є ключовими. Зв'язок з основною таблицею – один до багатьох.

Ідентифікатор	Розміщення	Тип ПДР	Ім'я ПДР
ID	ADDR	TREG	NAMREG

Рис. 3. Структура відношення розміщення даних

Якщо потрібний IP для формування асемблерного оператора є у наявності, то через відношення TABIP він буде знайдений. У разі масиву адреса операнда через відношення INDEX перераховується у відповідності із значеннями індексів та формується додатковими командами.

У разі відсутності IP необхідно сінтезувати команди пересилки для доступу до потрібного операнда. Блок-схема алгоритму синтезу асемблерних операторів подана на Рис. 4.

Алгоритм являє собою цикл аналізу операторів внутрішнього подання та синтезу асемблерних операторів програми користувача. Цикл закінчується по досягненню останнього оператора. У процесі аналізу знаходиться машинна команда з відповідною семантикою та готуються операнди. Якщо операнд присутній у відношенні TABIP, то відбувається синтез оператора, інакше формуються допоміжні асемблерні оператори по розміщенню потрібних операндів або їх адрес у ПДР. Далі у циклі відбувається пошук та формування наступних операндів, що вказані в операторі внутрішнього подання аж до остаточного формування оператора Асемблера ЦЕОМ. Якщо машинна команда має кілька форматів, то обирається найкращий за визначеним критерієм оптимальності [3, 4, 5]. У відношенні TABIP код розміщення може вказувати на стек з вказівкою на відповідний номер. При цьому при запису чергового даного у стек усі номери елементів стеку збільшуються на 1, а при читанні зі стеку – номери зменшуються на 1. З урахуванням довжини операнда у стеку обчислюється його фактична адреса. Таким чином, якщо після машинно-незалежної оптимізації внутрішнього подання вважається, що програма оптимальна, то її машинний код також буде максимально наближений до оптимального. Чому ж тільки наближений? Тому що тут не враховуються різні додаткові програмістські хитрощі, які не враховані у цьому загальному алгоритмі. Це пов'язано з деякими особливостями архітектури та системи команд конкретної ЦЕОМ. Компілятор мови С для сигнальних процесорів родини ADSP-21xx, який породжує машинні коди за обраною схемою, враховує деякі специфічні команди, що дозволяють об'єднати кілька дій в одній команді. Так, наприклад враховується наявність машинних команд типу A+B*C з одночасними пересилками даних. Фірмовий компілятор породжує кілька команд і не враховує особливостей архітектури та системи команд сигнального процесора.

Треба зазначити, що у блоці синтезу команд пересилок постійно виконуються дії по оновленню БД IP.

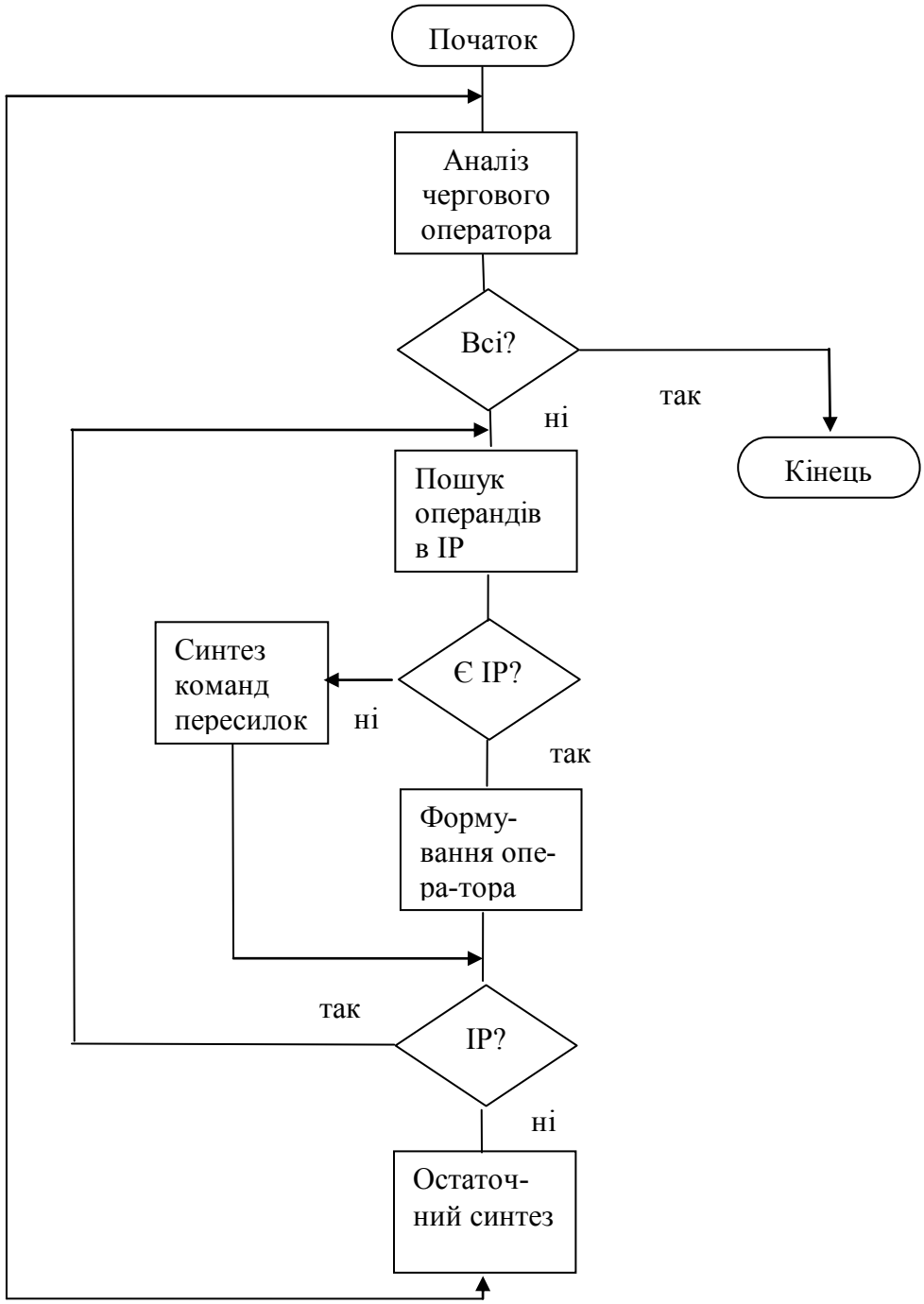


Рис. 4. Блок-схема алгоритму синтезу кодів команд

Висновки

Запропонована узагальнена модель даних ІР, а також принципово нова схема алгоритму синтезу машинних кодів програм. Це дозволяє здійснювати у компіляторах машинно-залежну оптимі-зацію кодів програм за обраним критерієм під час синтезу машинних кодів. Визначені групи машинних команд, які використовуються компіляторами та обрані оператори внутрішнього уявлення компіляторів.

Для подання даних про ІР обрано реляційну модель, яка дозволяє формалізувати операції над даними за допомогою операцій реляційної алгебри і формувати оптимальний асемблерний еквівалент програми користувача. Така схема синтезу машинного коду програм може бути застосована для різних компіляторів різних ЦЕОМ. При цьому за рахунок ускладнення схеми компіляції на виході маємо оптимальний код програми користувача у вигляді асемблерного тексту. Далі для отримання об'єктного файлу текст програми обробляється транслятором з Асемблера ЦЕОМ. Стосовно особливостей архітектури та системи команд конкретної ЦЕОМ, то вони мають бути враховані у схемі

синтезу кодів команд окремо. Запропонована схема синтезу коду програм може бути впроваджена в компіляторах взагалі для ЦЕОМ з довільною архітектурою.

Список літератури

1. Салапатов В. І. Структура данных для описания семантики и синтаксиса команд целевой ЭВМ. Вісник національного технічного університету України «КПІ», Інформатика, управління та обчислювальна техніка. № 41. Київ. 2004 с. 191-198.
2. Машинно-зависимая оптимизация кодов программ при их синтезе. ж. Электронное моделирование. № 2003. с. 33-44.
3. Салапатов В. І. Використання реляційної моделі даних для внутрішнього уявлення програми. Вісник національного технічного університету України «КПІ», Інформатика, управління та обчислювальна техніка. № 42. Київ. 2004 с. 191-198.
4. Салапатов В. І. Подання даних для синтезу коду у нормалізованому вигляді. Вісник Черкаського інженерно-технологічного інституту, 2001, №3, с.96-99.
5. Салапатов В. І. Особливості підвищення якості програм у сигнальних процесорів. Вісник національного технічного університету України «КПІ», Інформатика, управління та обчислювальна техніка. № 52. Київ. 2010. с. 75-79.

УДК 004.94:519.876.5

СТЕЦЕНКО І.В.

ПЕТРІ-ОБ'ЄКТНА МОДЕЛЬ СИСТЕМИ УПРАВЛІННЯ ТРАНСПОРТНИМ РУХОМ

В статье рассматривается имитационное моделирование системы управления транспортным движением, основывающееся на петри-объектной технологии. Динамика петри-объектов системы описывается стохастической временной сетью петри с конфликтными и многоканальными переходами, что позволяет воссоздать различные виды управления транспортным движением.

The article considers the simulation of traffic control's system based on Petri-object modeling. The dynamics of Petri-objects of the system are described by stochastic timed Petri net with conflict and multichannel transitions that allows to recreating various types of traffic control.

Вступ

Розрахунок параметрів діючих в Україні систем управління дорожнім рухом здійснюється виключно на підставі забезпечення безпеки руху та не передбачає швидке реагування на змінювання потоку транспорту [1]. Розвиток сучасних технологій відеоспостереження дорожнього руху та технічних засобів регулювання дорожнім рухом відкриває можливість розвитку технологій управління дорожнім рухом на перехресті, що ґрунтуються на поточних даних про показники дорожнього руху. Такі технічні засоби, як датчики дорожнього руху, керовані дорожні знаки та світлофорні об'єкти, що управляються центральним пунктом, складають достатню базу для створення інформаційно-аналітичної системи управління транспортним рухом на перехресті, яка б враховувала поточний стан дорожнього руху та здійснювала оперативне реагування на погіршення умов дорожнього руху.

Існуючі моделі транспортного руху не враховують встановлених засобів регулювання руху у місцях перетину доріг [2]. Важливість розробки деталізованої імітаційної моделі дорожнього руху підкреслюється в роботі [3]. В [4] запропонована формальна модель нерегульованого перехрестя. В [5] розглядається алгоритм відтворення руху авто по дорозі з урахуванням маневрів обгону та зміни смуги руху.

Складність системи обумовлює використання підходу, який поєднує в собі об'єктно-орієнтовану технологію та технологію імітаційного моделювання мережами Петрі, і отримав назву Петрі-об'єктного моделювання [6].

Постановка задачі моделювання

Модель системи управління транспортним рухом виходить з таких припущень: 1) відомі середні значення інтенсивностей надходження авто у визначених вхідних точках транспортної системи; 2) відомі значення середньої швидкості руху та довжина шляху між двома сусідніми перехрестями; 3) для кожного перехрестя відома кількість вхідних та вихідних напрямків руху, кількість смуг руху у кожному напрямку та ймовірності слідування авто, що надійшло у визначеному вхідному напрямку, до кожного з вихідних напрямків перехрестя; 4) для кожного перехрестя відомі засоби регулювання, які можуть бути використані на даному перехресті з огляду на безпеку руху і встановлення яких досліджується; 5) для перехрестя, регульованого світлофорами, відомі інтервали горіння жовтого сигналу світлофора, розрахованого з урахуванням структури перехрестя та вимог безпеки руху автомобілів, які в'їхали на перехрестя під час горіння зеленого сигналу світлофору.

Петрі-об'єктна модель системи управління транспортним рухом, яка розроблена, не враховує такі фактори: погодні умови; стан дорожнього покриття; рух пішоходів; рух автомобілів по відношенню один до одного; індивідуальні характеристики учасників автомобільного руху.

Транспортна система міста розглядається як сукупність перехресть, з'єднаних між собою дорогами (рис.1). Авто, що під'їжджають до перехрестя, рухаються по смугах руху до вхідних точок перехрестя. Переїзд перехрестя автомобілем розглядається як подія, при якій авто потрапляє з вхідної точки перехрестя до вихідної точки перехрестя. Кожна вхідна точка перехрестя характеризується однаковими правилами руху, які задаються розміткою проїзної частини, встановленими дорожніми знаками та

світлофорами. Авто, що знаходиться у вхідній точці перехрестя з заданою ймовірністю слідування прямує до вихідної точки перехрестя.. Ця ймовірність дорівнює нулю, якщо рух заборонений. Автомобіль, що потрапив у вихідну точку перехрестя, слідує по дорозі до наступного перехрестя, якщо є дорога, або вважається таким, що залишив транспортну систему. Якщо дорога переповнена (має місце транспортний затор), то авто не може потрапити у вхідну точку дороги через відсутність місця на ній. Потрапивши у вихідну точку дороги, авто опиняється у вхідній точці іншого перехрестя і т.д. доки авто не залишить транспортну систему або закінчився час моделювання.

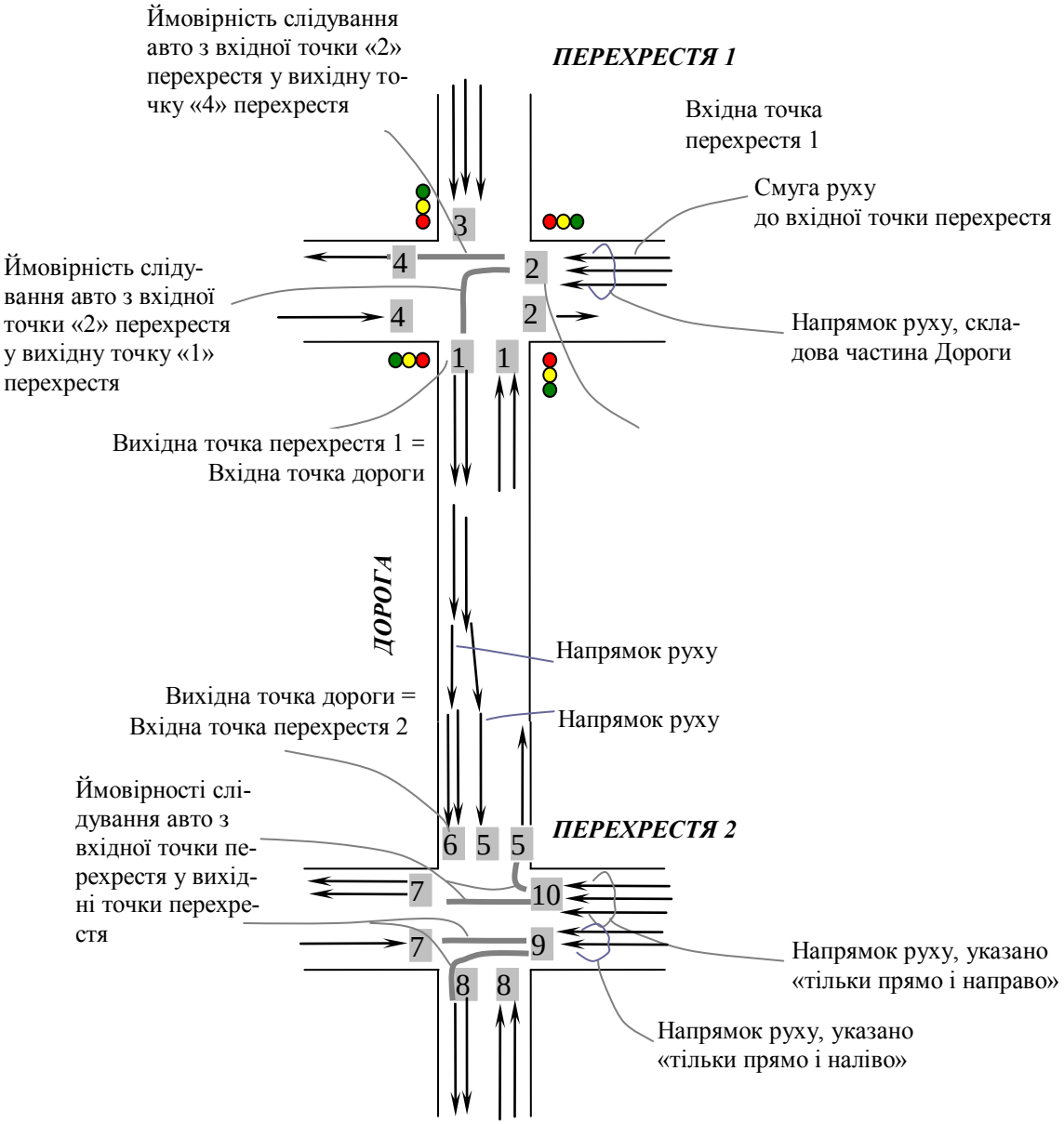


Рис. 1. Схема фрагмента транспортної системи, що включає два перехрестя і дорогу, що їх з'єднує

Технічними засобами регулювання на перехресті є дорожні знаки, які визначають перевагу того чи іншого напрямку руху, та світлофори, які сигналізують про надання дозволу на рух транспорту в тому чи іншому напрямку руху перехрестя. Алгоритм управління світлофорними об'єктами технічно реалізується дорожніми контролерами, що переключають сигнали світлофорів циклічно по заданій програмі.

Якість системи управління транспортним рухом оцінюється показниками, що вимірюють накопичення автомобілів у місцях перетину доріг. В конспекті [1] використовується середній час очікування транспортної одиниці. Але така величина не завжди правильно характеризує

стан дорожнього руху. Нехай, наприклад, на перехресті в одних напрямках спостерігається значне очікування в чергах, а в інших – спостерігається нульове. Середнє значення очікування, за рахунок нульових значень, приймає невелике значення. Чи може таке управління вважатись якісним? Очевидно, що обмеживши рух транспорту в тих напрямках, на яких спостерігається нульове значення, отримаємо додатковий резерв для зменшення очікування у більш завантажених напрямках. Отже, середнє очікування, а також середня кількість авто у чергах не можуть представляти оцінку якості управління.

В [7] запропоновано для оцінки якості управління транспортним рухом використовувати максимальну кількість автомобілів, які знаходяться в очікуванні:

$$z = \max(\ L_{11}, L_{12}, L_{13}, ... L_{kn}) \to \min\ , \qquad (1)$$

де L_{ji} – середня кількість машин, що очікують переїзду на *j*-ому перехресті в *i*-ому напрямку.

У правилах дорожнього руху вказуються дорожні перехрестя кількох типів: Т-подібне трьохстороннє, хрестоподібне чотирьох-стороннє, Х-подібне чотирьохстороннє, У-подібне трьохстороннє, багатостороннє перехрестя, площа, кільце. В [7] показано, що з хрестоподібних перехресть можна конструювати усі інші складні перехрестя, якщо розглядати напрямки руху як відкриті, так і закриті. В [8] розроблена технологія відшукання оптимальних параметрів управління дорожнього руху через регульовані перехрестя міста, що базується на імітаційній моделі та еволюційних методах оптимізації.

Об'єктно-орієнтований аналіз транспортної системи

Транспортний рух у місцях перетину доріг здійснюється за правилами дорожнього руху (розділ 16 правил дорожнього руху) в залежності від встановлених на перехресті засобів регулювання: авто здійснює проїзд регульованого перехрестя, якщо встановлений світлофорний об'єкт, здійснює проїзд нерегульованого перехрестя нерівнозначних доріг, якщо встановлені знаки дорожнього руху „Головна дорога" та „Уступи дорогу", які визначають перевагу напрямку руху, та проїзд нерегульованого перехрестя рівнозначних доріг, якщо не визначена перевага будь-якого з напрямків руху.

Функціонування світлофорного об'єкта задається послідовністю фаз. Кожна фаза складається з основного та проміжного тактів. Під час основного такту здійснюється рух транспорту на перехресті на зелений сигнал світлофора. Проміжний такт необхідний для забезпечення безпеки руху автомобілів та пішоходів і розраховується виключно з умов безпеки за формулами, які наведені в [1]. Сигнали світлофорів перемикаються дорожніми контролерами циклічно за заданою програмою. У табл. 1 представлена схема переключень сигналів *N*-фазного світлофорного циклу, що задає сигнали для *M* світлофорів відповідно до *M* вхідних точок транспортного руху. Такий підхід достатньо гнучкий, щоб відтворити усі можливі комбінації світлофорних сигналів. Якщо, наприклад, встановлений світлофор зі стрілкою, то потрібно створити додаткову вхідну точку для авто, які слідують у напрямку руху стрілки, і додати відповідний рядок у таблицю переключень сигналів світлофорів.

Табл. 1. N-фазний світлофорний цикл

Фаза/ Вхідна точка перехрестя	Фаза 1		Фаза 2		…		Фаза *N*	
	Основний такт	Проміжний такт	Основний такт	Проміжний такт	…	…	Основний такт	Проміжний такт
1	зелений	жовтий	червоний	жовтий	…	…	червоний	червоний і жовтий
2	зелений	жовтий	червоний	жовтий	…	…	червоний	жовтий
3	червоний	жовтий	зелений	жовтий	…	…	зелений	жовтий
4	червоний	жовтий	зелений	жовтий	…	…	зелений	жовтий
…	…	…	…	…	…	…	…	…
M	червоний	червоний і жовтий	зелений	зелений мигаючий			жовтий	жовтий мигаючий
Тривалість	t_1	t_2	t_3	t_3			t_{2N-1}	t_{2N}

Таким чином, управління світлофорним об'єктом характеризують такі величини: тривалість світлофорного циклу, кількість фаз у світлофорному циклі, кількість світлофорів, сигнали світлофорів відповідно до кожного такту і кожної фази світлофорного циклу.

Якщо світлофорний об'єкт на перехресті не встановлений, то регулювання рухом може

здійснюватись знаками дорожнього руху „Головна дорога” та „Уступи дорогу”. Встановлення знаку „Головна дорога” означає, що указаний напрямок руху має перевагу перед іншими напрямками руху перехрестя. Встановлення знаку „Уступи дорогу”, навпаки, означає, що указаний напрямок руху не має ніяких переваг перед іншими напрямками руху перехрестя.

Якщо на перехресті не встановлені світлофорний об’єкт чи знаки дорожнього руху, що указують на перевагу тих чи інших напрямків руху, то це перехрестя рівнозначних доріг. Регулювання рухом на такому перехресті здійснюється за правилом дорожнього руху (п. 16.12 Правил дорожнього руху), що транспорт, який наближається праворуч, має перевагу. Будемо називати таке правило „Пропустити авто праворуч”.

Регулювання транспортного руху за правилом „Пропустити авто праворуч” та регулювання дорожніми знаками мають спільну рису – вони встановлюють перевагу того чи іншого напрямку руху автомобілів перед іншими. Але дорожні знаки „Головна дорога” та „Уступи дорогу” встановлюють перевагу напрямку незалежно від поточного стану транспортного руху на перехресті, тобто безумовну перевагу напрямку руху. А правило „Пропустити авто праворуч” встановлює перевагу напрямку в залежності від наявності транспорту у напрямку справа, тобто в залежності від поточного стану транспортного руху на перехресті.

Регулювання транспортним рухом на перехресті світлофорним об’єктом та дорожніми знаками „Головна дорога” та „Уступи дорогу” теж мають спільну рису – вони встановлюють безумовну перевагу руху. Але дорожні знаки встановлюють перевагу руху тільки для транспорту, що рухається. Наприклад, якщо на головній дорозі не має транспорту, що рухається, то головна дорога втрачає свою перевагу і транспорт, який рухається в інших напрямках, набуває дозвіл для здійснення руху. А регулювання світлофорним об’єктом встановлює перевагу того чи іншого напрямку руху незалежно від наявності транспорту на ньому. Дійсно, червоний сигнал світлофора забороняє рух транспорту, навіть, якщо у напрямку з зеленим сигналом світлофора не рухається транспорт.

З порівняння засобів регулювання слідує, що світлофорне регулювання та регулювання іншими засобами суттєво відрізняються: перше не залежить від наявності автомобілів у тому чи іншому напрямку перехрестя. Ця різниця обумовлює представлення різних видів регулювання в моделі різними об’єктами.

Структура моделі системи управління транспортним рухом представляється такими об’єктами: Автомобіль, Дорога, Перехрестя, Світлофорне регулювання, Регулювання (дорожні знаки). Стан транспортної системи визначається кількістю транспортних одиниць, що накопичились у місцях перетину доріг, тобто на перехрестях. Об’єкт Перехрестя складається з вхідних та вихідних точок транспортного руху і визначає, з яких вхідних точок можна потрапити у які вихідні точки. Дорога – це ділянка транспортного руху між двома сусідніми перехрестями. Об’єкт Дорога містить множину напрямків руху. Початок напрямку руху знаходиться у вихідній точці одного перехрестя, а кінець – у вхідній точці іншого перехрестя.

Об’єкт Автомобіль представляє середньостатистичний об’єкт транспортного руху, який описується параметрами такими, як номер автомобіля, тип автомобіля, поточне розташування. Середня швидкість руху середньостатистичного автомобіля – це властивість дороги, на якій знаходиться автомобіль. А ймовірності слідування середньостатистичного автомобіля уздовж різних напрямків перехрестя – це властивість перехрестя.

Об’єкти Дорога, Перехрестя, Автомобіль є статичними і відтворюють тільки структурні властивості елементів транспортного руху. Автомобіль, хоч і змінює своє місце розташування в часі, але не містить власного опису правил, за якими змінюється його розташування. Змінювання місця розташування авто залежить від структури та засобів регулювання перехрестя, що переїжджає авто, або від властивостей дороги, уздовж якої їде. Тому автомобіль розглядається як статичний об’єкт.

Динамічними об’єктами моделі є Вхідна точка перехрестя, Вихідна точка перехрестя, Світлофорне регулювання, Регулювання (дорожні знаки), Генератор автомобілів. Динамічні об’єкти представляються Петрі-об’єктами і створюються з використанням конструктору класу Петрі-об’єкт (PetriSim), що передбачає обов’язкову передачу мережі Петрі, яка представляє динаміку функціонування об’єктів [2] у відповідне поле об’єкта.

Діаграма класів Петрі-об’єктної моделі системи представлена на рис. 2. Класи Вхідна точка перехрестя (CrossPointIn), Вихідна точка перехрестя (CrossPointOut), Світлофорне регулювання (TrafficLight), Регулювання

(TrafficControl), Генератор авто (GeneratorCars) наслідують клас Петрі-імітатор (PetriSim). Класи Вхідна точка перехрестя (CrossPointIn) та Вихідна точка перехрестя (CrossPointOut) слугують для створення класів Точка перехрестя (CrossPoint) та Напрямок руху (Direction). Класи Точка перехрестя (CrossPoint) та Напрямок руху (Direction), в свою чергу, слугують для створення класів Перехрестя (CrossRoad) та Дорога (Road). Клас Перехрестя (CrossRoad) передбачає використання об’єктів класу Світлофорне регулювання (TrafficLight) для створення своїх екземплярів. А клас Регулювання (TrafficControl) використовує клас Перехрестя (CrossRoad) для встановлення переваги того чи іншого напрямку руху.

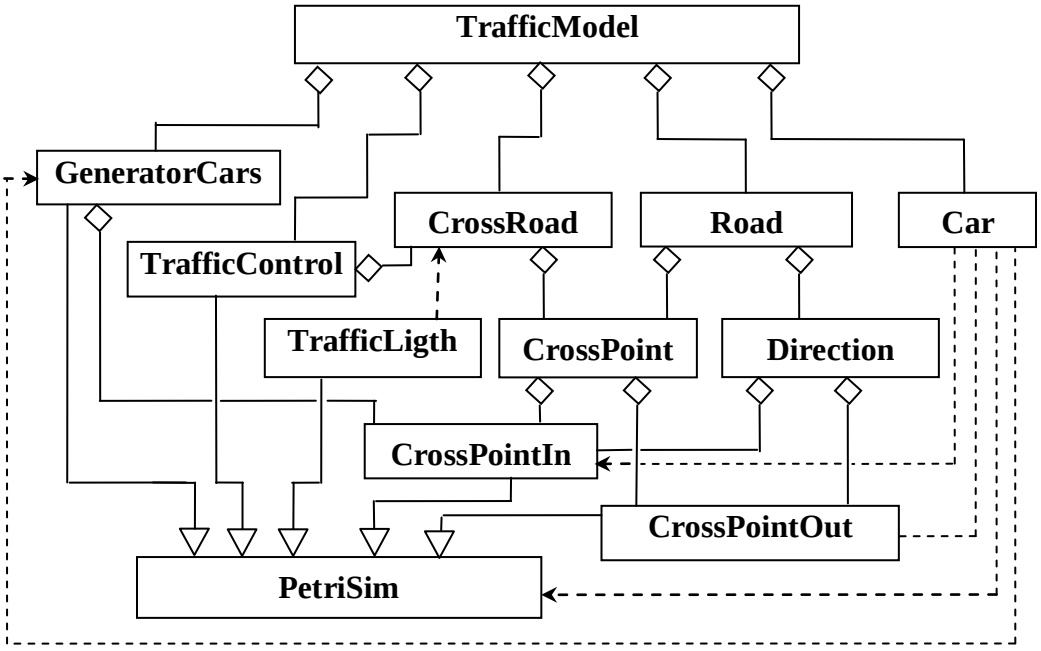


Рис. 2. Діаграма класів Петрі-об’єктної моделі системи управління транспортним

Регулювання руху з використанням дорожніх знаків „Головна дорога” та „Уступи дорогу” або правила „Пропустити авто праворуч” спирається на поточний стан автомобільного руху на перехресті. Звідси слідує, що опис класу Регулювання (TrafficControl) містить поле, що указує на екземпляр класу Перехрестя (Cross Road), для якого створюється регулювання.

Світлофорне регулювання містить динаміку, що визначає переключення сигналів світлофора незалежно від стану автомобільного руху. Тому клас Світлофорне регулювання (TrafficLight) створюється незалежно від класу Перехрестя (CrossRoad). Але клас Перехрестя (CrossRoad) залежить від класу Світлофорне регулювання (TrafficLight), оскільки автомобільний рух на перехресті, керованому світлофорним об’єктом, має враховувати поточні сигнали світлофорів у всіх своїх напрямках.

Клас Автомобіль (Car) використовується для формування списку автомобілів, що являються учасниками транспортного руху, основним класом моделі - класом Транспортна Модель (TrafficModel). Інформація про автомобіль використовується методами Петрі-об’єктів Вхідна точка перехрестя (CrossPointIn), Вихідна точка перехрестя (CrossPointOut) та Генератор авто (GeneratorCars) для визначення та змінювання розташування автомобіля.

Побудова Петрі-об’єктної моделі системи управління транспортним рухом

Складемо мережі Петрі-об’єктів моделі транспортного руху, керуючись технологією формалізованого представлення моделі системи стохастичною мережею Петрі з часовими затримками, викладеною в [8].

Мережа Петрі-об’єкта Генератор автомобілів представлена на рис. 3 і складається з однієї події „Генерувати авто”, що відбувається з інтервалом часу, заданим випадковим числом.

Мережа Петрі-об’єкта Вихідна точка перехрестя представлена на рис. 4 і враховує час переїзду, який залежить від довжини дороги та швидкості руху середньостатистичного автомобіля, та максимальну кількість автомобілів, що одночасно можуть здійснювати переїзд. Максимальна кількість автомобілів визначається кількістю смуг руху та довжиною середньостатистичного автомобіля.

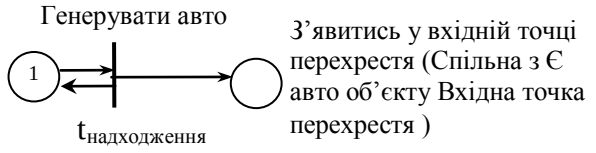


Рис. 3. Мережа Петрі-об'єкта Генератор автомобілів

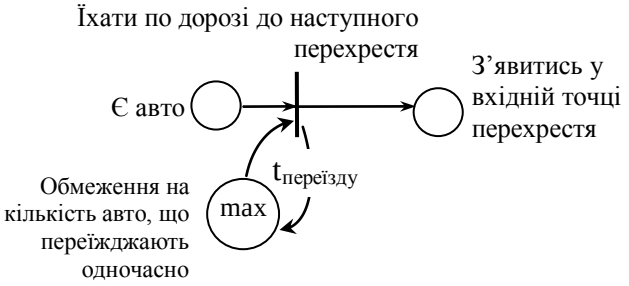


Рис. 4. Мережа Петрі-об'єкта Вихідна точка перехрестя

Мережа Петрі-об'єкта Вхідна точка перехрестя визначається переліком наступних подій: з'явилось авто у вхідній точці; авто має дозвіл на переїзд перехрестя; авто слідує до вихідної точки *j* перехрестя *j*=1,...*M*; переїзд до вихідної точки *j* перехрестя, *j*=1,...*M* (рис. 5).

Подія „З'явилось авто у вхідній точці" слугує для змінювання поточного стану об'єкту Вхідна точка перехрестя, що запам'ятовується у позиціях „Загальна кількість авто у вхідній точці перехрестя" та „Максимальна кількість авто у вхідній точці перехрестя". Значення маркірування у цих позиціях використовується об'єктом Регулювання. Подія „Авто має дозвіл на переїзд перехрестя" здійснюється за умови, що є автомобіль у вхідній точці перехрестя і є дозвіл на здійснення руху. Якщо перехрестя оснащене світлофорним об'єктом, то позиція, що відповідає за надання дозволу на рух, є спільною з об'єктом Світлофорне регулювання і маркер в позиції з'являється у відповідності до сигналів світлофорного об'єкта. Якщо перехрестя нерегульоване світлофорами, то маркер у позиції „Дозвіл на переїзд" з'являється в результаті функціонування об'єкту Регулювання.

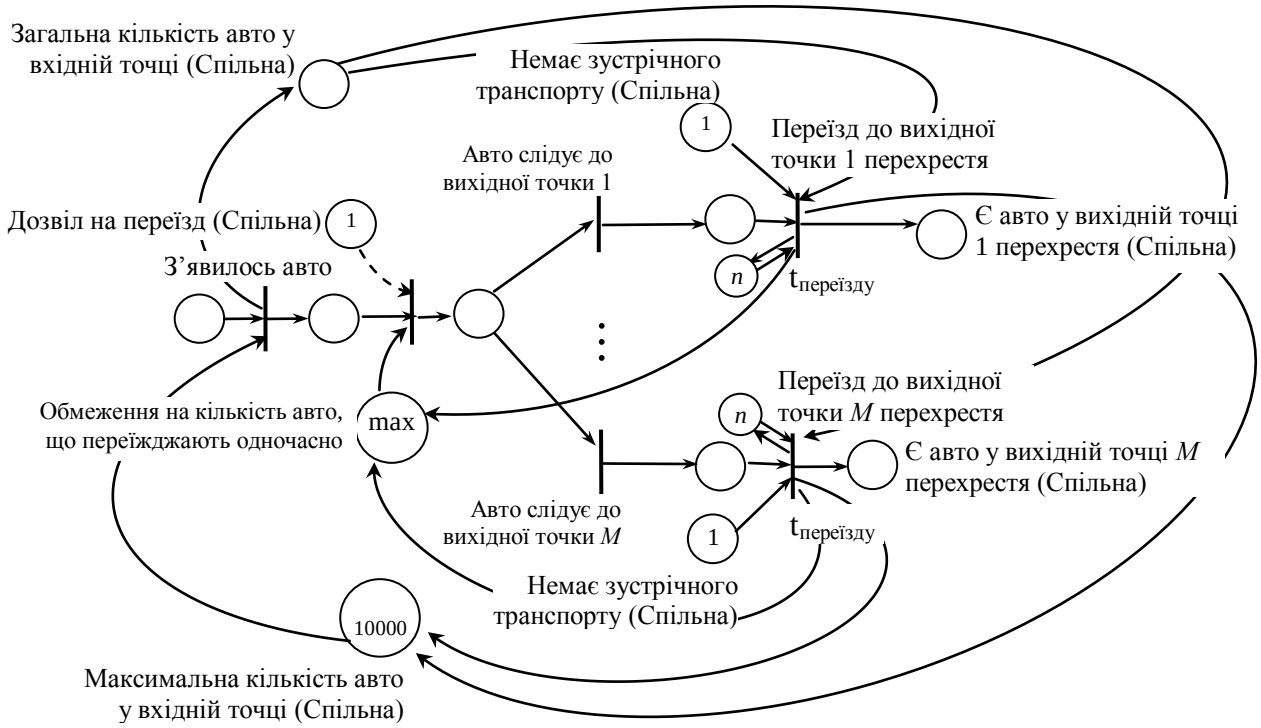


Рис. 5. Мережа Петрі-об'єкта Вхідна точка перехрестя

Авто, яке має дозвіл на переїзд перехрестя, відправляється у відповідності до маршруту свого слідування в одну з вихідних точок перехрестя. Переходи „Авто слідує до вихідної точки *j* перехрестя" є конфліктними. Конфлікт розв'язується випадковим вибором переходу у відповідності до заданих ймовірностей запуску переходів.

Траєкторії руху авто, які отримали дозвіл на переїзд з різних вихідних точок перехрестя, можуть перетинатись. Позиція „Немає зустрічного транспорту", яка є спільною для відповідних об'єктів Вхідні точки перехрестя, надає до-

звіл на переїзд одному з авто, що рухаються по зустрічних траєкторіях руху.

Переходи „Переїзд до вихідної точки *j* перехрестя" мають часову затримку, рівну тривалості переїзду перехрестя у відповідному напрямку руху.

Мережа Петрі-об'єкта Світлофорне регулювання для випадку *N*-фазного світлофорного об'єкта представлена на рис. 6. Чергування фаз (див. табл.1) утворює коло подій з відповідними переключеннями сигналів світлофорів. Тривалості фаз задаються при конструюванні мережі Петрі. Маркери в позиціях, які сигналізують про завершення того чи іншого такту, відразу зникають, оскільки являються умовою запуску наступного переходу кола подій. Тому створені позиції „Є зелене світло для авто у вхідних точках *i, j*,...", маркери в яких з'являються на початку основного такту фази і зникають на початку проміжного такту фази. Аналогічно можуть бути створені позиції для будь-якого сигналу світлофору, якщо вони використовуються іншими Петрі-об'єктами.

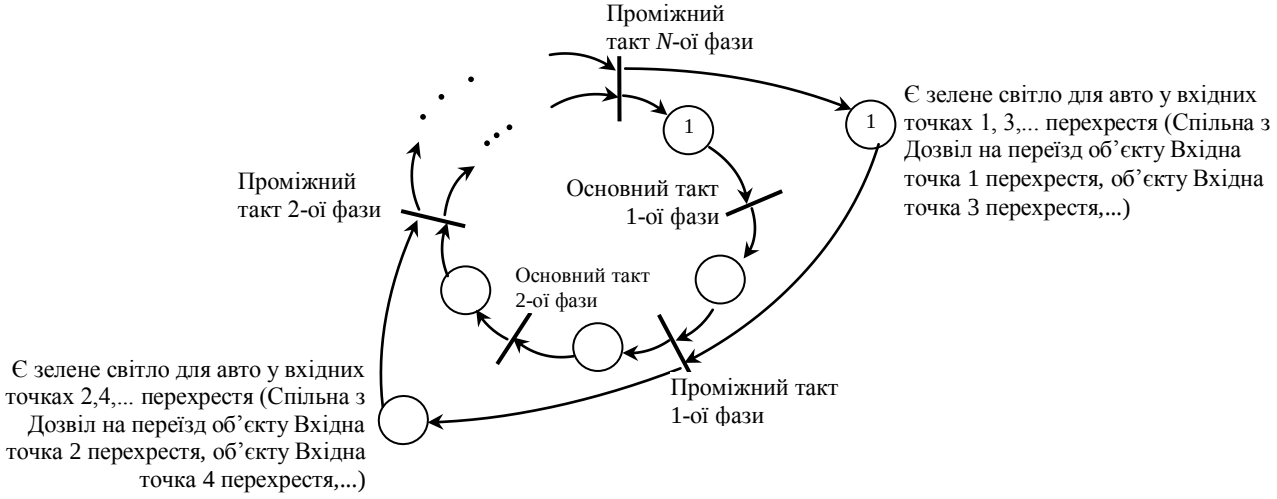


Рис. 6. Мережа Петрі-об'єкта Світлофорне регулювання

Регулювання за допомогою знаків дорожнього руху чи правила «Пропустити авто праворуч» спирається на інформацію про наявність авто у вхідних точках перехрестя, що надходить з позицій «Загальна кількість авто у напрямку *j*», «Максимальна кількість авто у напрямку *j*». Позиція «Загальна кількість авто у напрямку *j*» використовується для з'ясовування, чи є авто у *j*-напрямку руху, а позиція «Максимальна кількість авто у напрямку *j*» – для з'ясовування нульової кількості авто у *j*-напрямку руху на перехресті. Рух у головних напрямках перехрестя відновлюється за умови, що з'являється хоч одне авто в будь-якому з головних напрямків руху. А дозвіл на рух у неголовних напрямках перехрестя з'являється за умови, що немає жодного авто для руху у будь-якому з головних напрямків руху. На рис. 7 наведений приклад мережі Петрі-об'єкта Регулювання для випадку, коли дорожні знаки «Головна дорога» встановлені у напрямках 1 і 3.

Регулювання знаками дорожнього руху «Уступи дорогу» еквівалентно встановленню знаків «Головна дорога» для всіх інших напрямків перехрестя, для яких не встановлені знаки «Уступи дорогу».

Регулювання за правилом «Пропустити авто праворуч» використовується для перехресть, на яких не встановлені дорожні знаки, що визначають перевагу того чи іншого напрямку руху перехрестя, і передбачає, що напрямок руху праворуч від авто, яке надходить до перехрестя, має більшу перевагу за напрямок руху, в якому надходить авто. При моделюванні цього способу регулювання важливо враховувати усі умови, при яких дозвіл на рух виникає, та умови, при яких дозвіл на рух зникає. Авто, яке надходить, спричиняє подію «Надати дозвіл на рух у напрямку *j*» тільки, якщо у цьому напрямку наявні авто, а у напрямку праворуч спостерігається нульова кількість авто і, водночас, наявність дозволу на здійснення руху. Дійсно, якщо не має авто у напрямку руху, то не потрібно надавати дозвіл на рух. Одночасне виконання умов «спостерігається нульова кількість авто» та «спостерігається наявність дозволу на здійснення руху у напрямку праворуч» означає, що в поточний момент часу спостерігається припинення руху авто у напрямку праворуч. Виконання події «Надати дозвіл на рух у напрямку *j*» спричиняє виконання умови «є дозвіл на рух

у напрямку *j*» та виконання умови «не має дозволу на рух у напрямку ліворуч».

Іншими словами, дозвіл на рух у напрямку *j* з'являється, якщо у напрямку праворуч рух припиняється, а зникає, коли припиняється рух авто у напрямку *j* і надійшло авто у напрямку ліворуч. Таким чином, отримуємо мережу Петрі-об'єкта Регулювання, приклад якої у випадку руху за правилом «Пропустити авто праворуч» на чотирьохсторонньому хрестоподібному перехресті наведений на рис. 8.

Конструювання мережі Петрі-об'єкта Регулювання здійснюється для визначеного об'єкта Перехрестя і має з ним спільні позиції „Загальна кількість авто у напрямку *j*", „Максимальна кількість авто у напрямку *j*" та „Є дозвіл на рух у напрямку *j*".

Зв'язки між Петрі-об'єктами Вхідна точка перехрестя, Вихідна точка перехрестя, Світлофор, Генератор автомобілів представлені на рис. 9 і формуються виключно за допомогою спільних позицій.

Модель системи управління транспортним рухом реалізована засобами мови програмування Java (J2SE) та інтегрованого середовища Netbeans IDE 6.5. За результатами імітаційного моделювання спостерігається кількість авто, що очікують переїзду в кожному з об'єктів Вхідна точка перехрестя та розраховується критерій якості параметрів управління (1).

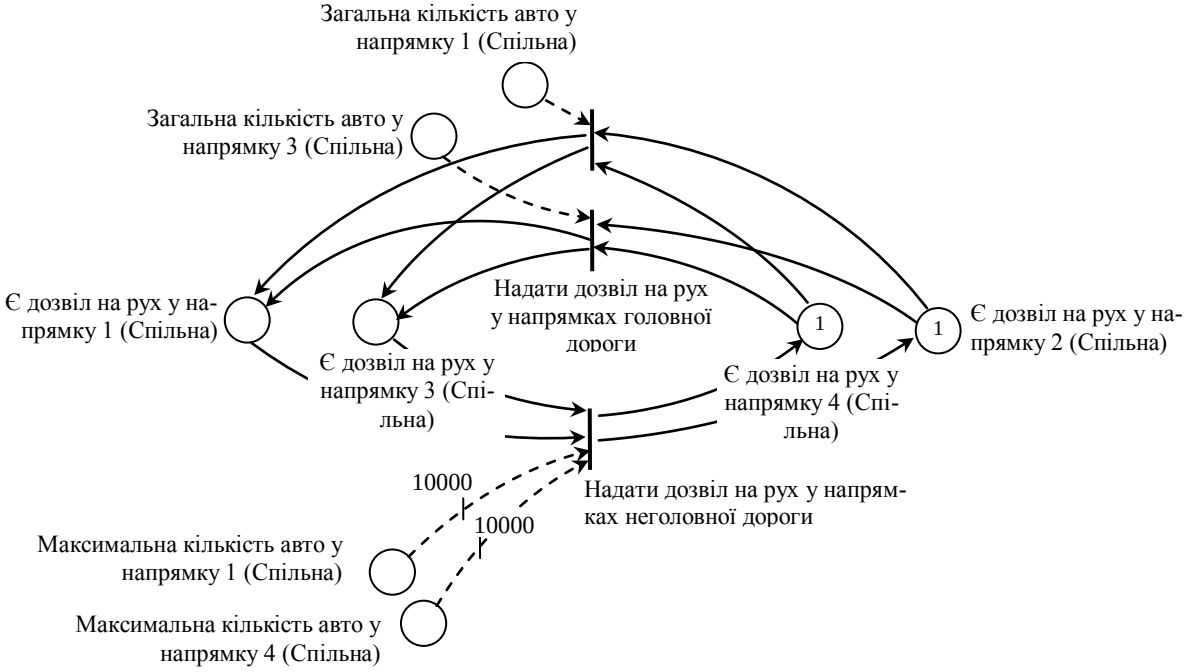


Рис. 7. Мережа Петрі-об'єкта Регулювання за знаками дорожнього руху «Головна дорога»

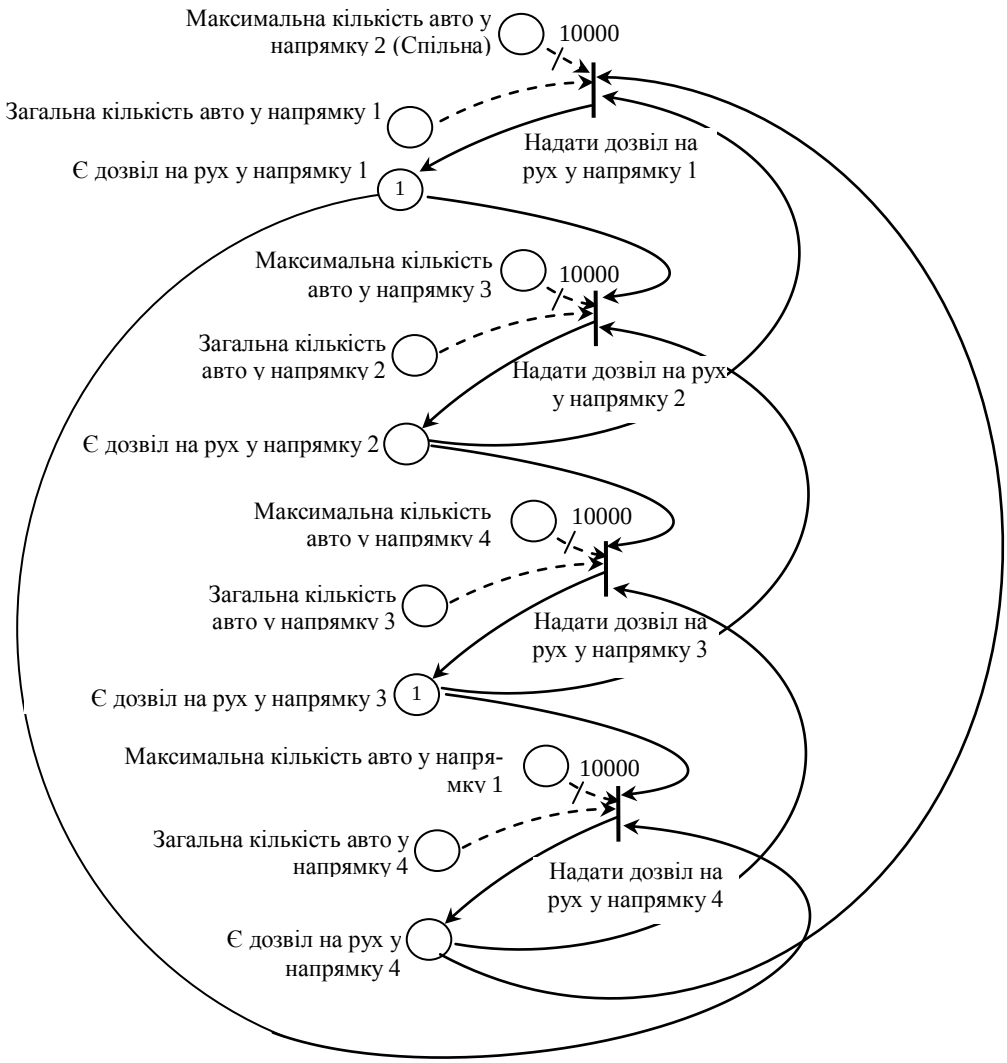


Рис. 8. Мережа Петрі-об’єкта Регулювання у випадку регулювання за правилом «Пропустити авто праворуч»

Висновки

Застосування Петрі-об’єктної технології моделювання до моделі системи управління дорожнім рухом дозволило вперше побудувати імітаційну модель системи з урахуванням різних способів регулювання, що використовуються в транспортних системах, та з урахуванням сумісного впливу управляючих параметрів. Петрі-об’єкти моделі системи управління транспортним рухом, які розроблені, можуть стати підґрунтям для розробки предметно-орієнтованої системи імітаційного моделювання транспортних систем.

Оптимізація параметрів управління за критерієм якості (1) з використанням еволюційної стратегії, описаної в [8], відкриває перспективу розробок у напрямку створення систем адаптивного управління транспортними потоками міста.

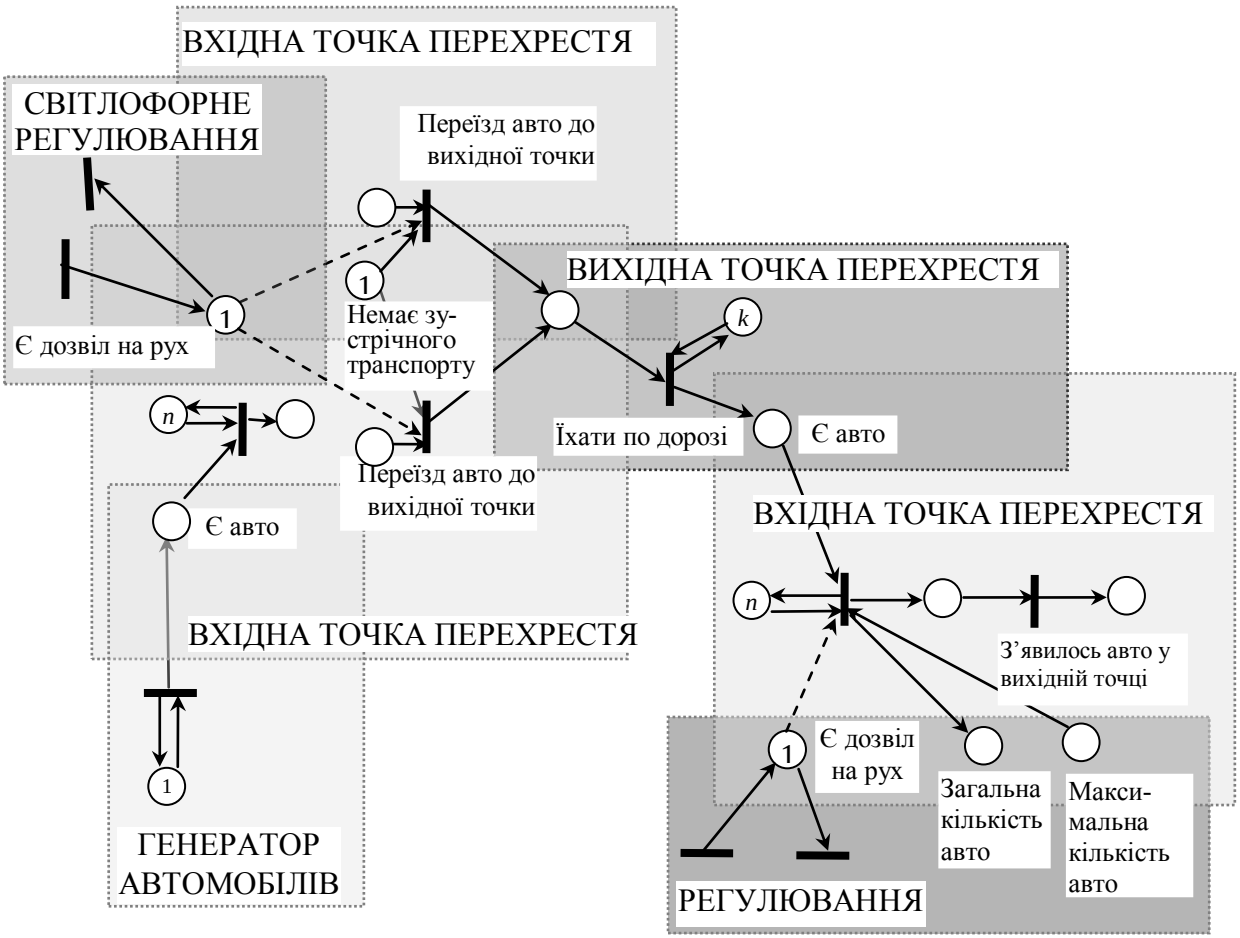


Рис. 9. Зв'язки між Петрі-об'єктами Вхідна точка перехрестя, Вихідна точка перехрестя, Світлофорне регулювання, Регулювання, Генератор автомобілів

Список літератури

1. Віниченко В.С. Конспект лекцій з дисципліни «Автоматизовані системи управління на транспорті» (для студентів 4 курсу всіх форм навчання напряму підготовки 1004 «Транспортні технології»). – Харків: ХНАМГ, 2007. – 68 с.
2. Інформаційне забезпечення для моделювання та керування транспортними потоками у великих містах. Автореф. дис. канд. техн. наук: 05.13.06 / М.А. Григоров; Одес. нац. політехн. ун-т. – О., 2005. – 18 с.
3. Томашевський В.М. Концептуальні основи імітаційного моделювання автомобільного дорожнього руху / Томашевський В.М., Печенежський Д.С. // Праці П'ятої Української конференції з автоматичного управління "Автоматика-98", – ч. III – Київ: НТТУ "КПІ ", 1998. С. 317 - 323.
4. Парамонов А.М., Томашевский В.М. Определение формальной модели нерегулируемого перекрестка в системе моделирования автомобильного дорожного движения // Нові технології. Науковий вісник КУЕІТУ. – Кременчук: КУЕІТУ, 2009. – №1(23).– С.135-138.
5. Парамонов А.М., Томашевский В.М. Формализация алгоритма моделирования движения автомобильного дорожного транспорта // Вісник НТУУ «КПІ». Інформатика, управління, обчислювальна техніка. – Київ, 2008. - №48. – С.7-12.
6. Стеценко И.В. Формальное описание систем средствами Петри-объектных моделей // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: ВЕК+, 2011. – № 53. – С.74-81.
7. Стеценко І.В. Імітаційне моделювання транспортного руху через світлофорні об'єкти / Стеценко І.В., Батора Ю.В. // Вісник Черкаського державного технологічного університету. – Черкаси, 2006. – №3. – С.75-79.
8. Стеценко І.В. Інформаційна технологія визначення оптимальних параметрів управління транспортним рухом через світлофорні об'єкти міста / Стеценко І.В., Батора Ю.В. // Математичні машини і системи. – Київ, 2007. – №.3,4 – С.211-217.
9. Стеценко І.В. Моделювання управляючих систем засобами мереж Петрі з інформаційними зв'язками // Вісник Черкаського державного технологічного університету. – Черкаси: ЧДТУ, 2011. – №3. – [В друку].

УДК 683.519

ФОМЕНКО А.О.

МЕТОДИКА РАСЧЕТА ПРОЕКТНЫХ РИСКОВ НА ОСНОВЕ ПРИМЕНЕНИЯ БАЙЕСОВСКИХ СЕТЕЙ

В статье описана методика расчета рисков на всех этапах жизненного цикла проекта на основе применения Байесовских сетей. Программная реализация данного метода осуществляется с помощью программного пакета Hugin.

The article describes the method of risks calculation on all stages of project life cycle on the basis of application of the Bayes networks. Programmatic realization of this method is carried out by the Hugin software package.

1. Постановка проблемы

Неотъемлемой частью создания качественного программного продукта является составление программного проекта. Анализ и управление проектными рисками становится общепризнанной методологией осуществления проектов и превратилось в неотъемлемую часть ведения бизнеса.

Основной целью управления программными проектами является достижение оптимального качества программного продукта при минимуме затрат, оценка и управление проектными рисками на всех этапах жизненного цикла разработки программного обеспечения.

Можно перечислить несколько преимуществ, которые анализ рисков дает менеджеру: более глубокое понимание специфики проекта, позволяющее создавать более реалистичные планы и бюджет проекта; понимание природы рисков и их потенциальных последствий дает возможность к их распределению между агентами, которые способны лучше ими управлять; возможность оценки резервов, создаваемых для обеспечения рисков, а это, в свою очередь, позволяет уменьшить вероятность срыва проекта [1].

Для успешной реализации перечисленных преимуществ необходимо грамотно уметь анализировать риски на качественном, содержательном уровне с целью их идентификации, выявления причин, оценки потенциальных последствий, разработке стратегии «реагирования» и т.д.

Поскольку сфера разработки программного обеспечения классифицируется как антропоцентрическая система, в которой центральным элементом является человек, то проектные риски неразрывно связаны с условиями его деятельности и влиянием на него факторов внешней и внутренней среды. Следовательно, пути снижения рисков лежат в изучении основных закономерностей влияния факторов среды на интеллектуальную деятельность специалистов, задействованных в разработке ПО [2].

2. Анализ исследований и публикаций

Решению данных проблем посвящено значительное количество научных исследований, выполненных в разное время отечественными и зарубежными учеными, среди которых работы В.В. Липаева, Г.Б. Мороза, А.Ф. Кулакова, С.А. Юдицкого, А.М. Вендрова, И. Соммервилла, М. Кантора, Д. Фокса, Д. Лефингвела, Г. Гласа, А. Шаллоуея и др. Стадии подготовки программных проектов, а также рисков, возникающих на данных этапах подробно описали Роберт Т. Фатрелл, Рассел Д. Арчибальд, Г. Дитхелм.

3. Изложение основного материала исследования

По данным статистики, представленной международной компанией Standish Group Chaos, из всех программных проектов, завершенных в 2010 году, только 32% программных проектов являются успешными, 44% являются спорными (имеющими перерасход средств, превышение бюджета, другие недостатки), а 24% являются провальными.

Данная ситуация обусловлена недостаточным вниманием оценке проектных рисков, а также проблемой выбора инструментов планирования.

В настоящее время на рынке представлено значительное количество универсальных программных пакетов для персональных компьютеров, автоматизирующих функции планирования и контроля календарного графика выполнения работ, а также оценки и анализа проектных рисков. Среди наиболее популярных можно привести следующие:

Microsoft Project (Microsoft), OpenProj, TurboProject (IMSI), Spider Project (Технологии управления Спайдер), Project Workbench (Applied Business Technology).

Однако, в современных средствах управления программным проектом задачи численной оценки рисков, обусловленных, например, человеческим фактором, не решаются, а их учет осуществляется на основе интуиции и здравого смысла менеджера [3].

Одним из методов комплексной оценки проектных рисков, является метод вероятностного моделирования с использованием байесовских сетей.

Байесовские сети (БС) представляют собой графовые модели вероятностных и причинно-следственных отношений между переменными в статистическом информационном моделировании. В БС могут органически сочетаться эмпирические частоты появления различных значений переменных, субъективные оценки «ожиданий» и теоретические представления о математических вероятностях тех или иных следствий из априорной информации. Это является важным практическим преимуществом и отличает байесовские сети от других методик оценки риска.

Понятие условной вероятности P(A|B) = x составляет основу байесовского подхода к анализу неопределенности. Совместная вероятности наступления событий A и B дается формулой полной вероятности:

$$P(A, B) = P(A \mid B) \cdot P(B) \qquad (1)$$

Имея в распоряжении информацию о зависимых переменных (следствия), можно определить сравнительные вероятности исходных переменных (причин) при помощи теоремы Байеса.

Пусть имеется условная вероятность P(A|B) наступления некоторого события A при условии, что наступило событие B. Теорема Байеса дает решение для обратной задачи – какова вероятность наступления более раннего события B, если известно, что более позднее событие A наступило.

Пусть A1,…An – полная группа несовместимых взаимоисключающих событий (альтернативных гипотез). Тогда апостериорная вероятность P(Aj|B) каждого из событий Aj при условии, что произошло событие B, выражается через априорную вероятность P(Aj):

$$P(A_j \mid B) = \frac{P(A_j) \cdot P(B \mid A_j)}{P(B) = \sum_{i=1}^{n} P(A_j) \cdot P(B \mid A_j)} \qquad (2)$$

Обратная вероятность P(B|Aj) называется правдоподобием, а знаменатель P(B) в формуле Байеса – свидетельством.

Совместная вероятность является наиболее полным статистическим описаним наблюдаемых данных. Совместное распределение представляется функцией многих переменных в задаче. В общем случае это описание требует задания вероятностей всех допустимых конфигураций значений всех переменных, что мало применимо в реальных задачах. В байесовских сетях, в условиях, когда имеется дополнительная информация о степени зависимости или независимости признаков, эта функция факторизуется на функции меньшего числа переменных:

$$P(A_i, \ldots A_n) = \prod_j P[A_j Pa(A_j)] \qquad (3)$$

где Pa(Aj) – состояния всех переменных-предков для переменной Aj. [4]

Исходя из ранее перечисленных особенностей БС, а так же используя (1), (2) и (3), можно эффективно использовать аппарат БС в задачах разработки модели риска и количественной оценки вероятности его возникновения и материализации, а также влияния на другие риски в проекте.

Основным достоинством использования байесовских сетей в задачах риск-менед жмента является возможность совместного учета количественных и качественных показателей, динамическая обработка поступающей информации, а также явные зависимости между причинами, влияющими на возникновение риска.

Алгоритм использования данного метода включает следующие шаги.

1. Проведение качественного анализа рисков с одновременной оценкой вероятности неблагоприятных событий. При этом необходимо определить степень влияния описываемого риска на проект.
2. Проведение анализа жизненного цикла разрабатываемого ресурса. Необходимо определить, на каких стадиях возникают риски, какие из задач выполняются параллельно, а какие последовательно, чтобы определить влияние одних рисков на другие.
3. Составление правил, которые описывают причинно-следственные связи хода выполнения проекта с учетом рисков.
4. Построение байесовской сети, и соответствующей характеристикам проекта.

5. Задание таблиц условных вероятностей для каждой из нелистьевых вершин байесовской сети.

6. Обучение байесовской сети, проверка адекватности модели.

Для составления БС и таблиц априорных вероятностей для каждого события используются данные, которые могут быть получены от экспертов в области или из опыта многократного выполнения подобных проектов.

Дальнейшее улучшение качества прогнозирования может быть достигнуто путем обучения Байесовской сети на имеющихся экспериментальных данных. Обучение традиционно разделяется на две составляющие – выбор эффективной топологии сети, включая, возможно, добавление новых узлов, соответствующих скрытым переменным, и настройка параметров условных распределений для значений переменных в узлах.

Для расчетов был использован программный пакет Hugin. Hugin являетсям программной реализацией системы принятия решений на основе байесовских сетей доверия. Hugin использует два основных режима работы:

1. Режим редактирования и построения причинно-следственной сети, а также заполнения таблиц условных вероятностей, являющихся количественным описанием БЗ.

2. Режим расчёта вероятностных оценок для принятия решения по всем событиям, входящим в причинно-следственную сеть. Расчёты могут осуществляться как на основе классической теории Байеса, так и на основе методов теории возможностей.

Построим БС, характеризующую поэтапную разработку проекта (рис. 1)

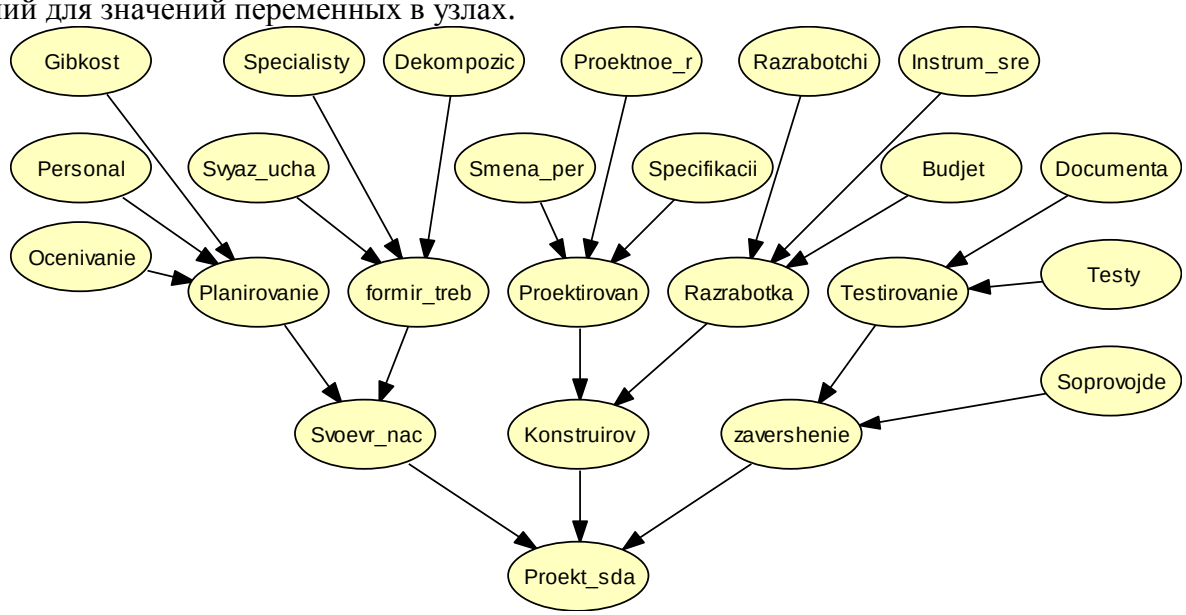


Рис. 1. Байесовская сеть

Зададим таблицы условных вероятностей БС, построенных для количественной оценки рисков на всех этапах разработки программного обеспечения (табл.1):

Табл. 1. Таблица условных вероятностей вершины «Проект сдан»

Вершина	**Состояние**							
Завершение работ	Успешно				Необходима доработка			
Конструирование	Успешно		Опоздание		Успешно		Опоздание	
Своевременная подготовка документации	Да	Нет	Да	Нет	Да	Нет	Да	Нет
Проект сдан в срок	0,99	0,85	0,8	0,65	0,85	0,75	0,7	0,01
Проект сдан с опозданием	0,01	0,15	0,2	0,35	0,15	0,25	0,3	0,99

Для того, чтобы дать вероятностную оценку риску необходимо использовать функцию распределения вероятностей. В окне состояний отобразится, что вероятность возникновения события «Проект сдан с опозданием» с учетом условных вероятностей причин равна 10,29%. Можно принять это как максимальную допустимую границу риска и сравнивать полученные результаты с ней. Если они не будут превышать максимальную допустимую границу, рекомендуется их принимать, иначе рекомендуется исследовать модель риска и внести в проект резервы времени на преодоление последствий материализации риска.

Для нахождения нижней границы значения искомого риска используется распространение аксимумов. Наименьшая вероятность

возникновения риска «Проект сдан с опозданием» составляет 2,13%.

На основе анализа факта срыва сроков проекта, можно сделать предположение, что с вероятностью 47% это произойдет из-за задержки на начальном этапе, с вероятностью 42% – на этапе конструирования, и с вероятностью 48%, что проблемы возникнут при тестировании проекта.

Для того, чтобы найти наиболее вероятную комбинацию состояний всех вершин, необходимо вновь использовать распространение максимумов. После пересчета получим новое распространение в окнах отображения сети и вершин. При этом каждое из состояний вершин, имеющее значение 100%, будет принадлежать к наиболее вероятной комбинации состояний. Для рассматриваемого примера будет получена одна уникальная комбинация, при которой наиболее вероятным является, что проект будет сорван тогда, когда событие «Завершение работ» находится в состоянии «Необходима доработка».

Рассчитаем вероятность наиболее вероятной комбинации состояний, полученных при наличии факта, что проект будет сдан с опозданием. Эта вероятность записывается в виде:

P (Своевременная подготовка документации = «успешно», Конструирование завершено = «успешно», Завершение работ = «необходима доработка» | Проект сдан = «с опозданием»)

и для ее определения можно воспользоваться формулой:

$$P(A, B) = P(A \mid B)\, P(B), \qquad (4)$$

переписав ее в виде:

$$P(A \mid B) = P(A, B) / P(B), \qquad (5)$$

Если выбрать состояние «с опозданием» для вершины Проект сдан и сделать расчет распространения сумм, то вероятность пребывания вершины в указанном состоянии будет равно 10,29%. Определив чему равно P(Проект сдан = «с опозданием»), необходимо определить чему равно P(Своевременная подготовка документации = «успешно», Конструирование завершено = «успешно», Завершение работ = «необходима доработка», Проект сдан = «с опозданием»). Это значение равно 4,9%. Теперь можно рассчитать искомую вероятность, которая будет равна:

$$P(A, B) / P(B) = 0.049 / 0.103 = 0.476 \qquad (6)$$

Таким образом, на основе поступившего факта срыва сроков проекта экспертная система делает вывод о том, что наиболее вероятной является ситуация, связанная с тем, что на завершающем этапе проект потребует доработки. При этом вероятность такой ситуации равна 0.476.

4. Выводы

В статье предложена модель оценки рисков на всех этапах создания программного обеспечения. Байесовские сети предоставляют удобный аппарат для исследования рисков, составления их модели и количественной оценки. Модель риска удобно представлять в виде ориентированного графа, это позволяет как можно точнее исследовать составляющие риска и включить максимальное количество факторов, влияющих на вероятность возникновения риска.

В байесовских сетях доверия есть возможность использовать как вероятности, полученные опытным путем, или из опыта многократного использования системы, так и экспертные оценки, что позволяет использовать в процессе составления модели риска знания эксперта-специалиста, выраженные в виде предположений.

Модель, построенная в виде байесовской сети доверия – динамична, она легко подстраивается под полученные свидетельства. Это позволяет вносить в систему данные, полученные на каждом этапе выполнения проекта. Усовершенствованные таким образом модели рисков могут использоваться в других аналогичных проектах.

Список литературы

1. Фатрелл Р. Управление программными проектами / Р. Фатрелл, Д. Шафер, Л. Шафер; пер. с англ. А. Бойко, А. И. Мороза, А. П. Сергеева. – М. : Издательский дом «Вильямс», 2003. – 1136 с.
2. Руководство по своду знаний по управлению проектами PMBOK / Project Management Institute, Inc., 2009. – 388 с.
3. Арчибальд Р. Управление высокотехнологичными программами и проектами / Р. Арчибальд; пер. с англ. Ю. П. Рогач, М. М. Сердюк. – М: ДМК-Пресс, 2002. – 460 с.
4. Тулупьев А. Л. Байесовские сети: Логико-вероятностный подход / А. Л. Тулупьев, С. И. Николенко, А. В. Сироткин – СПб. : Наука, 2006. — 607 с.

УДК 004.032.26; 004.272.86

ТУРЧЕНКО В.О.

ПОРІВНЯННЯ ЕФЕКТИВНОСТІ ГРУПОВОГО НАВЧАННЯ БАГАТОШАРОВОГО ПЕРСЕПТРОНУ НА ПАРАЛЕЛЬНОМУ КОМП'ЮТЕРІ ТА ОБЧИСЛЮВАЛЬНОМУ КЛАСТЕРІ

Паралельний метод групового навчання багатошарового персептрону (БШП) на основі алгоритму зворотного поширення помилки та дослідження ефективності розпаралелення цього методу на паралельному комп'ютері та обчислювальних кластерах представлено в цій статті. Модель БШП та послідовний метод групового навчання описані теоретично. Представлено алгоритмічний опис паралельного методу групового навчання. Ефективність розпаралелення методу досліджена на поступово збільшуваних розмірностях сценаріїв навчання. Результати експериментальних досліджень показали, що (i) обчислювальний кластер з комунікаційним інтерфейсом Infiniband показав кращу ефективність розпаралелення, ніж паралельний комп'ютер загального призначення з ccNuma архітектурою через менші комунікаційні втрати та (ii) ефективність розпаралелення представленого методу є достатньо високою для його успішного застосування на паралельних комп'ютерах та кластерах загального призначення, наявних в сучасних обчислювальних ГРІД-системах.

The development of a parallel method for batch pattern training of a multilayer perceptron with the back propagation training algorithm and the research of its efficiency on general-purpose parallel computer and computational clusters are presented in this paper. The model of a multilayer perceptron and the usual sequential batch pattern training method are theoretically described. An algorithmic description of the parallel version of the batch pattern training method is presented. The efficiency of parallelization of the developed method is investigated on the progressive increasing the dimension of the parallelized problem. The results of the experimental researches show that (i) the computational cluster with Infiniband interconnection shows better values of parallelization efficiency in comparison with general-purpose parallel computer with ccNuma architecture due to lower communication overhead and (ii) the parallelization efficiency of the method is high enough for its appropriate usage on general-purpose parallel computers and clusters available within modern computational grids.

1. Вступ

Штучні нейронні мережі (НМ) є чудовим механізмом для моделювання складних нелінійних систем. Вони є дуже доброю альтернативою традиційним методам рішення складних задач в багатьох сферах включаючи обробку зображень, розпізнавання образів, робототехніку, оптимізацію, моделювання процесів життєдіяльності та інших [1]. В загальному випадку, НМ складається з двох і більше шарів (рівнів) елементарних нейронів з'єднаних синапсами (ваговими коефіцієнтами). Кожен нейрон поточного шару отримує інформацію від нейронів попереднього шару, виконує операцію зваженої суми, використовує її як аргумент для обчислення функції активації та передає результат до нейронів наступного шару. Хоча обчислення в кожному нейроні та НМ загалом є дуже прості, загальне обчислювальне навантаження може бути значним, особливо на етапі навчання НМ (години та дні). Це, без сумніву, є однією з головних перепон до ефективного використання НМ для вирішення прикладних задач. Використання високопродуктивних паралельних комп'ютерів, суперкомп'ютерів, кластерів та обчислювальних ГРІД-систем загального призначення для прискорення етапу навчання НМ є шляхом усунення цієї перепони. Тому розробка паралельних методів навчання НМ та дослідження їх ефективності розпаралелення на таких класах паралельних комп'ютерних систем все ще залишається актуальною науковою проблемою.

Беручи до уваги паралельну природу штучних НМ, багато дослідників досліджували методи їх розпаралелення на спеціалізованому апаратному забезпеченні та трансп'ютерах [2-5], однак ці рішення вимагають наявності згаданих пристроїв для використання широкою науковою спільнотою. В той же час, високопродуктивні паралельні комп'ютери та обчислювальні ГРІД-системи загального призначення широко застосовуються зараз для наукових експериментів та моделювання у віддаленому режимі. Декілька ГРІД-базованих підходів до реалізації НМ представлені в [6-7], однак питання ефективності розпаралелення процесів навчання в них не досліджуються.

Паралельний алгоритм навчання БШП на багатопроцесорному комп'ютері, кластері та обчислювальній ГРІД-системі з використанням бібліотеки MPI (Message Passing Interface) досліджено у [8]. Дослідження проведено на великих моделях НМ, які обробляють велику кількість навчальних векторів (близько 20000), що поступають з Великого Андронного Коллайдера. Однак реалізація розпаралелення відносно «малої» архітектури БШП 16-10-10-1 (16 нейронів у вхідному шарі, два схованих шари по 10 нейронів у кожному та один вихідний нейрон) з 270 внутрішніми налаштовуваними зв'язками (загальна кількість вагових коефіцієнтів та порогів НМ) не забезпечує прискорення розпаралелення через значні втрати часу на передачу повідомлень між паралельними гілками алгоритму, тобто прискорення є меншим, ніж 1 (час виконання паралельної версії програми є більшим від часу виконання послідовної програми, що реалізує той самий алгоритм).

Разом с тим, малі моделі НМ з кількістю зв'язків менше ніж 270, широко використовуються для вирішення практичних задач завдяки кращим узагальнюючим властивостям. В цих моделях кількість нейронів схованих шарів, як правило, є значно меншою від кількості навчальних векторів [1]. Тому розпаралелення малих моделей НМ, що навчаються на не дуже великій кількості вхідних даних, але можуть вимагати великої кількості епох навчання, також є важливою задачею.

В роботах [9-10] автором цієї статті розроблено паралельний метод групового навчання БШП за алгоритмом зворотного поширення помилки та досліджена ефективність його розпаралелення на паралельному комп'ютері загального призначення. Показано, що деталі реалізації паралельного алгоритму насправді відіграють важливу роль. Наприклад, останні версії програмного забезпечення MPI, що мають покращені властивості колективних комунікацій (покращений модуль реалізації колективних функцій пакету Open MPI), показують зменшення комунікаційних втрат, що веде до відповідного підвищення ефективності розпаралелення паралельного алгоритму на обчислювальному кластері [11].

Метою цієї статті є оцінка ефективності розпаралелення розробленого паралельного методу групового навчання за алгоритмом зворотного поширення помилки для БШП на обчислювальних кластерах з комунікаційними інтерфейсами Infiniband та Gigabit Ethernet, та її порівняння з ефективністю розпаралелення на паралельному комп'ютері загального призначення.

2. Метод групового навчання БШП за алгоритмом зворотного поширення помилки

Розпаралелення БШП з стандартним послідовним алгоритмом зворотного поширення помилки не є ефективним на паралельному комп'ютері загального призначення (прискорення менше одиниці) через високі втрати часу на синхронізацію та передачу повідомлень між паралельними процесорами [12]. Тому доцільно використати метод групового навчання, згідно з яким модифікація вагових коефіцієнтів та порогів нейронів здійснюється в кінці кожної епохи навчання, тобто після обробки всіх навчальних векторів з вхідної навчальної вибірки замість модифікації вагових коефіцієнтів та порогів нейронів після обробки кожного навчального вектору в звичайному послідовному режимі навчання.

Вихідне значення тришарового персептрону (рис. 1) визначається згідно з виразом:

$$y = F_3\left(\sum_{j=1}^{N} w_{j3}\left(F_2\left(\sum_{i=1}^{M} w_{ij}x_i - T_j\right)\right) - T\right), \qquad (1)$$

де N – кількість нейронів у схованому шарі, w_{j3} – ваговий коефіцієнт від j-го нейрону схованого шару до вихідного нейрону, w_{ij} – вагові коефіцієнти від i-го нейрону вхідного шару до j-го нейрону схованого шару, x_i – елементи вхідного навчального вектору, T_j – пороги нейронів схованого шару та T – поріг вихідного нейрону [1, 13]. В цій моделі використано стандартну сигмоїдну функцію активації $F(x) = 1/(1+e^{-x})$ для нейронів схованого (F_2) та вихідного (F_3) шарів, однак в загальному випадку ці функції можуть бути різними.

Метод групового навчання за алгоритмом зворотного поширення помилки складається з наступних кроків [13]:

1. Встановити сумарну квадратичну помилку (Sum Squared Error) в необхідне мінімальне значення SSE=$E_{\min}$ та кількість епох навчання t;

2. Проініціалізувати пороги та вагові коефіцієнти нейронів значеннями з відрізку (0…0.5) [13];

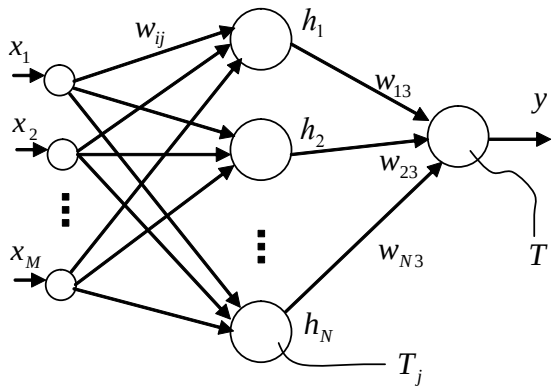


Рис. 1. Структура тришарового персептрону

3. Для вхідного навчального вектору pt виконати наступну навчальну ітерацію:
   3.1. Обчислити вихідне значення БШП $y^{pt}(t)$ згідно з виразом (1);
   3.2. Обчислити помилку вихідного нейрону $\gamma_3^{pt}(t) = y^{pt}(t) - d^{pt}(t)$, де $y^{pt}(t)$ є обчисленим вихідним значенням БШП, а $d^{pt}(t)$ є бажаним вихідним значенням (ціль навчання);
   3.3. Обчислити помилку нейронів схованого шару $\gamma_j^{pt}(t) = \gamma_3^{pt}(t) \cdot w_{j3}(t) \cdot F_3'(S^{pt}(t))$, де $S^{pt}(t)$ - зважена сума вихідного нейрону;
   3.4. Обчислити зміни вагових коефіцієнтів та порогів всіх нейронів та додати їх до відповідних значень змін, отриманих для попередніх навчальних векторів
   $s\Delta w_{j3} = s\Delta w_{j3} + \gamma_3^{pt}(t) \cdot F_3'(S^{pt}(t)) \cdot h_j^{pt}(t)$,
   $s\Delta T = s\Delta T + \gamma_3^{pt}(t) \cdot F_3'(S^{pt}(t))$,
   $s\Delta w_{ij} = s\Delta w_{ij} + \gamma_j^{pt}(t) \cdot F_2'(S_j^{pt}(t)) \cdot x_i^{pt}(t)$,
   $s\Delta T_j = s\Delta T_j + \gamma_j^{pt}(t) \cdot F_2'(S_j^{pt}(t))$, де $S_j^{pt}(t)$ та $h_j^{pt}(t)$ - зважена сума та вихідне значення j-го нейрону схованого шару відповідно;
   3.5. Обчислити SSE використовуючи вираз $E^{pt}(t) = \frac{1}{2}\left(y^{pt}(t) - d^{pt}(t)\right)^2$;
4. Повторити крок 3 вище для кожного навчального вектора pt, де $pt \in \{1,..., PT\}$, PT – розмір усієї вхідної навчальної вибірки;
5. Обчислити вагові коефіцієнти та пороги нейронів використовуючи вирази:
   $w_{j3}(PT) = w_{j3}(0) - \alpha(t) \cdot s\Delta w_{j3}$,
   $T(PT) = T(0) + \alpha(t) \cdot s\Delta T$,
   $w_{ij}(PT) = w_{ij}(0) - \alpha(t) \cdot s\Delta w_{ij}$,
   $T_j(PT) = T_j(0) + \alpha(t) \cdot s\Delta T_j$ де $\alpha(t)$ – крок навчання;
6. Обчислити загальну SSE $E(t)$ для навчальної епохи t використовуючи вираз $E(t) = \sum_{pt=1}^{PT} E^{pt}(t)$;

Якщо значення $E(t)$ більше, ніж бажана мінімальна помилка $E_{\min}$, то збільшити номер навчальної епохи до $t+1$ та знову перейти до кроку 3, в іншому випадку – завершити процес навчання.

3. Паралельний метод групового навчання бшп за алгоритмом зворотного поширення помилки

Аналіз методу групового навчання в розділі 2 вище показує, що послідовне виконання кроків 3.1-3.5 для всіх навчальних векторів у вибірці навчання може бути розпаралелено, тому що операції сумування значень $s\Delta w_{j3}$, $s\Delta T$, $s\Delta w_{ij}$ та $s\Delta T_j$ є незалежними одна від одної. Для розробки паралельного методу необхідно розділити весь об'єм обчислень між головним процесором *Master* (буде виконувати функції призначення частин програмного коду робочим паралельним процесорам та власні обчислення) та робочими процесорами *Workers* (будуть виконувати частини програмного коду паралельно).

Алгоритми процесорів *Master* та *Worker* наведено на рис. 2. *Master* починає функціонування з визначення (i) кількості векторів *PT* у вхідній навчальній вибірці та (ii) кількості процесорів *p*, на яких завантажено паралельну програму на виконання. *Master* розділяє всі навчальні вектори на рівні частини відповідно до кількості робочих процесорів *Workers,* а також призначає одну частину обчислювальної роботи для себе. Після цього *Master* посилає робочим процесорам *Workers* номери визначених навчальних векторів для здійснення навчання.

Кожен робочий процесор *Worker* виконує наступні операції для кожного патерну *pt* зі всіх *PT/p* патернів, що були йому призначені:

- Виконати кроки 3.1-3.5 та 4. Значення часткових сум вагових коефіцієнтів $s\Delta w_{j3}$, $s\Delta w_{ij}$

та порогів $s\Delta T$, $s\Delta T_j$ обчислюються на цьому кроці;

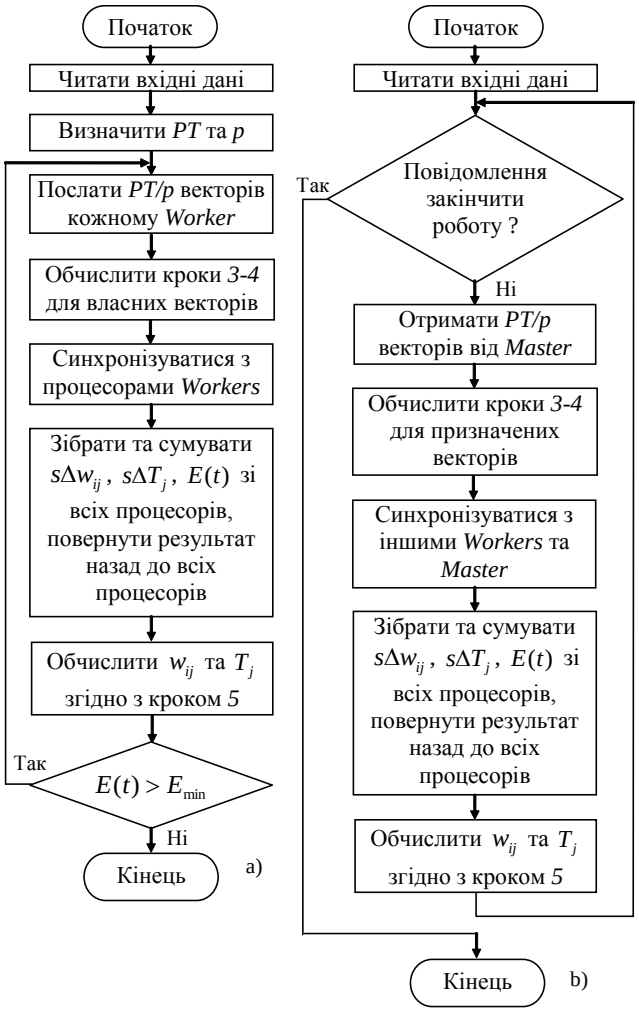


Рис. 2. Алгоритми головного та робочих процесорів Master (a) та Worker (b)

- Обчислити часткові значення SSE для призначених навчальних векторів.

Після обробки всіх призначених векторів кожним процесором виконується тільки одна функція колективної комунікації (вона також здійснює і сумування). Синхронізація з іншими процесорами автоматично забезпечується внутрішньою реалізацією цієї функції колективної комунікації [14]. Однак з алгоритмічної точки зору операція синхронізації показана окремо на рис. 2 перед операцією збору даних. Після збору даних сумарні значення $s\Delta w_{j3}$, $s\Delta T$, $s\Delta w_{ij}$ та $s\Delta T_j$ посилаються до всіх процесорів, що працюють паралельно. Використання тільки однієї функції колективної комунікації, що повертає всі зібрані значення з паралельно працюючих процесорів *Workers* назад до всіх процесорів у групі, дозволяє зменшити комунікаційні втрати паралельного алгоритму. Після отримання всі сумарні значення $s\Delta w_{j3}$, $s\Delta T$, $s\Delta w_{ij}$ та $s\Delta T_j$ поміщаються в локальну пам'ять кожного процесора. Кожен процесор використовує ці сумарні значення для обчислення нових значень вагових коефіцієнтів та порогів згідно з кроком 5 методу. Потім нові значення вагових коефіцієнтів та порогів використовуються на наступній епосі методу навчання. Так як сумарне значення $E(t)$ також отримується в результаті операції збору даних, то головний процесор *Master* вирішує – продовжувати навчання чи ні.

Програмний код розроблено на мові програмування C з використанням стандартних MPI-функцій. Паралельна частина алгоритму починається викликом функції *MPI_Init()*. Функція *MPI_Allreduce()* здійснює (i) збір сумарних по кожному процесору часткових вагових коефіцієнтів $s\Delta w_{j3}$, $s\Delta w_{ij}$ та порогів $s\Delta T$, $s\Delta T_j$, (ii) їх подальше сумування та (iii) відсилку кінцевих (просумованих по всіх векторах та процесорах) значень вагових коефіцієнтів та порогів до всіх інших процесорів, що працюють паралельно в групі. Так як вагові коефіцієнти та пороги фізично розміщуються в сегменті коду програми в різних змінних/матрицях, то з метою використання тільки одного виклику функції *MPI_Allreduce()* доцільно «запакувати» ці різні змінні/матриці в один вектор даних перед відсилкою даних, а на приймачі здійснити обернене «розпакування» отриманих даних у відповідні змінні/матриці. Використання тільки одного фізичного виклику функції *MPI_Allreduce()* дозволяє додатково зменшити комунікаційні втрати паралельного алгоритму. Функція *MPI_Finalize()* завершує паралельний алгоритм.

4. Результати експериментальних досліджень

Експериментальні дослідження розробленого паралельного методу здійснені на трьох паралельних обчислювальних системах загального призначення:

- обчислювальний кластер *Reef*, розміщений у Суперкомп'ютерному та мережевому центрі м. Познань, Польща. Кластер складається з 285-ти обчислювальних вузлів. Для проведення експериментальних досліджень використано групу з 32-х обчислювальних ядер, що розмішуються в чотирьох обчислювальних вузлах. Кожен вузол вміщує два чотири-

ядерних процесора Intel Xeon E5345 з тактовою частотою 2.33 ГГц. Вузли з'єднані між собою інтерфейсами Gigabit Ethernet, Fast Ethernet та Infiniband. Експерименти проведено в режимі, коли для з'єднання між вузлами використано тільки інтерфейс Infiniband. Бібліотека Open MPI 1.2.8 використана для проведення експерименту. В подальшому викладенні цей кластер позначено ідентифікатором "Cluster Infiniband";

- обчислювальний кластер *Battlecat*, розміщений в Лабораторії інноваційних обчислень (ICL) університету шт. Теннессі, США. Кластер складається з одного головного вузла та 7-ми робочих вузлів. Головний вузол складається з двох процесорів Dual Xeon 1.6 ГГц з 4 Мб кеш-пам'яті та 2 Гб локальної пам'яті. Кожен робочий вузол має один двох-ядерний процесор Intel Core 2 Duo 2.13 ГГц з 2 Мб кеш-пам'яті та 2 Гб локальної пам'яті. Вузли з'єднані між собою інтерфейсом Gigabit Ethernet. Бібліотека Open MPI 1.4.1 та 16 обчислювальних ядер використано для проведення експерименту. Цей кластер є важливим для проведення експериментальних досліджень, так як в режимі використання 8-ми та 16-ти ядер кластер має гетерогенну архітектуру як з точки зору обчислювальної потужності ядер, так і з точки зору швидкості передачі даних між ядрами. В подальшому викладенні цей кластер позначено ідентифікаторами "Cluster Gigabit Ethernet" або "CGE";
- паралельний комп'ютер NEC TX7 *Crati*, розташований в Суперкомп'ютерному центрі обчислювальної інженерії, Калабрійський університет, Італія. Комп'ютер складається з 28 одноядерних 64-х розрядних процесорів Intel Itanium 2 з тактовою частотою 1 ГГц та загальною пам'яттю 64 Гб. Кожен процесор має кеш другого рівня розміром 3 Мб. Комп'ютер має архітектуру ccNuma (неоднаковий доступ процесорів до загальної пам'яті з когерентною кеш-пам'яттю) та міжпроцесорний інтерфейс ccNuma link. Бібліотека NEC MPI/EX 1.5.2 та 28 процесорів використано для проведення експерименту. В подальшому викладенні цей комп'ютер позначено ідентифікатором "Computer ccNuma" або "CccNUMA".

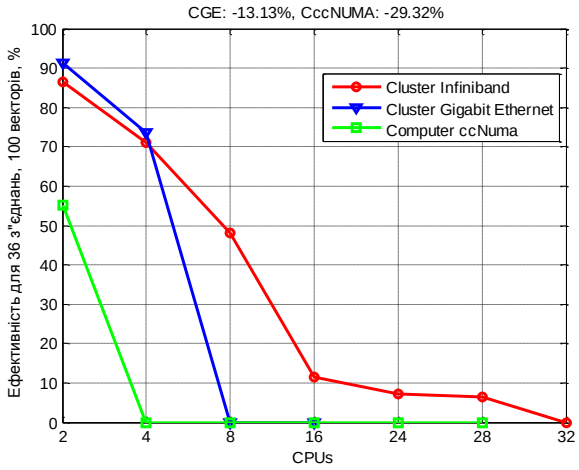
Для проведення експериментальних досліджень використано 8 сценаріїв збільшення розміру БШП та кількості навчальних векторів: БШП 5-5-1 (5 нейронів вхідного шару * 5 нейронів схованого шару = *25 зв'язків* – вагових коефіцієнтів, 5 нейронів схованого шару * 1 вихідний нейрон = ще *5 зв'язків* – вагових коефіцієнтів, всього є 6 (5 схованих та 1 вихідний) нейронів, що здійснюють обробку інформації, – кожен з має поріг = *6 зв'язків* – порогів. Всього в БШП 5-5-1 є *36 налаштовуваних зв'язків*, що змінюють свої значення в процесі його навчання) та 100 навчальних векторів, БШП 10-10-1 (*121 налаштовуваний зв'язок*) та 200 навчальних векторів, БШП 15-15-1 (*256 налаштовуваних зв'язків*) та 400 навчальних векторів, БШП 20-20-1 (*441 налаштовуваний зв'язок*) та 600 навчальних векторів, БШП 30-30-1 (*961 налаштовуваний зв'язок*) та 800 навчальних векторів, БШП 40-40-1 (*1681 налаштовуваний зв'язок*) та 1000 навчальних векторів, БШП 50-50-1 (*2601 налаштовуваний зв'язок*) та 5000 навчальних векторів та БШП 60-60-1 (*3721 налаштовуваний зв'язок*) та 10000 навчальних векторів. Ці сценарії збільшення розміру БШП та кількості навчальних векторів вибрані з метою дослідження ефективності розпаралелення методу як на малих, так і на великих розмірностях обчислювальної задачі. Нейрони схованого та вихідного шарів мали сигмоїдну функцію активації. Кількість епох навчання складала 10^4. Крок навчання був сталий і дорівнював $\alpha(t) = 0.01$. Вирази $S=Ts/Tp$ та $E=S/p\times 100\%$ використано для знаходження прискорення S та ефективності розпаралелення E, де Ts – час виконання послідовної програми, Tp – час виконання паралельної версії тієї ж самої програми на p процесорах паралельної обчислювальної системи.

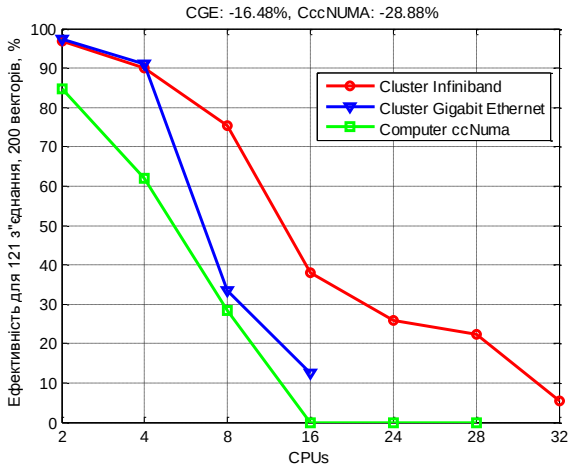
Значення ефективності розпаралелення методу групового навчання БШП за алгоритмом зворотного поширення помилки наведено на рис. 3 для всіх сценаріїв розпаралелення на всіх паралельних обчислювальних системах. Як видно, кращі результати розпаралелення отримані на системі Cluster Infiniband у порівнянні з системами CGE та CccNUMA для всіх сценаріїв, при цьому значення ефективності розпаралелення є приблизно однаковими на двох останніх системах. Тому обчислено усереднені спади ефективності розпаралелення для цих двох систем на тій самій кількості процесорів для кожного сценарію у порівнянні з Cluster Infiniband. Ці спади показані вгорі відповідного результату на рис. 3. Наприклад, для сценарію 36 зв'язків, 100 векторів система Cluster Infiniband переважає Cluster Gigabit Ethernet на 13.13%, а

Computer ccNuma на 29.32% ефективності розпаралелення в середньому.

Аналіз рис. 3 показує, що ефективність розпаралелення підвищується при збільшенні кількості зв'язків в БШП та збільшенні кількості вхідних навчальних векторів через збільшення складності обчислювальної частини алгоритму. Однак ефективність розпаралелення зменшується для того самого сценарію при збільшенні кількості паралельних процесорів через збільшення комунікаційних втрат. Це підтверджується шляхом оцінки часу обчислення та комунікації для 4 найменших сценаріїв на рис. 4. В кожній з 5-ти груп стовпчиків, три стовпчики для кожного сценарію описують обчислювальну (зелений колір) та комунікаційну (жовтий колір) частини алгоритму на відповідних паралельних системах – зліва-направо – Cluster Infiniband, CGE та CccNUMA на 1, 2, 4, 8 та 16 процесорах відповідно. Як видно, система Cluster Infiniband (ліва колонка в кожній групі) показує найменші комунікаційні втрати для всіх сценаріїв, тому вона забезпечує найкращу ефективність розпаралелення.

Отримані значення ефективності розпаралелення є достатньо високі, зокрема навіть для «малих» сценаріїв з 36, 121 та 256 зв'язками система Cluster Infiniband показує прискорення розпаралелення на 16-ти та більше процесорах. Однак з метою підвищення ефективності доцільно малі сценарії задачі розпаралелювати на малій кількості процесорів, тому що час навчання таких сценаріїв не є значним. Аналіз абсолютної тривалості часу виконання на рис. 4 показує, що паралельний комп'ютер CccNuma є швидшим, ніж кластери незважаючи на (i) нижчу тактову частоту процесорів та (ii) нижчі значення ефективності розпаралелення. Цей факт пояснюється внутрішньою оптимізацією програмного коду, реалізованою розробниками комп'ютера NEC TX-7. Результати експериментальних досліджень показали, що у випадку використання обчислювальної ГРІД-системи для ефективного мапування (призначення) процесу паралельного навчання НМ доцільно враховувати не тільки ефективність розпаралелення, але й час виконання задачі також. Це може бути забезпечено шляхом використання брокера ресурсів [15-16], що зараз розробляється в рамках бібліотеки для паралельного навчання НМ PaGaLiNNeT. Цей брокер дозволяє вибрати конкретну паралельну систему з конкретною кількістю паралельних процесорів (серед наявних в ГРІД), що можуть забезпечити певний консенсус між мінімізацією часу навчання НМ та максимізацією ефективності розпаралелення для конкретного сценарію внутрішніх налаштовуваних зв'язків НМ та кількості вхідних навчальних векторів. Наприклад, якщо сценарій БШП 20-20-1 (441 зв'язок) та 600 навчальних векторів поступає на розпаралелення, брокер може вибрати 8 процесорів кластеру Cluster Infiniband (ефективність розпаралелення 92.7%, час обчислення – 6.48 сек.) або 4 процесори паралельного комп'ютера CccNuma (ефективність розпаралелення 92.4%, час обчислення – 5.94 сек.). Такий механізм забезпечує високу ефективність розпаралелення любого сценарію вхідної задачі навчання НМ, так само як і економне використання ГРІД-інфраструктури.





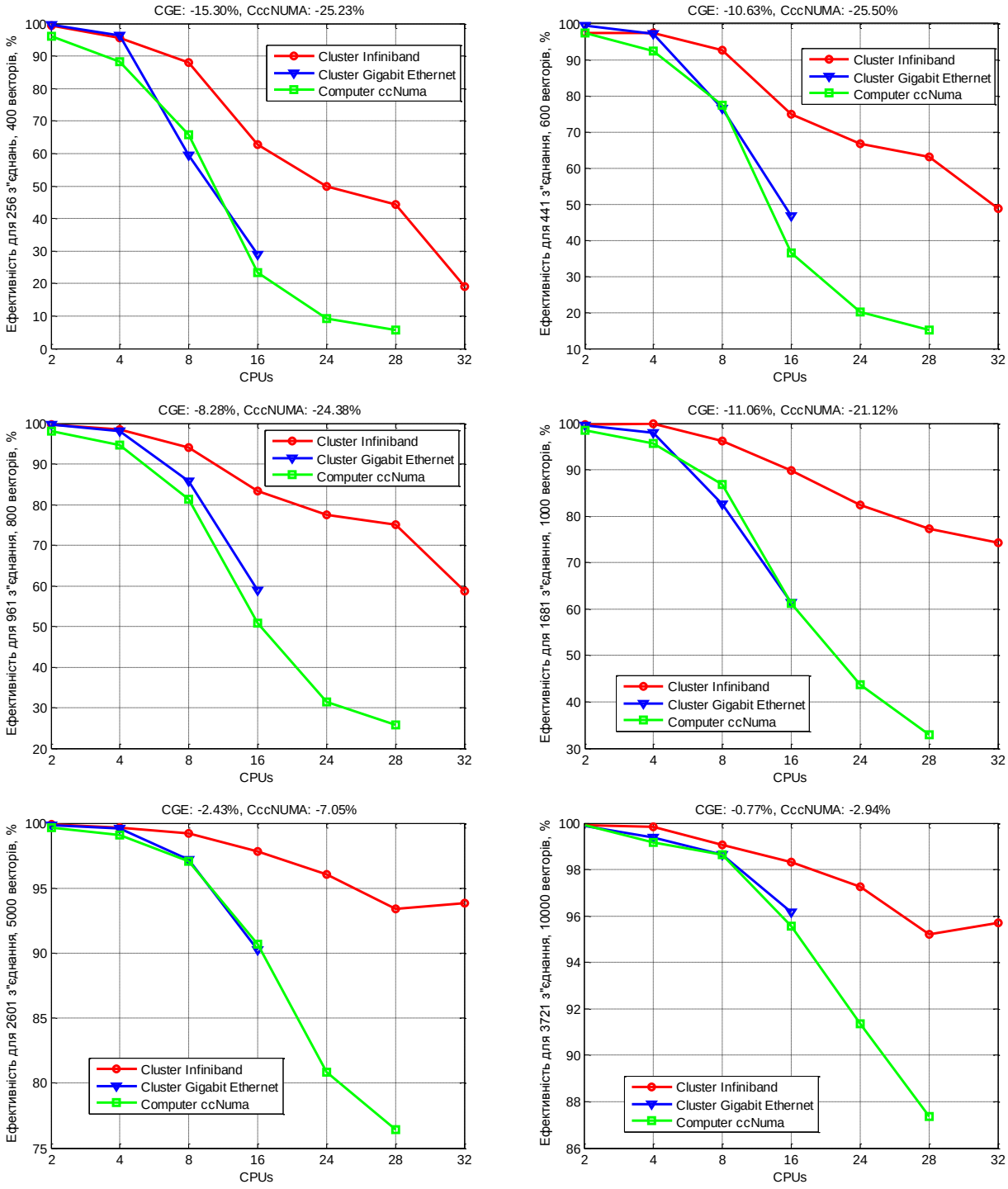
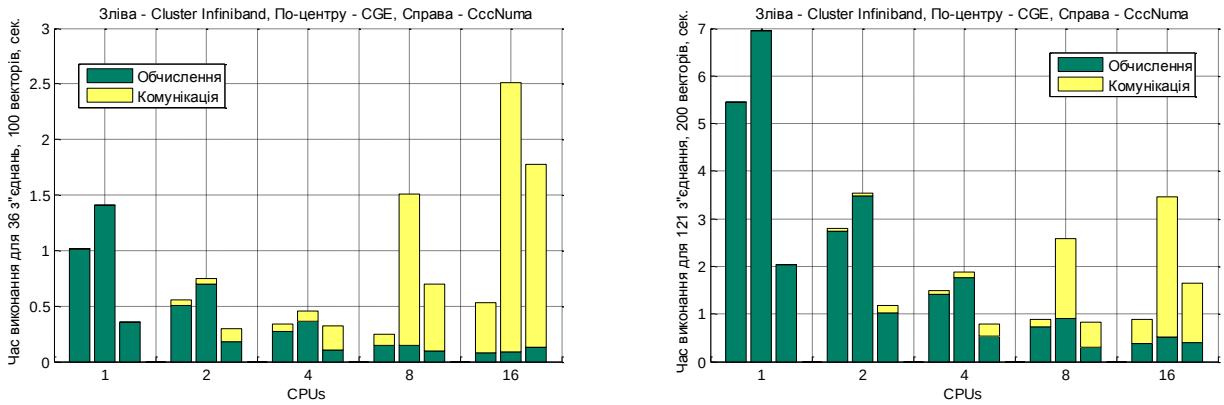


Рис. 3. Ефективність розпаралелення на різних обчислювальних системах



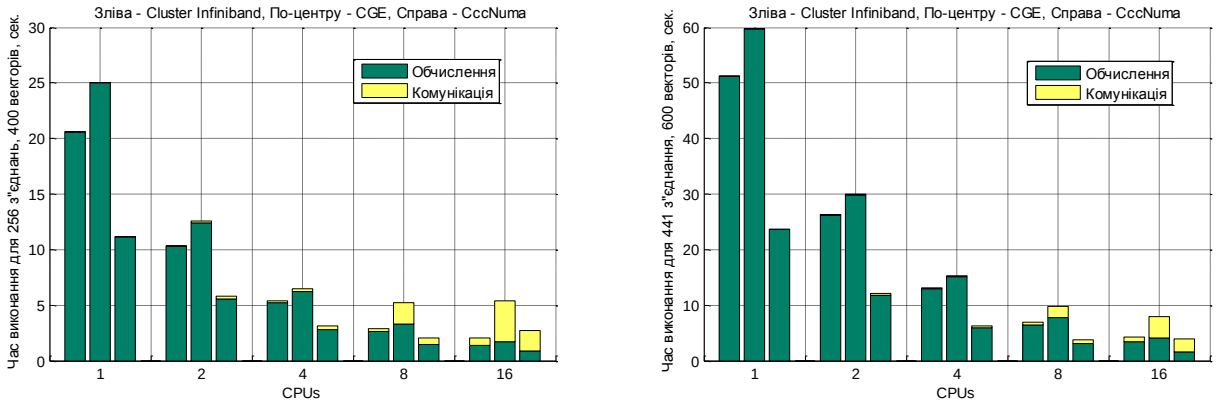


Рис. 4. Аналіз часу виконання обчислювальної та комунікаційної частин алгоритму на різних паралельних системах

Доцільно зазначити, що отримані малі значення часу навчання НМ, особливо для малих сценаріїв, показані в цій статті тільки з метою проведення експериментальних досліджень ефективності розпаралелення розробленого методу. Під час досліджень використано відносну малу кількість епох навчання – 10^4 з урахуванням того, що ефективність розпаралелення методу групового навчання не залежить від кількості епох навчання [17]. Для реальних задач модель БШП вимагає $10^6 \ldots 10^9$ епох для забезпечення достатньо хорошого навчання моделі, в цьому випадку реальний час обчислень буде набагато більшим і застосування паралельних методів навчання стає все більш необхідним.

6. Висновки

В цій статті представлено паралельний метод групового навчання багатошарового персептрону на основі алгоритму зворотного поширення помилки та дослідження його ефективності на обчислювальних кластерах з комунікаційними інтерфейсами Infiniband та Gigabit Ethernet у порівнянні з паралельним комп'ютером ccNuma архітектури. Отримані експериментальні результати показали, що (i) завдяки малим комунікаційним втратам кластер з комунікаційним інтерфейсом Infiniband забезпечує кращу ефективність розпаралелення, ніж суперкомп'ютер з ccNuma архітектурою та (ii) ефективність розпаралелення розробленого паралельного методу є достатньо високою для його ефективного використання на паралельних обчислювальних системах загального приз-начення, наявних у сучасних ГРІД-системах.

Майбутні дослідження доцільно провести в напрямку оцінки ефективності розпаралелення розробленого методу в середовищі серединного програмного забезпечення ГРІД-систем. Експерименти показали, що паралельний комп'ютер з меншою продуктивністю процесорів, але з оптимізованою архітектурою, забезпечує швидке виконання розробленого паралельного методу. Тому для правильного вибору паралельної обчислювальної системи, що має виконувати цей метод в середовищі ГРІД, доцільно ввести додатковий критерій вибору – «вартість» паралельної системи, що може відображати справжню вартість обчислювальної системи, її архітектурні особливості, завантаження задачами та інші необхідні характеристики.

Подяки

Це дослідження фінансовано міжнародною «в'їздною» стипендією ім. Марії К'юрі №221524-908524 "PaGaLiNNeT – Parallel Grid-aware Library for Neural Networks Training" 7-ї Рамкової програми Європейського союзу. Автор також дякує адміністрації та технічним працівникам Суперкомп'ютерного та мережевого центру м. Познань (Польща), Лабораторії інноваційних обчислень (ICL) університету шт. Теннессі (США) та Суперкомп'ютерного центру обчислювальної інженерії Калабрійського університету (Італія) за використання їх обчислювальної ГРІД-інфраструктури.

Список використаних джерел

1. Haykin S. Neural networks and learning machines / S. Haykin. – Prentice Hall, 2008. – 936 p.
2. Mahapatra S., Mahapatra R., Chatterji B. A Parallel Formulation of Back-propagation Learning on Distributed Memory Multiprocessors // Parallel Computing, 1997. – Vol. 22, No. 12. – P. 1661-1675.
3. Hanzálek Z. A Parallel Algorithm for Gradient Training of Feed-forward Neural Networks // Parallel Computing, 1998. – Vol. 24, No. 5 -6. – P. 823-839.
4. Murre J.M.J. Transputers and Neural Networks: An Analysis of Implementation Constraints and Performance // IEEE Transactions on Neural Networks, 1993. – Vol. 4, No. 2. – P. 284-292.
5. Topping B.H.V., Khan A.I., Bahreininejad A. Parallel Training of Neural Networks for Finite Element Mesh Decomposition // Computers and Structures, 1997. – Vol. 63, No. 4. – P. 693-707.
6. Vin T.K., Seng P.Z., Kuan M.N.P., Haron F. A Framework for Grid-based Neural Networks // Proceedings of First International Conference on Distributed Frameworks for Multimedia Applications. – 2005. – P. 246-250.
7. Krammer L., Schikuta E., Wanek H. A Grid-based Neural Network Execution Service Source // Proceedings of 24th IASTED International Conference on Parallel and Distributed Computations and Networking. – 2006. – P. 35-40.
8. De Llano R.M., Bosque J.L. Study of Neural Net Training Methods in Parallel and Distributed Architectures // Future Generation Computer Systems, 2010. – Vol. 26, Issue 2. – P. 183-190.
9. Turchenko V., Grandinetti L. Efficiency Analysis of Parallel Batch Pattern NN Training Algorithm on General-Purpose Supercomputer // Lecture Notes in Computer Science. – Berlin, Heidelberg: Springer-Verlag, 2009. – No. 5518. – P. 223 - 226.
10. Turchenko V., Grandinetti L. Minimal Architecture and Training Parameters of Multilayer Perceptron for its Efficient Parallelization // Proceedings of 5th International Workshop on Artificial Neural Networks and Intelligent Information Processing. – Milan (Italy). – 2009. – P. 79-87.
11. Turchenko V., Grandinetti L., Bosilca G., Dongarra J. Improvement of parallelization efficiency of batch pattern BP training algorithm using Open MPI // Elsevier Procedia Computer Science, 2010. – Vol. 1, Issue 1. – P. 525-533.
12. Turchenko V., Grandinetti L. Efficiency Research of Batch and Single Pattern MLP Parallel Training Algorithms // Proceedings of 5th IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS2009). – Rende (Italy). – 2009. – P. 218-224.
13. Головко В.А. Нейронные сети: обучение, модели и применение. – М.: Радиотехника, 2001. – 256 с.
14. http://www.open-mpi.org/
15. Turchenko V., Grandinetti L. Strategy of Resource Brokering for Efficient Parallelization of MLP Training // Proceedings of the 2010 International Conference on High Performance Computing & Simulation (HPCS 2010). – Caen (France). – 2010. – P. 140-149.
16. Турченко В.О. Методологія брокерування Грід-ресурсів на основі Парето-оптимізації // Вимірювальна та обчислювальна техніка в технологічних процесах. – 2011. – № 1. – С. 312-318.
17. Turchenko V. Scalability of Parallel Batch Pattern Neural Network Training Algorithm // Artificial Intelligence, Journal of National Academy of Sciences of Ukraine, 2009. – Vol. 2. – P. 144-150.

УДК 681.03

ХИМИЧ А. Н.,
ЧИСТЯКОВА Т. В.,
БАРАНОВ А. Ю.

АВТОМАТИЧЕСКИЙ АДАПТИВНЫЙ РЕШАТЕЛЬ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ ДЛЯ ГИБРИДНЫХ СИСТЕМ

Предлагается автоматический решатель для исследования и решения систем линейных алгебраических уравнений с приближенно заданными исходными данными на гибридных системах с функцией адаптивной настройки алгоритма, программы и архитектуры компьютера на свойства решаемой задачи. При этом пользователи освобождаются от самостоятельного исследования свойств задачи с целью выбора необходимого алгоритма решения, анализа достоверности получаемых результатов, а также проблем, связанных с программированием на сложных гибридных системах.

An automatic solver is proposed for the investigating and solving of linear algebraic systems with approximately given input data on hybrid computer systems able to adaptively adjust algorithm, program and computer's architecture according to characteristics of the problem being solved. In so doing the users are exempted from investigation of characteristics of the problem being solved in order to select and appropriate solution algorithm, analyze the reliability of the obtained results as well as to solve problems related to the programming on complex hybrid systems.

1. Введение

В настоящее время наблюдается определенная тенденция в развитии вычислительных систем. С одной стороны, как и прежде, продолжается рост производительности компьютеров за счет увеличения числа процессорных ядер, а с другой стороны, становятся все болем популярными гибридные системы, гетерогенные системы, высокая производительность которых обусловлена применением вычислительных ресурсов принципиально новой архитектуры.

Первое направление развития, требует от алгоритмов большой степени параллелизма на однотипных процессорных ядрах, который на программном уровне реализуется с помощью специальных систем параллельного программирования таких, например, как MPI.

Второе направление требует от алгоритма более сложной многоуровневой параллельной модели, которая учитывает различные архитектуры используемых вычислительных ресурсов.

Можно выделить следующие этапы работ, которые необходимо предусмотреть при разработке параллельных алгоритмов для гибридных систем:

- декомпозиция задачи, т.е. разделение задачи на подзадачи, которые могут бать реализованы в значительной степени не зависимо друг от друга;
- выделение для сформированного набора подзадач информационные взаимосвязи, которые должны осуществляться в ходе решения задачи;
- определение необходимых вычислительных устройств центрального процесора (CPU) и графического ускорителя (GPU) для эффективной реализации подзадач и распределение подзадач и данных между ними.

Программа, реализующая алгоритм решения задачи на гибридной системе с одним CPU и одним GPU, будет состоять из программных кодов для CPU, написанных, например, на Си или C++, и кодов для GPU, написанных, например, с использованием технологии CUDA.

Известно, что многие программные приложения из различных предметных областей науки и производства зависят от успешного решения систем линейных алгебраических уравнений (СЛАУ), к которым сводятся решаемые задачи. Несмотря на большое внимание к созданию программного обеспечению по этой проблематике на различные типы компьютерных архитектур [1, 2], многие проблемы эффективного его использования остаются.

Прежде всего, при использовании современного программного обеспечения для решения СЛАУ возникают проблемы решения задач с приближенно заданными исходными данными:

- исследование корректности постановки математической задачи в компьютере;
- определение обусловленности компьютерной модели задачи;

- оценка погрешности получаемого компьютерного решения.

Другая проблема связана с использованием вычислительных ресурсов компьютеров параллельной архитектуры, с согласованием распределения ядер CPU и процессоров GPU, а также оптимизацией коммуникационных расходов между ядрами CPU и процессорами GPU. Кроме того, пользователю необходимо осваивать новые языки программирования для написания программ на различные вычислительные ресурсы.

Предлагается адаптивный решатель для исследования и решения СЛАУ на гибридных системах, реализующий автоматическое исследование и решение задачи с приближенно заданными исходными данными, анализ получаемых решений, осуществляя при этом динамическое планирование вычислений на каждом шаге алгоритма на необходимых вычислительных ресурсах.

2. Реализация автоматического исследования и решения задачи с автоматическим планированием вичислений

Основные концептуальные принципы автоматического адаптивного решателя для СЛАУ на гибридных системах:

- возможность решения задач с приближенно заданными исходными данными;
- естественные для пользователя формы ввода исходных данных;
- автоматизация процессов исследования математических свойств компьютерной модели задачи, выбора необходимого алгоритма и синтеза программы решения;
- решение задачи с оценкой достоверности получаемых компьютерных результатов;
- получение не только решения задачи, но и протокола процесса решения задачи с анализом выявленных ее свойств и достоверности полученных результатов;
- реализация автоматического планирования вычислений на требуемых вычислительных устройствах CPU и GPU при выполнении конкретного алгоритма для эффективного решения задачи.

Автоматическое планирование вычислений предполагает двухуровневое планирование:

1-й уровень − автоматическое исследование математических свойств компьютерной модели задачи и выбор соответствующего алгоритма решения с оценками достоверности (оценка наследственной погрешности в математическом решении и оценка вычислительной погрешности) [3];

2-й уровень − автоматическое планирование реализации конкретного алгоритма на предоставленные вычислительные ресурсы гибридной системы с целью наименьших затрат компьютерного времени и мощностей составляющих компонентов гибрида.

В данном программном обеспечении реализованы известные алгоритмы прямых методов Холецкого и Гаусса – для решения СЛАУ с плотными невырожденными матрицами, а также метод сингулярного разложения – для нахождения нормального обобщенного решения систем с матрицами неполного ранга. Выбор необходимого алгоритма решения задачи и синтез необходимой программы происходит в процессе исследования ее характерных свойств в компьютере (вид матрицы системы, положительная определенность матрицы, вырожденность матрицы, плохая обусловленность и т. д.) с учетом приближенного характера исходных данных.

Для реализации планирования вычислений второго уровня используется система StarPU [4], которая позволяет планировать выполнение задачи с использованием многоядерных CPU и процессоров GPU в различном сочетании, наиболее эффективном для задачи.

Параллелизм задачи осуществляется, используя последовательные коды программ, с указанием частей алгоритмов (подзадач), которые реализуются на различных вычислительных устройствах гибрида: только на ядре CPU, только на GPU, на обоих устройствах одновременно (гибридное задание). StarPU динамически планирует вычисления с учетом затрат времени выполнения каждой подзадачи на каждом доступном вычислительном устройстве. При этом автоматически выполняется асинхронное копирование необходимых данных как между CPU и GPU, так и между GPU. Исходные данные задачи один раз регистрируются в среде StarPU, и в дальнейшем система сама копирует необходимые данные между CPU и GPU, максимально перекрывая время выполнения коммутационных связей временем выполнения математических операций.

На рис. 1 представлен фрагмент схемы исследования и решения СЛАУ.

Здесь можно выделить следующие этапы работ:

1. Пользователь задает (программно, из файла или из клавиатуры, заполняя оконные формы) исходные данные СЛАУ: количество строк (*N*) и столбцов (*M*) матрицы, элементы матрицы (*A*) и их максимальная относительная погрешность (*EA*), элементы правых частей (*B*) и их максимальная относительная погрешность (*EB*), а также указывается вид матрицы – симметричная или нет.

2. Если матрица симметричная, то для исследования свойств матрицы и решения системы выбирается метод Холецкого (программа PPFA). При этом с помощью StarPU определяются необходимые вычислительные устройства: ядра CPU и процессоры GPU, а также соответствующая программа для реализации алгоритма решения: только на ядрах CPU, на одном ядре CPU с использованием процессора GPU, на нескольких ядрах CPU с использованием нескольких процессоров GPU.

3. Если матрица в компьютере оказалась положительно определенной, то решение СЛАУ продолжается (программа PPSL с подключением StarPU для автоматизации планирования вычислений 2-го уровня), иначе матрица исследуется на вырожденность методом Гаусса (программа GEFA с подключением StarPU).

4. Если матрица системы в компьютере оказалась невырожденной, то решение СЛАУ продолжается (программа GESL с подключением StarPU), иначе вычисляется нормальное обобщенное решение методом наименьших квадратов (программа SVD с подключением StarPU).

5. По окончанию вычислительного процесса выдается протокол решения задачи с указанием методов, которыми исследовалась задача, задействованных вычислительных ресурсов, выявленных свойств, оценок достоверности. Решение СЛАУ пользователь может сохранить по желанию на различных носителях информации.

Рассмотрим, как реализуется планирование вычислений второго уровня с помощью StarPU на примере алгоритма факторизации $n \times n$ симметричной положительно определенной матрицы системы методом Холецкого: $A=LL^T$.

За основу вычислительной схемы берем схему реализации блочного алгоритма Холецкого на основе последовательных кодов четырех операций из пакетов программ LAPACK [5] и BLAS [6]:

xPOTRF – факторизация ведущего диагонального блока матрицы;

xSYRK – модификация диагональных (нижних треугольных) блоков матрицы;

xGEMM – преобразование блоков матрицы, которые находятся под диагональю справа от ведущего блока;

xTRSM – преобразование блоков матрицы, которые находятся в одном столбике с ведущим блоком (решение СЛАУ с треугольной матрицей).

Здесь вместо x указывается разрядность вычислений: S − одинарная разрядность, D − удвоенная разрядность.

Псевдокод реализации $A=LL^T$-факторизации матрицы методом Холецкого имеет вид:

```
for (k = 0; k  < Nt; k++)
A[k][k] <– DPOTRF(A[k][k])
for (m = k+1; m  < Nt; m++)
A[m][k] <– DTRSM(A[k][k],A[m][k])
for (n = k+1; n  < Nt; n++)
A[n][n] <– DSYRK(A[n][k],A[n][n])
for (m = n+1; m  < Nt; m++)
A[m][n]     <–     DGEMM(A[m][k],A[n][k],
A[m][n])
```

Как мы видим из псевдокода, операция во второй строке может быть выполнена раньше, чем был полностью выполнен предыдущий шаг в цикле. Для начала ее выполнения достаточно, чтобы на предыдущем шаге было определено A[k][k]. Аналогичные рассуждения можно привести и для остальных операций. Таким образом, чтобы приступить к выполнению операций на шаге k-цикла, не обязательно, чтобы шаг k-1 был завершен. Достаточно, чтобы были получены необходимые для каждой конкретной операции блоки матрицы. StarPU дает возможность заранее объявить все операции, которые будут нужны для выполнения алгоритма, а также определить между ними зависимости, которые дают возможность контролировать порядок выполнения операций. И затем все операции будут выполняться в том порядке и на том количестве процессоров, которые обеспечат самую высокую эффективность реализации алгоритма.

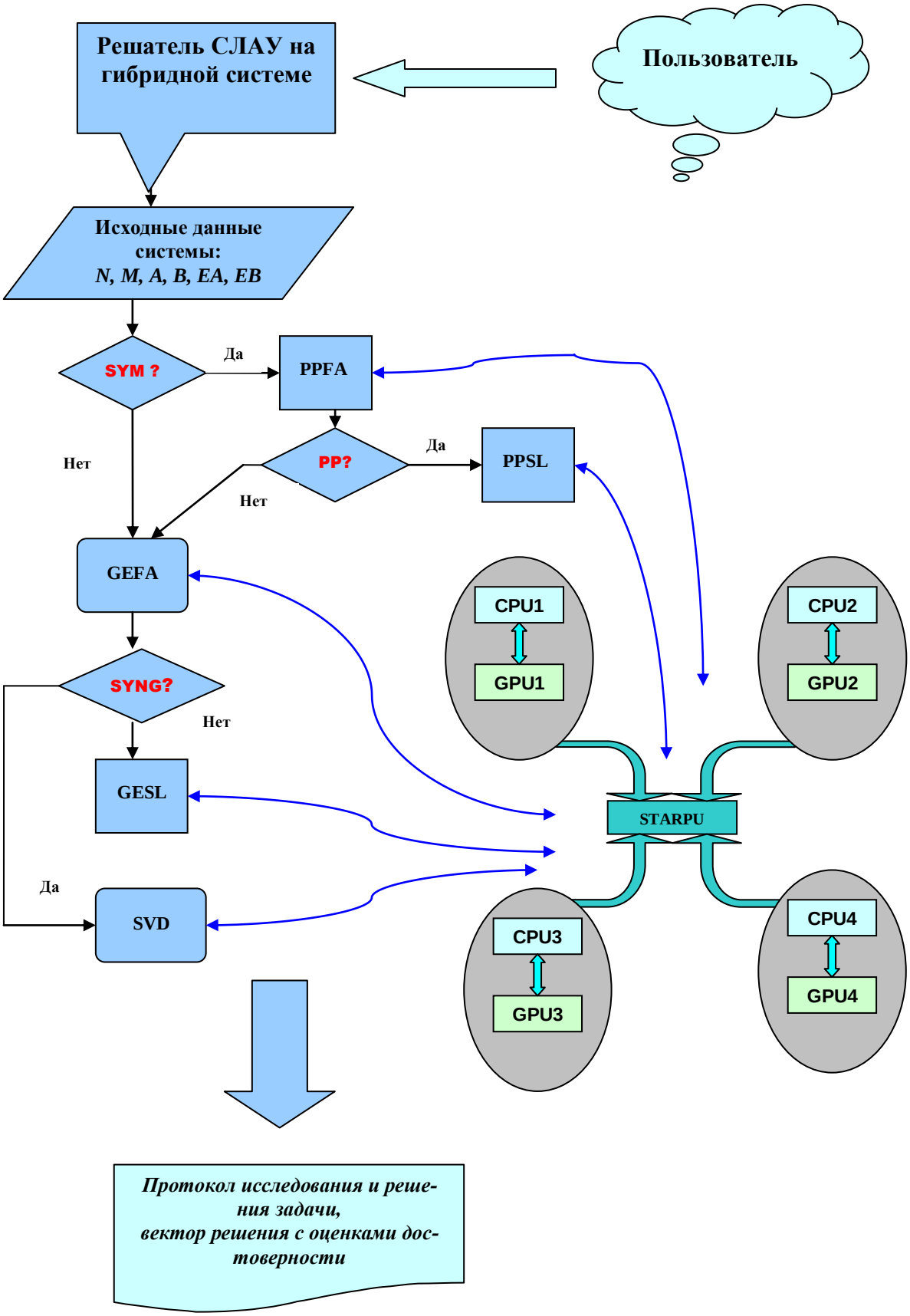


Рис. 1. Схема автоматического исследования и решения СЛАУ

В итоге, алгоритм состоит из выполнения следующих процедур:

- разделение задачи на подзадачи;
- параллельные подзадачи (xTRSM, xDSYRK, xDGEMM) “планируются“ для эффективного выполнения на GPU с учетом асинхронности, величины блока;
- последовательная подзадача xPOTRF – разложение диагонального блока выполняется на CPU;
- малые задания на CPU частично покрываются большими заданиями GPU (умножение матриц).

Таким образом, алгоритм Холецкого представляется в виде направленного ациклического графа (рис. 2), в котором вершины представляют собой операции, которые нужно выполнить, а направленные ребра – зависимости между ними. Для каждой последующей подзадачи необходимые данные будут готовы тогда и только тогда, когда будут выполнены все предыдущие подзадачи, от которых она зависит. Таким образом, для решения СЛАУ методом Холецкого нужно выполнить эти операции в определенной последовательности.

StarPU предоставляет набор планировщиков, с помощью которых можно добиться максимально быстрой реализации алгоритма в зависимости от архитектуры гибридной системы. Для алгоритма Холецкого самым эффективным является Heft-планировщик, основанный на Heterogeneous Earliest Finish Time (HEFT) [7]. Для каждого доступного вычислительного устройства этот планировщик подсчитывает величину Avail(Pi), определяющую в какое время каждое из вычислительных устройств может быть доступным, т. е. все подзадачи, которые на нем выполнялись, завершатся. Причем, новая подзадача T будет выполняться на том вычислительном устройстве, которое минимизирует время её завершения.

В зависимости от ожидаемого времени выполнения $\underset{P_i}{Est}(T)$, это условие можно выразить следующей формулой:

$$\min_{P_i} (Avail(P_i) + \underset{P_i}{Est}(T)).$$

Разработчику алгоритма требуется задать функции, которые возвращают ожидаемое время выполнения задачи $\underset{P_i}{Est}(T)$. Для рассматриваемых операций оценка времени выполнения может быть легко получена, так как на каждом шаге алгоритма время выполнения операции зависит только от порядка матриц. StarPU предоставляет структуру starpu perfmodel t, полями которой являются указатели на функции, вычисляющие время выполнения подзадач. Таким образом, имеется возможность заранее задавать модель, которая описывает время выполнения задач. Также StarPU позволяет задавать модель, которая базируется на истории запуска задач на предыдущих шагах. Например, в случае с алгоритмом Холецкого, время выполнения факторизации диагонального блока можно брать таким, каким оно было получено на самом первом шаге.

Таким образом, автоматическое исследование свойств задачи и построение соответствующей вычислительной схемы, а также синтез программы решения задачи при автоматическом планировании вычислений каждой подзадачи на требуемые устройства гибридной системы позволяют при минимальных затратах вычислительных ресурсов и компьютерного времени получить решение с оценками его достоверности.

3. Заключение

Многие научно-технические задачи сводятся к решению систем линейных алгебраических уравнений большой размерности. Для эффективного решения таких больших задач требуется эффективное использование ресурсов одновременно и CPU, и GPU. В условиях приближенных исходных данных свойства компьютерных моделей априори не известны. Поэтому необходимы новые подходы в создании программного обеспечения на гибридные системы, которые бы обеспечивали пользователей достоверным компьютерным решением при эффективном использовании вычислительных ресурсов сложных гибридных систем.

Эту цель реализует автоматический решатель СЛАУ с функцией адаптивной настройки алгоритма, программы и архитектуры гибридной системы на свойства решаемой задачи для ее эффективного решения с оценкой достоверности компьютерных результатов.

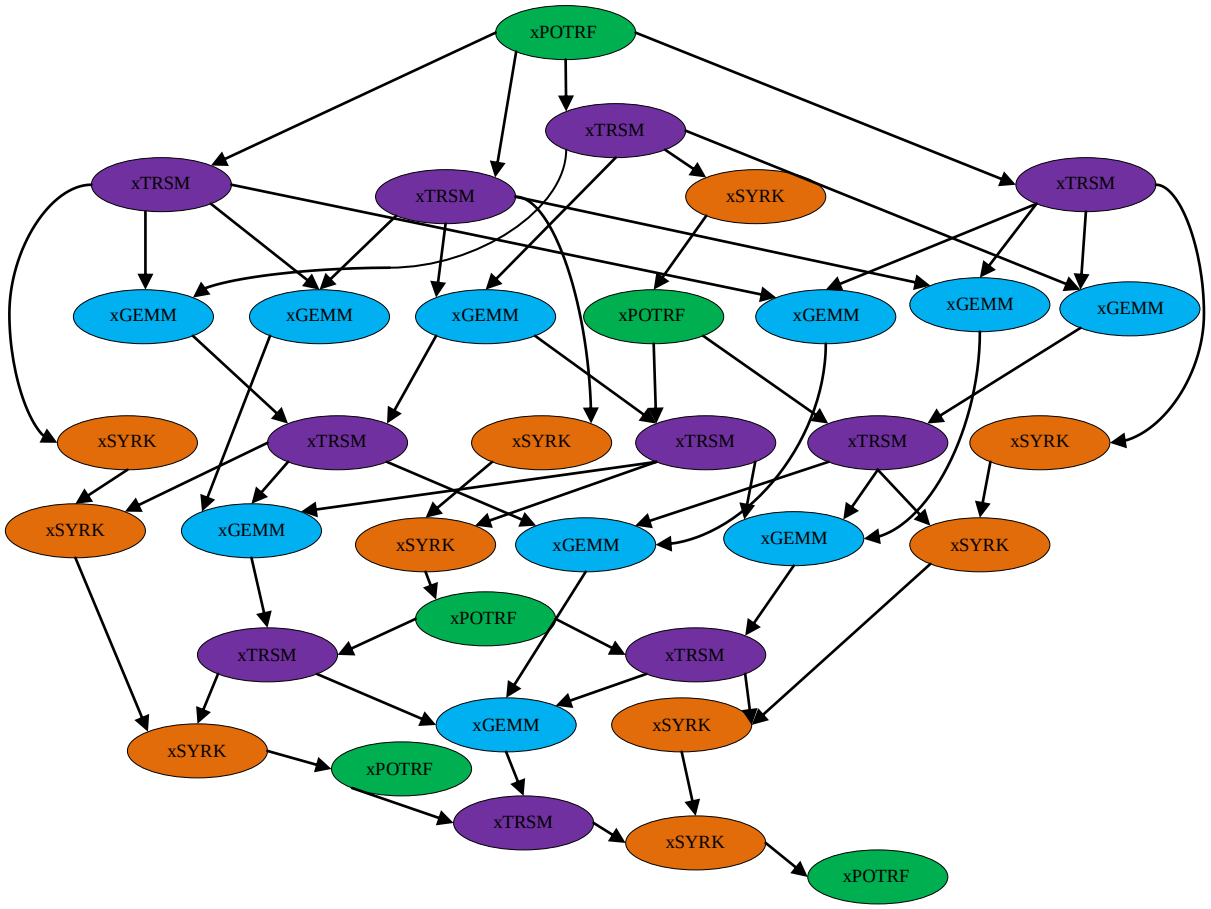


Рис. 2. Схема автоматического исследования и решения СЛАУ

Список литературы

1. ScaLAPACK Home Page. The ScaLAPACK (or Scalable LAPACK) library includes a subset of LAPACK routines redesigned for distributed memory MIMD parallel computers. [Електронний ресурс], http://www.netlib.org/scalapack/scalapack_home/
2. CULA, GPU-accelerated linear algebra library. [Електронний ресурс], http://www.culatools.com/.
3. Химич А.Н., Молчанов И.Н., Попов А.В., Чистякова Т.В., Яковлев МФ. Параллельные алгоритмы решения задач вычислительной математики. – Киев: Наук. думка, 2008. – 248 с.
4. C. Augonnet, S. Thibault, R. Namyst, P. Wacrenier. StarPU: A Unified Platform for Task Scheduling on Heterogeneous Multicore Architectures. Concurrency and Computation: Practice and Experience, Euro-Par 2009 best papers issue, 2010. Accepted for publication.
5. LAPACK - Linear Algebra PACKage. [Електронй ресурс], http://www.netlib.org/lapack/.
6. BLAS (Basic Linear Algebra Subprograms. [Електронний ресурс], http://www.netlib.org/blas/
7. H. Topcuoglu, S. Hariri, and Min-You Wu. Performance-effective and low-complexity task scheduling for heterogeneous computing. Parallel and Distributed Systems, IEEE Transactions on, 13(3):260–274, 2002.

УДК 004.724.4(045)

КУЛАКОВ Ю.А.,
КОГАН А.В.,
ПИРОГОВ А. А.

РАЗРАБОТКА И МОДЕЛИРОВАНИЕ ПРОЦЕССА БЕЗОПАСНОЙ МНОГОПУТЕВОЙ ПЕРЕДАЧИ ИНФОРМАЦИИ В МОБИЛЬНЫХ СЕТЯХ

Предложен способ многопутевой безопасной маршрутизации. Организация процесса нахождения множества путей определена с помощью модифицированного алгоритма Дейкстры. Разбиение сообщения осуществляется на основе пороговой схемы Шамира. Разработана программа и приведен пример моделирования процесса безопасной многопутевой передачи информации.

A method for secure multipath routing. Organizing the process of finding the set of paths defined by a modified Dijkstra algorithm. Splitting messages is based on Shamir's threshold scheme. The program is an example of modeling and process secure multipath transmission.

1. Введение

В настоящее время область применения беспроводных сетей значительно расширилась, появляются новые требования к используемым технологиям. В частности, актуальным вопросом является безопасность в беспроводных сетях. Это связано с тем, что прослушивание передающей беспроводной среды не составляет особого труда. Кроме того, существующие методы повышения безопасности ориентированы в основном на сети фиксированной структуры. В то же время целый ряд беспроводных технологий позволяют обеспечивать высокую мобильность узлов, для которых известные методы защиты информации, используемые в сетях со статической структурой, требуют значительных накладных расходов, а возможно и не применимы вообще.

Известные методы многопутевой маршрутизации в мобильных компьютерных сетях направлены на повышения качества передачи информации и обеспечения равномерной загрузки компьютерной сети [1,2] и, как правило, не обеспечивают требуемого уровня защиты информации. В данной работе реализован способ безопасной передачи информации с использованием многопутевой маршрутизации в виде программы.

2. Обзор существующих решений

Многопутевая маршрутизация – хорошо изученная проблема в литературе о компьютерных сетях. Некоторые протоколы маршрутизации, такие как DSR [3] осуществляют равномерную балансировку нагрузки, где маршрутизаторы распространяют трафик между несколькими путями с равными метриками маршрута. Другим примером является BitTorrent [4], протокол обмена файлами использующий peer-to-peer соединение, который использует многопутевую маршрутизацию, чтобы быстро и эффективно распределять файлы.

Многопутевая маршрутизация также была предложена [5] для уменьшения эффективности атаки, в сетях с низкой латентностью анонимности. В отличие от совокупности сетей, что препятствуют этим атакам посредством дозирования сообщения и внедрения искусственных задержек, Фейгенбаум [6] вводит новую структуру для анонимного общения, называемая как многоуровневая меш-топология, которая защищает от таких атак, не задерживая пользовательский трафик.

3. Организация процесса многопутевой безопасной передачи информации

Многопутевая безопасная маршрутизація предполагает разбиение отправляемого сообщения на части и передачу их по множеству непересекающихся максимально безопасных путей. Алгоритм нахождения максимального количества непересекающихся путей [7] является модификацией алгоритма Дейкстры с помощью которого находиться множество кратчайших непересекающихся путей. В работе [8] предложен алгоритм нахождения оптимального количества непересекающихся путей с максимальным уровнем безопасности, которая оценивается с помощью коэффициента S_{Li} характеризующего вероятность того, что путь L_i безопасен.

$$S_{Lj} = \prod_{i=1}^{N} (1 - p_i),$$

где: p_i – вероятность того, что узел скомпрометирован.

Значение вероятности p_i определяет уровень защиты i-го узла. Начальное значение p_i задается при генерации сети, затем в процессе работы оно может меняться в зависимости от атак на сеть.

Разбиение сообщения на части осуществляется с помощью пороговой схемы Шамира [9], при этом используется интерполяционный многочлен Лагранжа [10].

Сформированные пакеты передаются по мобильной сети в соответствии с найденными путями. Выбор пути осуществляется на основании теории игр [11], с помощью которой каждый игрок $i \in \{1,..., n\}$ (часть сообщения) выбирает стратегию $x_i \in S$ (путь) из набора стратегий $x = \{x_1,..., x_n\}$, гирок i получает выигрыш $f_i(x)$ [12], то есть максимально лучший путь из всего набора путей, по которому будет происходить передача части сообщения.

При доставке первого пакета к получателю включается счетчик ожидания пакетов (время ожидания равно длине самого длинного пути минус длина пути текущего пакета плюс 1). Если не доставлены все пакеты в период ожидания, то посылается сервисный пакет с просьбой переслать неполученные сообщения по другим путям, и увеличивается время ожидания пакетов на величину равную длине пути сервисного пакета плюс длина максимально длинного пути плюс 1.

После того когда все части сообщения доставлены, оно восстанавливается с помощью интерполяционного многочлена Лагранжа.

С целью повышения надежности передачи сообщений по беспроводным каналам передачи данных предусмотрен режим контроля передачи информации между смежными узлами, который заключается в следующем. После передачи кадра данных, передающий узел прослушивает передающую среду на предмет активности принимающего узла. В том случае, если принимающий узел не передает кадр данных следующему узлу, передающий узел на основании теории игр выбирает новый путь к получателю.

Если пакет попадает в скомпрометированный узел, то есть попадает к злоумышленнику, то информация об этом лавинообразно распространяется по сети.

4. Моделирование процесса безопасной многопутевой передачи информации

В рамках данной работы была разработана программа моделирования процесса безопасной многопутевой передачи информации. На первом шаге моделирования (Такт: 0) с помощью модифицированного алгоритма Дейкстры осуществляется поиск множества непересекающихся путей, для каждого из которых вычисляется значение S_{Li}, характеризующее надежность доставки информации по каждому из выбранных путей. Пути и степень их надежности, а также опции вершин отображаются в сервисном окне (рис. 1), при данной топологии сети формируется 4 маршрута с различной надежностью.

На втором шаге осуществляется разбиение исходного сообщения, на пары символов (с дополнением символов «0» в начало сообщения в случае несоответствия длины сообщения с требованиями алгоритмов разбиения/сборки сообщения). Далее полученные пары символов преобразуются в соответствии с измененным пороговым алгоритмом в функцию, из которой получаем пары значений <коэффициент> |<значение функции>. Из этих пар значений формируются пересылаемые пакеты. Преобразования отображаются в «Окне отправителя».

В данном примере исходное сообщение 092C04 с помощью функции: $(09x^2 + 2Cx + 04) \bmod 257$ делится на 3 части:

Пакет №0 с содержимым: 1|57.

Пакет №1 с содержимым: 2|128.

Пакет №2 с содержимым: 3|217.

Для передачи трех пакетов среди 4 маршрутов для передачи частей сообщения выбираются 3 наиболее надежных маршрута:

L_1: B_0->B_{10}->B_{12}->B_{18}->B_7, S_{L1}= 0,863;

L_2: B_0->B_{13}->B_{17}->B_8->B_5->B_7, S_{L1}= 0,817;

L_3: B_0->B_{14}->B_{11}->B_2->B_{16}->B_7, S_{L1}= 0,817.

На первом шаге маршрутизации пакет №0 передается по самому надежному пути L_1 в направлении узла B_{10}. Пакет №1 передается по пути L_2 в направлении узла B_{13}. Пакет №2 передается по пути L_3 в направлении узла B_{14}.

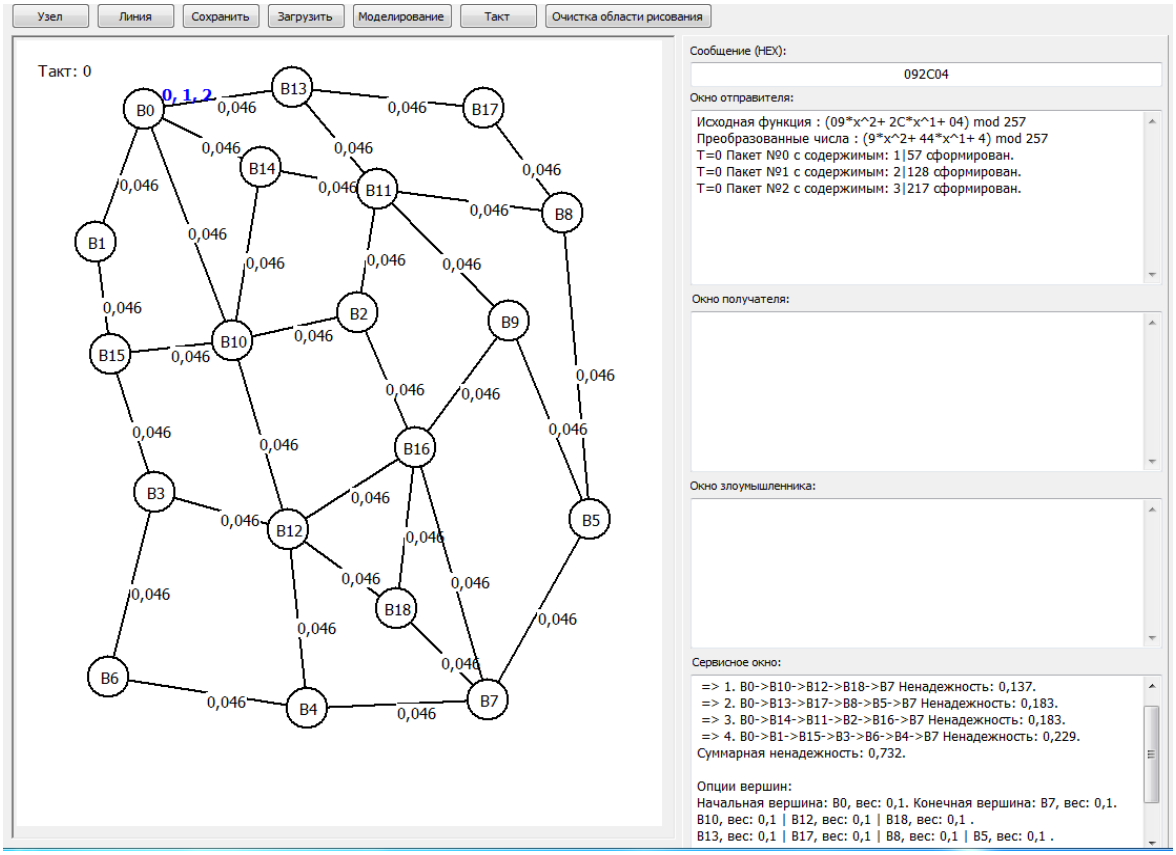


Рис. 1. Формирование множества непересекающихся путей

После 6 такта маршрутизации все пакеты сообщения достигают адресата и в окне получателя появляется следующая информация:

T=5 Ожидание всех пакетов до: 8.

T=5 Получен пакет №0 с содержимым: 1|57.

T=6 Получен пакет №1 с содержимым: 2|128.

T=6 Получен пакет №2 с содержимым: 3|217.

T=6. Все пакеты получены.

Получена формула: 217*(x-2)/(3-2)*(x-1)/(3-1)+128*(x-3)/(2-3)*(x-1)/(2-1)+57*(x-3)/(1-3)*(x-2)/(1-2)

Разложение формулы:

9*x^2+ 44*x^1+ 4

Преобразование коэффициентов:

09*x^2+ 2C*x^1+ 04

Сообщение: 092C04

Это свидетельствует о корректности передачи сообщения.

В том случае, когда злоумышленник перехватывает информацию по всем маршрутам передачи сообщений, например, в вершинах B_{10}, B_{11} и B_{17} (рис.2), он может собрать целиком все сообщение.

В моделирующей программе этот факт отражается в окне злоумышленника:

T=2 Получен пакет №0 с содержимым: 1|57

T=3 Получен пакет №1 с содержимым: 2|128

T=3 Получен пакет №2 с содержимым: 3|217

T=3 !!! Все пакеты получены !!!

При перемещении или отключении вершины, вершина в которой в данный момент находиться пакет осуществляет реконфигурацию исходного маршрута Например, при исключении вершины B_2 (рис.3) пакет №2 из вершины B_{11} перенаправляется в вершину B_9, при этом маршрут L_3 соответствующим образом реконфигурируется L_3: B_0->B_{14}->B_{11}->B_9->B_{16}->B_7. Соответствующая коррекция маршрута отражается в сервисном окне.

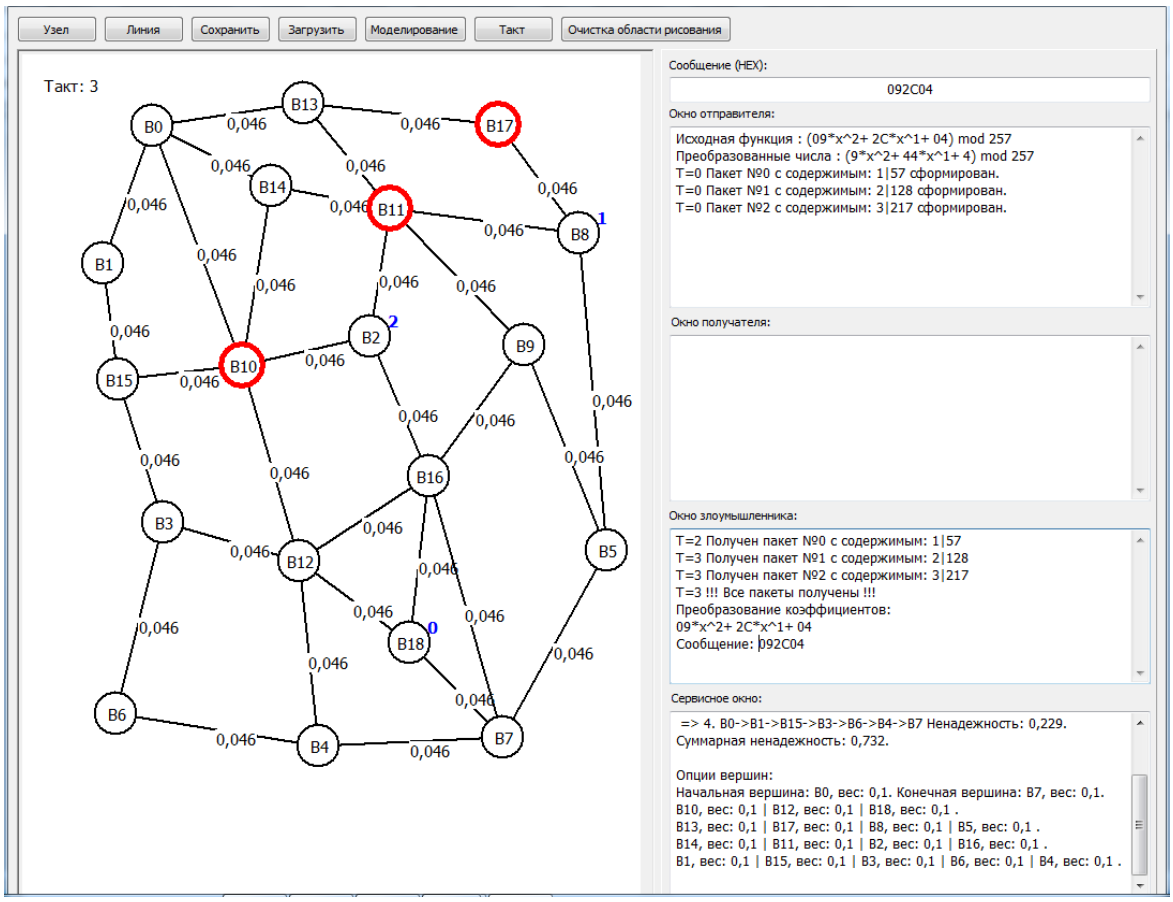


Рис.2. Злоумышленник перехватил все сообщение

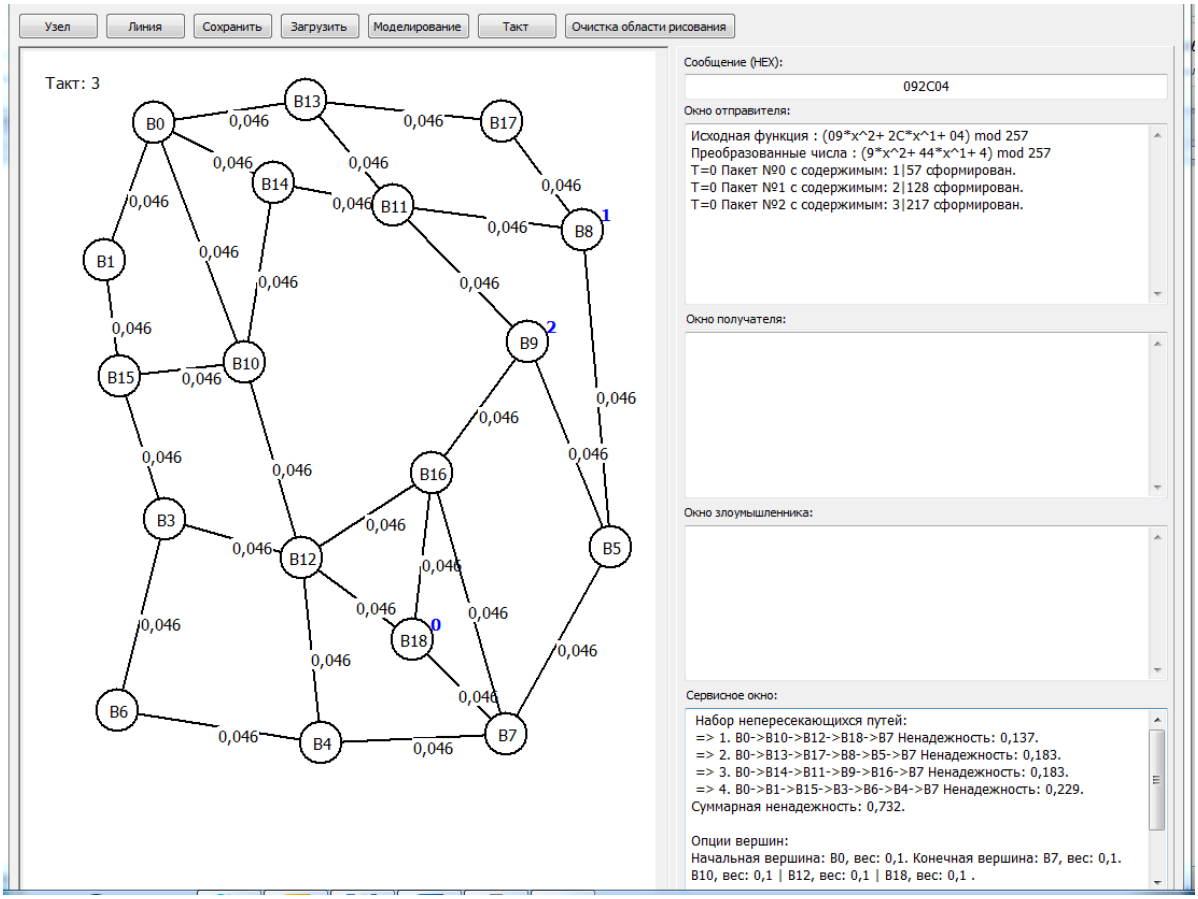


Рис. 3. Реконфигурация маршрута

Выводы

Предложенный в работе способ многопутевой маршрутизации за счет анализа надежности маршрутов позволяет обеспечить максимально безопасную передачу информационных сообщений и равномерно загрузит все каналы связи. Использование элементов теории игр также способствует выбору оптимального пути из всего набора независимых путей.

Предложенный в работе режим контроля передачи информации между смежными узлами по сравнению с известными методами квитирования передачи информации позволяет оперативно реагировать на изменения в топологии мобильной сети и своевременно осуществлять коррекцию маршрутов. Использование способа динамической распределенной маршрутизации позволяет сократить задержку передачи отдельных частей сообщения.

Чем ближе злоумышленник находиться к источнику или получателю информации, тем больше вероятность перехвата и восстановления всего сообщения. Вероятность перехвата равна единице при прослушивании всех каналов передачи информации источника или получателя информации. В этом случае необходимо принимать дополнительные меры по защите информации, например, использовать элементы BitTorrent маршрутизации.

Список литературы

1. Multipath optimized link state routing for mobile ad hoc networks Ad Hoc Networks / Jiazi Yi, Asmaa Adnane, Sylvain David and Benoît Parrein – Vol. 9, Issue 1, January 2011. – P. 28 – 47.
2. A. Tsirigos, Z. J. Haas. Analysis of multipath routing, Part 2: mitigation of the effects of frequently changing network topologies. IEEE Trans. on Wireless Communications, 2004, 3(2): 500–511.
3. D. B. Johnson, D. A. Maltz, Y.C. Hu. The dynamic source routing protocol for mobile Ad Hoc networks (DSR). draft- ietf-manet-dsr-09.txt, 2003.
4. BitTorrent protocol specification. BitTorrentSpecification, June 2011. Accessed August 2011.
5. Vitaly Shmatikov and Ming-Hsui Wang. Timing analysis in low-latency mix networks: Attacks and defenses. In Proceedings of ESORICS 2006, pages 18–33, September 2006.
6. Joan Feigenbaum, Aaron Johnson, and Paul Syverson. Preventing active timing attacks in low-latency anonymous communication. In Proceedings of the 10th Privacy Enhancing Technologies Symposium, pages 166–183, 2010.
7. Кулаков Ю.А., Максименко Е.В., Рущак О.А. Повышение уровня безопасности передачи информации в мобильных сетях // Вісн. Національного техн. ун-ту України “КПИ”: Інформатика, управління та обчислювальна техніка. – К.: ТОВ “ВЕК+”, 2007. – Вип. 47. – С.297–304.
8. Ю. А. Кулаков, А. В. Левчук. Многопутевая маршрутизация в беспроводных сетях. Вып. 4 (26), Проблеми інформатизації та управління: Зб.наук.пр.– К.: Вид-во Нац. авіац. ун-ту «НАУ-друк», 2010. – С.142-147.
9. Marti S., Giuli T., Lai K., Baker M. Mitigating routing misbehavior in mobile ad hoc networks // the 6th annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobi-Com'OO) - 2000 - Boston(MA, USA) - P.255-265
10. Кулаков Ю.А., Деревянчук А.О., Алгоритмы безопасной маршрутизации для мобильных компьютерных сетей // Проблеми інформатизації та управління: Зб.наук.пр.– К.: НАУ, 2009.– Вип.3(27) С.99-
11. T. Hui et al., “A game theory based load-balancing routing with cooperation stimulation for wireless ad hoc networks,” in Proc. the 11th IEEE international conference on high performance computing and communications, 2009, pp. 266-272.
12. M. Naserian et al., “Game theoretic approach in routing protocol for wireless ad hoc networks,” Ad Hoc Networks, vol.7, no. 3, pp. 569-578, May. 2009.

УДК 004.724.4

ГУСЕВ Е.И.,
КУЛАКОВ А.Ю.

ИСКЛЮЧЕНИЕ БЛОКИРОВАНИЯ ОБЩЕГО РЕСУРСА В РАСПРЕДЕЛЁННЫХ СИСТЕМАХ

В статье рассматривается алгоритм реализации распределенных транзакций, исключающий блокирование общего ресурса при выполнении и фиксации транзакции. Такой подход существенно улучшает пропускную способность распределенных систем.

The article provides asynchronous algorithm for distributed transactions, which excludes share blocking during transaction execution and commit. This approach significantly increases the throughput of distributed systems.

1. Введение

В настоящее время в связи с бурным ростом Grid технологий особо остро стала проблема использования общего ресурса в распределённых системах. Имеющиеся в арсенале технологии обладают низкими показателями масштабирования, как при увеличении рабочей нагрузки, так и при увеличении количества узлов распределённой системы. Повсеместной практикой сегодня является выделение процессов, затрагивающих общие ресурсы, из общего потока распределяемых операций. На практике это выглядит как единая база данных обслуживающая многих узлов масштабированной Grid системы. Эта единая база данных (БД) становится узким местом всей системы, препятствуя дальнейшему наращиванию производительности. Следует отметить, что разработчики аппаратного обеспечения давно уловили тенденцию развития серверов БД, и предлагают довольно мощные аппаратные конфигурации для сервера центральной БД. Также следует отметить то, что разработчики системы управления базой данных (СУБД) предлагают новые, так называемые, кластерные решения, целью которых так же является увеличение производительности центральной БД. Однако оба подхода имеют существенные недостатки.

Увеличение аппаратных средств не обладает гибкостью распределения ресурсов, свойственного Grid системам. Обеспечение надёжности таких систем осуществляется также традиционными методами, что менее эффективно, чем обеспечивать надежность средствами Grid.

Как правило, в современных распределенных системах сервера СУБД проектируются с многократной избыточностью аппаратных ресурсов, чтобы исключить нехватку ресурса в случае увеличения нагрузки с течением времени и застраховаться от пиковых всплесков активности.

Кластерные системы также как и Grid системы позволяют динамически наращивать мощность. Следует отметить высокие показатели надежности таких систем. К основным недостаткам кластерных систем следует отнести:

- нелинейный рост служебного трафика между узлами кластера при увеличении узлов системы (плохая масштабируемость);
- специальные требования к оборудованию и системам хранения;
- сложность развертывания и сопровождения системы и, как следствие, более высокую ее стоимость.

Учитывая все вышеперечисленное, актуальной является задача масштабируемости Grid систем, использующих общий ресурс. Целью данной работы является предложение способа, альтернативного традиционным, позволяющего эффективнее решать проблему общего ресурса. Основной идеей является оптимизация существующих алгоритмов реализации распределённых транзакций. Предлагаемый подход – использовать асинхронную фиксацию распределённых транзакций без блокирования общего ресурса, который содержит в себе потенциал ликвидации недостатков современных систем.

2. Существующие алгоритмы распределённых транзакций

На сегодняшний день самым распространенным алгоритмом распределённых транзакций является протокол двухфазной фиксации (Two-phase commit protocol) [1,2]. Этот протокол ис-

пользуется в самых распространённых на данный момент системах [3].

Алгоритм двухфазной фиксации[4] функционирует следующим образом: одна нода объявляется координатором, а остальные ноды в сети объявляются когортами. Протокол инициируется координатором после завершения последнего шага транзакции. Когорты затем отвечают сообщением согласия или прерывания в зависимости от того обработалась транзакция когортой или нет.

Фаза запроса или голосования заключается в следующем:

1. Координатор шлет запрос фиксации (query to commit message) всем когортам и ждёт ответа от всех когорт.
2. Когорты выполняют транзакцию до точки запроса на фиксацию.
3. Каждая когорта отвечает согласием или прерыванием, если когорта обнаруживает сбой, из-за которого фиксация невозможна.

Фаза завершения транзакции зависит от результата ее выполнения. В случае успеха, то есть координатор получил согласие от всех когорт во время фазы голосования:

1. Координатор шлёт сообщение фиксация всем когортам.
2. Каждая когорта завершает операцию и освобождает ресурсы, удерживаемые во время транзакции.
3. Каждая когорта шлёт подтверждение координатору.
4. Координатор завершает транзакцию когда все подтверждения получены.

В случае неуспеха, когда хотя бы одна когорта проголосовала отрицательно во время фазы голосования (или координатор дождался критического времени) осуществляются следующие операции:

1. Координатор шлёт сообщение откат всем когортам.
2. Каждая когорта отменяет операцию используя информацию UNDO и освобождает ресурсы, удерживаемые во время транзакции.
3. Каждая когорта шлёт подтверждение координатору.
4. Координатор отменяет транзакцию когда все подтверждения получены.

В настоящее время также существует алгоритм трехфазной фиксации и алгоритм Кейдара-Долева и их модификации для различных топологий распределенной системы, когда в ней существуют цепочки связей между узлами, принимающими участие в транзакции. Алгоритм трёхфазной фиксации [5]. устраняет один из ключевых недостатков предыдущего алгоритма – невозможность восстановления сбоя координатора и члена когорты во время фазы фиксации. Если сбой произошел только у координатора и не было членов когорты, получивших сообщение фиксации – такая ситуация может быть интерпретирована как отсутствие фиксации. Иными словами алгоритм повышает устойчивость распределённой системы по сравнению с базовым алгоритмом, а не повышает её производительность.

Главным недостатком алгоритма трёхфазной фиксации транзакций является то, что он не восстанавливает состояние системы в случае сегментации сети. Этот недостаток устраняется алгоритмом Кейдар и Долева [6]. В n-узловой системе возможны ситуации, когда какие из узлов в какой то момент времени недоступны. Алгоритм фиксации Paxos [7] предполагает работу с системой, в которой используется кворум «выживших» узлов системы для принятия решения о фиксации транзакции, однако, и в этом случае имеет место блокирование ресурса.

3. Алгоритм асинхронной фиксации распределённых транзакций без блокирования общего ресурса

Главным недостатком всех вышеперечисленных выше алгоритмов фиксации транзакции является то, что процесс фиксации транзакции требует блокирования ресурсов на время фиксации и ожидания ответов всех участников транзакционного взаимодействия. Это существенно повышает время отклика системы и приводит к генерации большого количества небольших сетевых пакетов обмена информацией. Альтернативой предлагается алгоритм асинхронной фиксации распределённых транзакций без блокирования общего ресурса.

Отличительной особенность алгоритма является то, что блокируются объекты только на локальной ноде, а распределённых блокировок не существует. Конфликты разрешаются задним числом, после обмена пакетами курьерской сверки.

Алгоритм состоит из следующих 5 фаз:

1. *Локальная фиксация транзакции*.
2. *Формирования пакета курьерской сверки со списком локально зафиксированных тран-*

закций.

Поскольку основные задержки при классических алгоритмах возникают при межпроцессном обмене сообщениями между нодами, логичным решением будет собирать в пакет списки локально зафиксированных транзакций, и обмениваться списками зафиксированных транзакций. Фаза содержит возможности по оптимизации, поскольку возможно определять вероятность конфликта варьируя частотой обмена, размерами пакета и производными функциями характера приложения.

3. Фоновая сверка транзакций всех узлов на каждом узле.

На данной фазе определяется был ли конфликт за общие ресурсы и какие из транзакций в случае конфликта будут а какие не будут отменены. В случае конфликта мы переходим на следующие фазы.

4. *Обработка конфликтов*.

Формирование и применение отменяющих транзакций для каждой ноды. На этой фазе осуществляется разрешение конфликта. Используя журналы (REDO) и UNDO, можно отменить любую транзакцию и вернуть данные на момент «до начала транзакции». На этой фазе появляются возможности по оптимизации алгоритма, определяя набор операций из трафика, которые целесообразно «откатывать».

5. *Формирование и фиксация (может использовать и синхронные алгоритмы) корректирующих транзакций с учетом сделанных с момента локальной фиксации изменений.* Фаза повторяет те транзакции, которые не требуют отмены на части нод, на других нодах, на которых в результате конфликта были отменены конфликтующие с неотменёнными транзакции. Фаза обеспечивает устойчивость алгоритма.

Дополнительные требования: UNDO журналы должны логировать условия выборки строк.

Условия выборки строк должны зависеть только от состояния данных распределенной системы, то есть не должно быть зависимостей от времени, случайно сгенерированного числа или состояния другой системы, которое невозможно воспроизвести в любой момент времени.

4. Сравнение алгоритмов

Одним из наиболее распространённых способов оценки производительности современных OLTP систем является тест TPC-C, который и использовался при сравнении алгоритма асинхронной фиксации транзакции со стандартным алгоритмом двухфазной фиксации.

В рамках данной работы был проведен сравнительный анализ эффективности алгоритма двухфазной фиксации транзакции и алгоритма асинхронной фиксации распределённых транзакций без блокирования общего ресурса (рис.1).

При этом следует отметить, что отличительной особенностью предложенного алгоритма является его комплиментарность к существующим синхронным алгоритмам, то есть, часть операций выполняется с помощью алгоритма асинхронной фиксации, а остальные операции выполняются с помощью классического алгоритма. Это позволяет выделить из трафика то подмножество операций, на которых вероятность конфликтов близка к нулю. Проанализировав характер загрузки TPC-C [8] теста приходим к выводу, что для операций вставки новой строки, которые составляют 90% загрузки конфликт исключен. Таким образом выигрыш на субтрафике операций вставки это фактически $\eta=0.9*(Tасинх/Tсинх)$.

На примере СУБД Oracle, где для реализации распределённых транзакций используется алгоритм двухфазной фиксации транзакции, проведено моделирование операций вставки. В первом случае будем использовать стандартный синхронный алгоритм (реализован в СУБД), а во втором случае реализовано будет триггер, отправляющий пакет на каждую вставку на удаленную СУБД средствами Oracle Streams. В дальнейшем оптимизируя алгоритм триггер

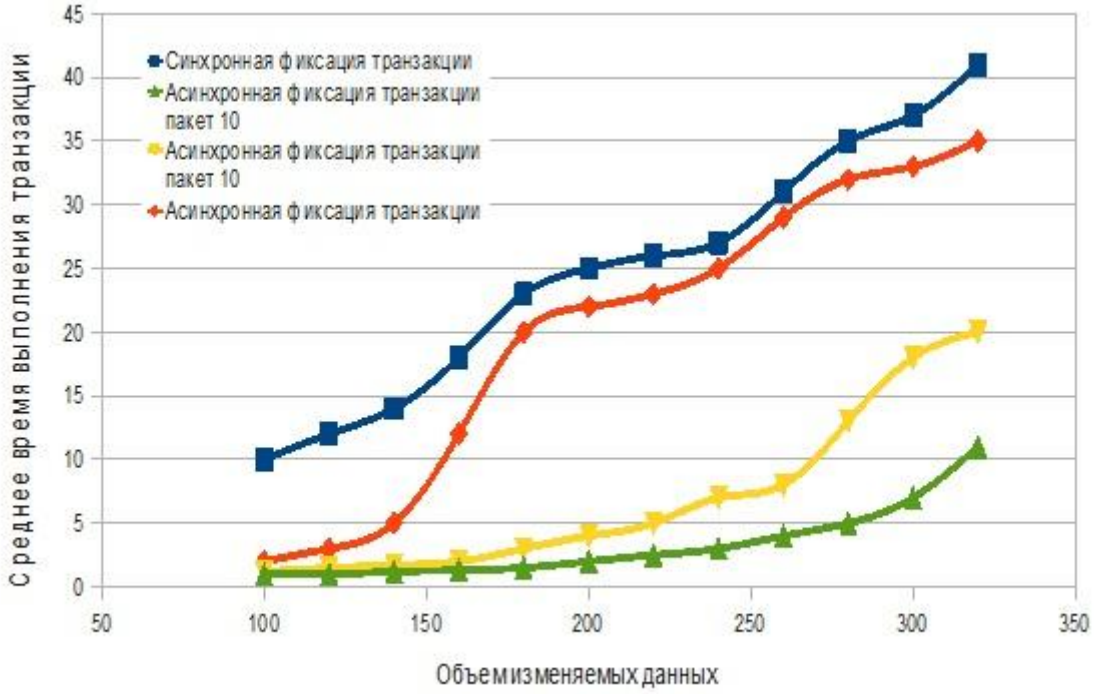


Рис. 1

будет использовать накопление в промежуточной таблице изменений таблицы и их отправку по накоплению таблицы. На рис.2 представлены результаты моделирования стандартного алгоритма двухфазной фиксации и предлагаемого алгоритма асинхронной фиксации распределённых транзакций без блокирования общего ресурса в зависимости от объема данных, передаваемых при одной транзакции.

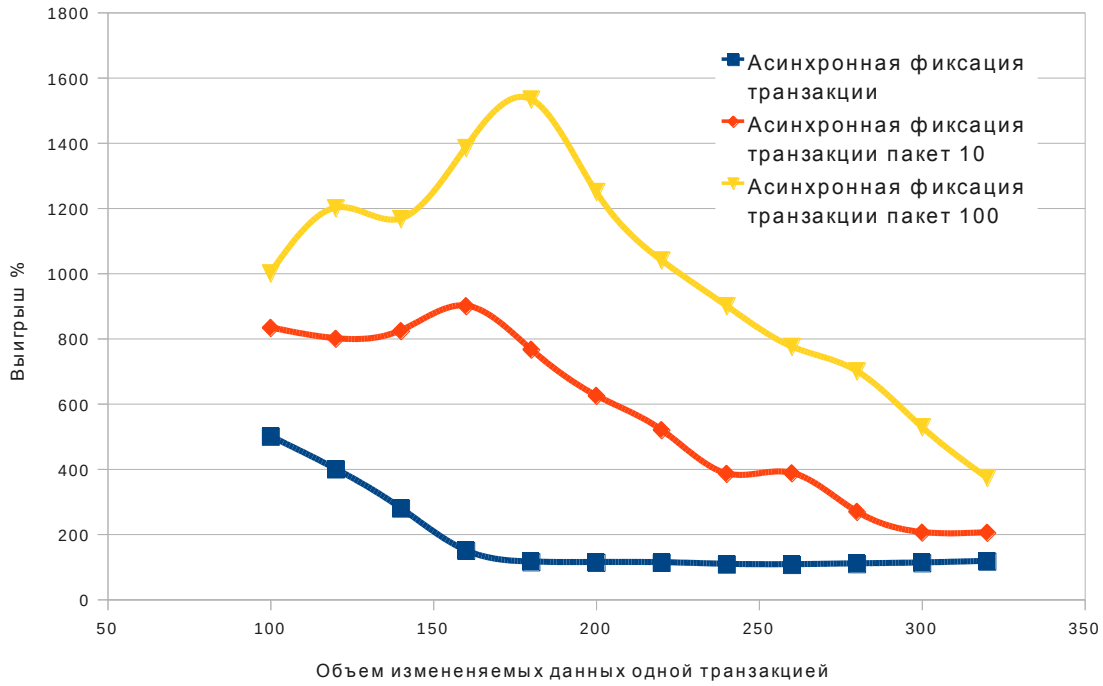


Рис. 2

5. Выводы

Алгоритм асинхронных транзакций позволяет существенно повысить пропускную способность распределённых систем. Однако в отличие от приведенного в статье исследования, где в качестве примера использовался хорошо известный тест, в реальной жизни семан-

тический уровень (отношения бизнес-сущности с табличной моделью) требует более глубокого анализа применения алгоритма. Задача сводится к определению объектов, к которым возможно применение алгоритма (Не нарушается бизнес-логика общего ресурса), и к которым целесообразно это применение (Суммарный проигрыш на фазах 4 и 5 в случае конфликта больше суммарного выигрыша при не блокировании ресурса). Важной особенностью является то, что алгоритм может использоваться одновременно с синхронными алгоритмами, при условии что на каждый объект, участвующий в распределённых транзакциях (либо непосредственно директивами внутри транзакции) будут выписаны операции на которые следует применять асинхронную фиксацию и обработчики исключений в случае конфликта. Таким образом, в реальных системах возникает задача определения оптимальной сферы применения алгоритма в зависимости от вероятности возникновения конфликтных ситуаций. Сфера применения алгоритма не ограничена базами данных использующими язык SQL и может быть существенно расширена при условии сохранения транзакционности (дискретности относительного времени в системах) и логирования запросов к СУБД.

Список литературы

1. Gerhard Weikum, Gottfried Vossen (2001): Transactional Information Systems, Chapter 19, Elsevier, ISBN 1-55860-508-8
2. Philip A. Bernstein, Eric Newcomer (2009): *Principles of Transaction Processing*, 2nd Edition, Chapter 8, Morgan Kaufmann (Elsevier), ISBN 978-1-55860-623-4
3. X/Open CAE Specification Distributed Transaction Processing: The XA Specification ISBN: 1 872630 24 3
4. Philip A. Bernstein Vassos Hadzilacos, Nathan Goodman (1987): Concurrency Control and Recovery in Database Systems, Chapter 7, Addison Wesley Publishing Company, ISBN 0-201-10715-5
5. Skeen, Dale; Stonebraker, M. (May 1983). "A Formal Model of Crash Recovery in a Distributed System". IEEE Transactions on Software Engineering 9 (3): 219–228. doi:10.1109/TSE.1983.236608
6. Keidar, Idit; Danny Dolev (December 1998). "Increasing the Resilience of Distributed and Replicated Database Systems". *Journal of Computer and System Sciences (JCSS)* 57 (3): 309–324. doi:10.1006/jcss.1998.1566
7. Jim Gray and Leslie Lamport : Consensus on Transaction Commit (2005) Microsoft Research MSR-TR-2003-96
8. Francois Raab, Walt Kohler, Amitabh Shah: Overview of the TPC Benchmark C: The Order-Entry Benchmark http://www.tpc.org/tpcc/detail.asp

УДК 683.322

РУСАНОВА О. В.,
ЯРОХ Ю. А.

ПЛАНИРОВАНИЕ ВЫЧИСЛЕНИЙ В ГЕТЕРОГЕННЫХ КЛАСТЕРНЫХ СИСТЕМАХ

В статье предлагается универсальный подход HETS к планированию вычислений, как для однородных, так и неоднородных кластерных систем. Данный подход является комбинацией списочного метода и метода, использующего дублирование. Разработана программная модель для реализации предлагаемого и наиболее известных способов планирования вычислений в гетерогенных кластерных системах. Представлены результаты исследований, которые подтверждают более высокую эффективность подхода HETS по сравнению с известными способами.

The article proposes a universal approach for task scheduling, HETS, both for homogeneous and heterogeneous cluster systems. This approach is the combination of list and duplication scheduling methods. Software for implementation of the proposed and most well-known scheduling methods in heterogeneous computing cluster systems was developed. The results of researches, which are presented in the article, confirm the higher efficiency of the approach HETS in comparison with known methods, were presented.

1. Введение

Стремительное развитие кластерных систем последних лет привело к абсолютному их доминированию среди суперкомпьютеров мира. Повышение производительности кластерных систем непосредственно связано с увеличением общего числа ядер, используемых в них. Кроме того, увеличивается количество гетерогенных кластерных систем. Это приводит к усложнению решения задачи эффективного использования ресурсов в таких системах. Совершенствование методов планирования вычислений является одной из наиболее важных задач, решение которой ведет к росту реальной производительности и эффективности работы кластерных систем.

Существующие способы планирования вычислений обычно разделяют на четыре группы: списочные, кластерные, генетические и с использованием дублирования [1]. Списочные способы планирования являются на сегодняшний день наиболее предпочтительными для применения в кластерных системах, поскольку они, в большинстве случаев, позволяют обеспечить лучшие результаты планирования и имеют меньшую временную сложность по сравнению с другими способами [2]. В то же время известно, что использование способов планирования с дублированием в кластерных системах рекомендуется к применению для сильно связных задач [1].

Поэтому в работе предлагается комбинированный подход, в котором сочетаются преимущества двух способов – списочного и с использованием дублирования, что ведет к повышению эффективности планирования в гетерогенных кластерных системах.

2. Постановка задачи планирования

В соответствии с теорией расписаний задача планирования определена, если заданы модели пользовательской задачи и кластерной системы, результаты планирования, а также используемый критерий оптимизации.

Пользовательская задача, как правило, может быть представлена в виде направленного ациклического графа DAG *(Directed Acyclic Graph)*. DAG, $G = (V, E)$, состоит из множества V вершин (v) и множества E дуг (e). Обычно DAG называют графом задачи. Вершинам такого графа соответствуют различные части программы (подзадачи), а дугам – взаимосвязи между ними. Дуга $(i, j) \in E$ между подзадачами n_i и n_j представляет межвершинную коммуникацию. Результат подзадачи n_i должен быть передан подзадаче n_j до начала ее выполнения. Подзадача без предшественников называется входной, n_{entry}, а подзадача без приемников называется выходной n_{exit}. Среди предшественников n_i, тот, который последним производит коммуникацию, является наиболее влиятельной родительской вершиной (MIP – *Most Influential Parent*) и обозначается $MIP(n_i)$ [1]. Наиболее длинный путь графа задач является критическим путем (CP – *Critical Path*) [1]. Подзадача считается готовой к выполнению, если все подзадачи предшественники были выполнены.

Вес подзадачи, n_i, обозначаемой как w_i, представляет вычислительную стоимость подзадачи. Вычислительная стоимость подзадачи, n_i на процессоре p_j, – $w_{i,j}$. Вес дуги, обозначаемой как $c_{i,j}$, представляет коммуникационную стоимость между подзадачами n_i и n_j. Коммуникационная стоимость подзадач, назначенных на один и тот же процессор, равна нулю. Средняя вычислительная и коммуникационная стоимость подзадачи n_i обозначаются $\overline{w_i}$ и $c_{i,j}$ соответственно, где первый параметр – это средняя вычислительная стоимость всех процессоров системы, а второй – это средняя коммуникационная стоимость между подзадачей n_i и всеми подзадачами приемниками.

Моменты времени раннего начала *(Earliest Start Time)* и завершения *(Earliest Finish Time)* выполнения подзадачи n_i на процессоре p_j определяются следующим образом:

$$EST(n_i, p_j) = \begin{cases} 0 & n_i = n_{entry} \\ EFT(MIP(n_i), p_k),\ p_k \in P & j = k \\ EFT(MIP(n_i), p_k),\ p_k \in P + C_{MIP(n_i),i} & j \neq k \end{cases}$$

$$EFT(n_i, p_j) = EST(n_i, p_j) + w_{i,j}$$

Результат планирования может быть представлен с помощью модифицированной диаграммы Ганта [3], в которой расписание работы каждого процессора включает не только очередь исполняемых подзадач, но и порядок пересылаемых данных всех его каналов.

Задача планирования заключается в следующем. Пусть имеется кластерная система, состоящая из K узлов. Каждый i–й узел состоит из P_i процессоров. Задано приложение из m подзадач с помощью DAG графа. Известна также топология кластерной системы, описанная с помощью неориентированного графа. Требуется найти i–й узел кластерной системы, который обеспечивает оптимальное решение заданного приложения.

В данной работе решаются следующие задачи оптимизации планирования:

1. Необходимо найти такой i–й узел кластерной системы, который обеспечивает минимальное общее время выполнения заданного приложения (T_i). Математическая модель этой задачи может быть записана следующим образом. Найти

$$\min_{i=1,R} \{T_i\} \qquad (2.1)$$

$$T_i = \sum_{i=1}^{m} \sum_{l=1}^{P_i} t_{jli} * X_{jli} + S_i + \max\{Tr, Tf_i\} \qquad (2.2)$$

где t_{jli} – время выполнения j–й подзадачи в l-м процессоре i-го узла кластерной системы; S_i – время доставки входных данных и результатов приложения в (из) i-й (i-го) узел (узла) кластерной системы; Tr – время готовности приложения к выполнению в узлах системы; Tf_i – время освобождения i-го узла кластерной системы для выполнения заданного приложения в эксклюзивном режиме; $X_{jli} = 1$, если j–я подзадача выполняется l–м процессоре i-го узла, $X_{jli} = 0$ в противном случае.

2. Необходимо найти i–й узел кластерной системы с минимальной стоимостью, который обеспечивает выполнение заданного приложения за ограниченное время *(Tz)*. Математическая модель этой задачи может быть записана следующим образом. Найти

$$\min_{i=1,R} \{C_i\} \qquad (2.3)$$

при ограничении

$$T_i \leq Tz\ (i = 1, K) \qquad (2.4)$$

В соотношении (2.3) C_i – стоимость i-го узла при выполнении заданного приложения. В данном случае вначале находится подмножество узлов, где время выполнения приложения соответствует ограничению (2.4). Затем среди них определяется узел, в котором приложение выполняется с минимальной стоимостью (соотношение (2.3)).

3. Необходимо найти i-й узел кластерной системы с минимальной стоимостью, который обеспечивает минимальное общее время выполнения заданного приложения. Математическая модель этой задачи может быть записана следующим образом. Найти узел кластерной системы, удовлетворяющий соотношениям (2.1) и (2.2) при ограничении

$$C_i \leq C_{\min} \qquad (2.5)$$

В этом случае вначале, в соответствии с соотношениями (2.1) и (2.2), определяется подмножество узлов, где приложение выполняется за минимальное время. Затем среди них, в соответствии с (2.5), выбирается узел с минимальной стоимостью.

3. Существующие подходы планирования вычислений в кластерных системах

Генетические способы планирования.

Генетические способы планирования вычислений (GA [4], AIS [5], PSGA [6], Push-Pull [7]

и другие) являются наиболее широко известными подходами, использующими направленный случайный поиск [8].

С помощью генетических способов планирования можно получить, близкие к оптимальным [9]. Однако часто такие способы сводятся к полному перебору всех возможных вариантов. В связи с этим, время работы генетических алгоритмов несоизмеримо больше времени работы алгоритмов, основанных на любых других подходах.

Поэтому, использование генетических способов планирования для реальных кластерных систем проблематично [1].

Кластерные способы планирования.

Кластерные способы планирования обычно состоят из трех фаз. На первой, сильно связные задачи группируются в кластеры, которых может быть неограниченное количество, используя линейную или нелинейную кластерную эвристику. На второй фазе кластеры назначаются на процессоры, используя коммуникационную чувствительную или коммуникационную нечувствительную стратегию. На третьей фазе кластеры сливаются (объединяются) или декластеризуются в зависимости от количества доступных процессоров. Если объединенные два кластера обеспечивают уменьшение времени выполнения, то они сливаются.

Обычно кластерное планирование на первой фазе предполагает использование неограниченного количества процессоров, исходя из своих потребностей. Поэтому, когда нет нужного их количества, то происходит перестройка кластеров под доступное количество процессоров. Конечно, это влияет на качество и на эффективность планирования. Кроме этого, процесс кластеризации в общем виде имеет экспоненциальную временную сложность.

Кластерное планирование чаще всего применяется для полносвязных однородных систем, а также для систем, где нет ограничений на выделяемые ресурсы. Применение кластерных способов планирования вычислений для неполносвязных кластерных систем с неоднородными узлами не рекомендуется [1].

Способы планирования с дублированием.

Основная идея способов планирования вычислений с дублированием (HDBS [10], LDBS [11] , DBUS [12] и другие) состоит в том, чтобы производить назначение некоторых подзадач одновременно на несколько процессоров (рис.1). Таким образом, снижается негативное влияние пересылок данных при сильной связности графа задач. Способы планирования с дублированием отличаются стратегией выбора подзадач, которые стоит продублировать. Эти подходы показывают достаточно высокую эффективность на сильно связных графах задач. Однако его эффективность падает с уменьшением связности DAG.

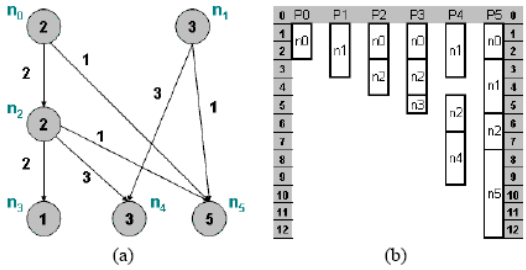


Рис. 1. Пример планирования с дублированием, где (a) граф задач, (b) диаграмма выполненного планирования с дублированием

Кроме того, способы планирования, основанные на дублировании, могут привести к необоснованной загрузке процессоров.

Поэтому использование способов планирования с дублированием в кластерных системах рекомендуется к применению на сильно связных графах задач [1].

Списочные способы планирования.

Списочные способы планирования (HEFT [13], LMT [14], MH [15], CPOP [13] и другие) являются на сегодняшний день доминантными среди эвристических способов планирования, поскольку они, в большинстве случаев, позволяют обеспечить лучшие результаты планирования и имеют меньшую временную сложность по сравнению с другими способами планирования.

Списочное планирование заключается в формировании очереди готовых подзадач с помощью одного из эвристических методов и затем оптимального назначения очереди вычислительных работ на процессоры. Все списочные способы отличаются подходами, применяемыми при выполнении указанных этапов.

На первом этапе для каждой подзадачи определяется приоритет. Основными способами определения приоритетов являются: *b-level* и *t-level* [1]. После этого подзадачи сортируются в порядке убывания их приоритетов. Способ определения приоритетов влияет на время выполнения задачи.

На втором этапе каждая готовая к выполнению подзадача назначается на процессор, исходя из политики выбора процессора.

В списочных способах планирования вычислений широко используется техника вставки, когда осуществляется назначение подзадачи на процессор между двумя уже погруженными подзадачами, если при этом имеется достаточный слот (промежуток) времени, а также техника дублирования подзадач (рис.2).

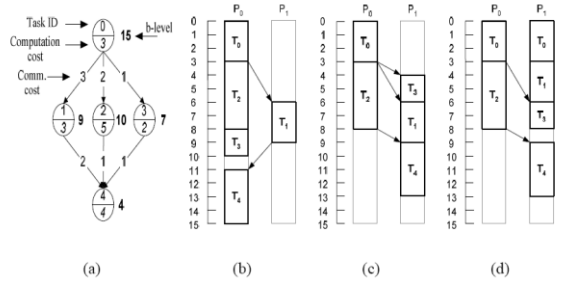


Рис.2. Примеры списочного планирования, где (a) граф задач, выполненное списочное планирование (диаграмма): (b) обычное, (c) с техникой вставки, (d) с дублированием

Для неоднородных кластерных систем более предпочтительно использовать способы списочного планирования вычислений, которые могут обеспечить субоптимальные результаты планирования при высокой скорости работы [1].

4. Способ планирования вычислений HETS

В работе предлагается новый эффективный способ планирования вычислений (HETS – *Heterogeneous Effective Task Scheduling*) в гетерогенных кластерных системах с произвольной топологией, на основе списочного подхода с использованием преобразования структуры DAG.

Целью предлагаемого способа планирования вычислений HETS является повышение реальной производительности кластерных систем с произвольной топологией. Данный подход является комбинацией списочного способа и способа, использующего дублирование, и позволяет минимизировать их недостатки [1].

Способ HETS, как и все списочные способы планирования, включает этап формирования очереди подзадач на основании их приоритетов, а также этап назначения готовых подзадач на процессоры. Однако HETS использует особую политику определения приоритетов: каждая подзадача графа имеет сложный приоритет, который состоит из статической и динамической составляющей. Статический приоритет каждой подзадачи вычисляется, так же как и в способе планирования HEFT, на основании значения *b-level* [1]. Динамический приоритет каждой подзадачи в начале погружения равен ее статическому приоритету, однако в процессе планирования он изменяется.

Способ планирования HETS явно не выполняет дублирование подзадач, однако, путем эквивалентного преобразования графа задач на втором этапе его работы, достигается эффект дублирования, при этом временная сложность такого подхода ниже.

Таким образом, HETS состоит из трех этапов, которые выполняются последовательно:

1. Начальное определение приоритетов подзадач графа (статическая приоритезация).

Статические приоритеты вершин определяются на основании критических путей до конца графа задачи:

$T_{кр.i} = b\text{-}level(n_i)$,

где $T_{кр.i}$ (CP_i) – критический путь *i*-й подзадачи до конца графа.

Значение *b-level* подзадачи определяется как сумма вычислительных и коммуникационных стоимостей самого длинного пути начиная от рассматриваемой вершины до результирующей в графе задачи. Значения *b-level* подзадачи n_i определяются как [1]:

$$b(n_i) = \overline{w}_i + \max_{n_j \in imed\_succ(n_i)} \left\{ \overline{c}_{i,j} + b(n_j) \right\},$$

где $imed\_succ(n_i)$ – множество непосредственных приемников подзадачи n_i, $\overline{w}_i$ – среднее значение веса *i*-й вершины (следствие гетерогенности кластерной системы), $c_{i,j}$ – среднее значение веса дуг от *i*-й к *j*-й вершине.

Таким образом: $Ps_i = T_{кр.i}$, где Ps_i – статический приоритет i-й вершины.

2. Преобразование графа задач.

В способе планирования HETS предусмотрены различные преобразования графа задачи. При этом выполняется «мягкое» дублирование, то есть подзадача дублируется только тогда, когда это целесообразно. Таким образом, сохраняется преимущество способов планирования с дублированием, заключающихся в эффективности их работы на сильносвязных графах задач.

Способ планирования вычислений HETS выполняет следующие преобразования графа задач [16]:

- Дублирование подзадач.

- Объединение предшествующей и последующей подзадач.
- Объединение двух параллельных ветвей графа задач.
- Объединение двух предшествующих подзадач.

Преобразование графа, путем дублирования задач, имеет смысл применять только тогда, когда подзадачи-приемники будут выполняться на тех же процессорах, что и подзадачи-источники. В этом случае веса пересылок будут аннулированы.

Эффективность данного подхода связана с соотношением потерь на дополнительные вычисления за счет дублирования подзадач и выигрыша за счет уменьшения времени пересылок. Благоприятным фактором является сохранение времени критического пути графа задач после применения процедуры дублирования.

Способ планирования вычислений HETS прдусматривает применение преобразование графа, путем дублирования, в случаях, когда:

- Подзадача является входной и имеет две или больше подзадач-наследников.
- Если подзадача имеет подзадачипредшественники и имеет две или больше подзадач-наследников, а также выполняется условие:

$$Sum\ (e_1 \dots e_n) > Sum\ (e_{01} * n \dots e_{0m} * n),$$

где $e_1 \dots e_n$ – веса дуг между подзадачей-кандидатом на дублирование и ее подзадачами-наследниками, $e_{01} \dots e_{0m}$ – веса дуг подзадачи-кандидата на дублирование и ее подзадачами-предшественниками, n – количество подзадач-наследников у подзадачи-кандидата на дублирование, m – количество подзадач предшественников у подзадачи-кандидата на дублирование.

На рис.3 продемонстрирован граф задач до применения преобразования (слева), путем дублирования подзадач, и граф задач после применения преобразования (справа), путем дублирования подзадач. Различные варианты объединения предусматривают слияние подзадач графа и инцидентных им соединений. В случае объединения выполняется «укрупнение» подзадач графа.

В способе планирования вычислений HETS применяется преобразование графа, путем объединение предшествующей и последующей подзадач в случаях, когда:

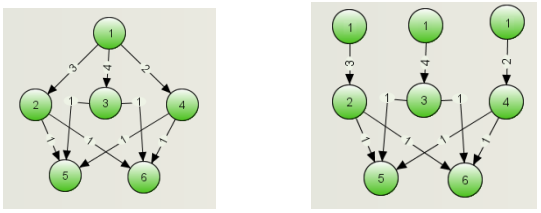


Рис. 3. Граф задач до и после применения преобразования, путем дублирования подзадач

- Было выполнено дублирование подзадач.
- Если подзадача имеет единственную предшествующую и единственную последующую вершину.

На рис. 4 продемонстрирован граф задач до применения преобразования, путем объединение предшествующей и последующей подзадач, и граф задач после применения такого преобразования. При этом веса подзадач 2, 3 и 4 увеличиваются на единицу.

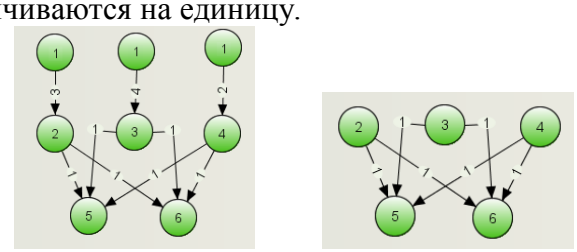


Рис. 4. Граф задач до и после применения преобразования, путем объединения предшествующей и последующей подзадач

В случае использования преобразования графа, путем объединение двух параллельных ветвей графа задач, производится слияние независимых подзадач графа. При этом веса сливаемых соединений и подзадач объединяются. В результате получается граф, в котором число соединений, а, следовательно, и количество пересылок уменьшается.

Алгоритм такой трансформации заключается в проверке целесообразности объединения соседних подзадач и в итеративном процессе их объединения. Резерв времени для каждой подзадачи определяется как разность между поздним и ранним сроками начала ее исполнения без изменения общего времени выполнения графа. Объединение проводится, если резерв времени одной ветви больше времени выполнения второй ветви, с которой она объединяется. Процесс продолжается итеративно до тех пор, пока уменьшается время выполнения задачи.

В случае объединения двух предшествующих подзадач производится слияние независимых подзадач и дуг графа, при этом их веса

объединяются. В результате уменьшается количество пересылок в графе.

Уменьшение количества пересылок приводит к сокращению числа конфликтов при пересылках в системе, однако при этом увеличивается объем самих пересылок (которые объединились в одну), а также объем вычислений (объединение вершин графа). Следовательно, эффективность такого преобразования зависит от производительности процессоров в системе, а также от числа и пропускной способности ее каналов.

3. Назначение задач процессорам и доопределение приоритетов задач (динамическая приоритезация).

До начала назначения на процессоры все динамические приоритеты подзадач равны их статическим приоритетам.

Перед выбором подзадачи для назначения осуществляется доопределение приоритетов, а точнее обновление их динамической составляющей:

- Для всех доступных (готовых) подзадач для всех процессоров определяется текущий запас времени по формуле:

$$FT_{i,j} = T_{кр.гр} - T_{кр.\,i} - T_{тек.j},$$

где $FT_{i,j}$ – запас времени i-й подзадачи на j-м процессоре и $T_{тек.j}$ – текущий такт j-го процессора.

- Для всех доступных (готовых) подзадач определяется средний запас времени по формуле:

$$FT_i = Sum\,(FT_{i,1} \ldots FT_{i,n}) / n,$$

где n – количество процессоров компьютерной системы.

- Если $FT_i < 0$, то $P_i = Ps_i + |FT|$, где P_i – комплексный приоритет i-ой вершины.

Подзадача с наибольшим комплексным приоритетом P выбирается для назначения. При равенстве комплексных приоритетов выбирается подзадача с большей связностью, а при равенстве связностей – с меньшим весом.

В HETS при выборе процессора для назначения учитываются все процессоры, даже если некоторые из них заняты в момент назначения.

Назначение подзадач на процессоры основано на выборе того из них, который обеспечивает минимальное время окончания выполнения выбранной подзадачи (EFT). Значение EFT подзадачи на конкретном процессоре включает время доставки для нее исходных данных с учетом топологии кластерной системы, а также время выполнения подзадачи на процессоре с учетом его производительности.

5. Методика сравнения способов планирования вичислений

Сравнение способов планирования вычислений основывается на следующих метриках:

- *Scheduling Length (SL). SL – makespan* определяется временем выполнения задач графа на кластерной системе.
- *Scheduling Length Radio (SLR).* Значение *SLR* графа задач фактически определяет, во сколько раз *SL* больше CP_{MIN} , и вычисляется по формуле:

$$SLR = \frac{makespan}{\sum_{n_i \in CP_{MIN}} \min_{p_j \in Q} \{w_{i,j}\}}$$

где CP_{MIN}. – сумма минимальных вычислительных стоимостей подзадач. Способ планирования вычислений с минимальным значением *SLR* является лучшим, с точки зрения времени выполнения задачи.

- *Speedup (SU)* или ускорение. Значение *SU* определяется отношением времени выполнения задачи на однопроцессорной системе ко времения ее выполнения на параллельной системе:

$$Speedup = \frac{\min_{p_j \in Q} \{\sum_{n_i \in V} w_{i,j}\}}{makespan}$$

- *Number of Occurrences of Better Quality of Schedules (NOBQS).* Данный показатель определяет число случаев лучшего качества планирования. Выполняется подсчет лучшего, равного и худшего качества планирования каждого из способов планирования в сравнении со всеми другими по отдельности и вместе. Более качественным является планирование, обеспечивающее меньшее время выполнения графа задач *(SL).*
- *Running time (RT). RT* определяет время работы самого алгоритма планирования. При статическом планировании этот показатель учитывается лишь при выборе алгоритмов планирования, с равными значениями *SLR*. Тот из них, который имеет меньшее *RT*, является более предпочтительным для практического применения.

6. Экспериментальные результаты проведенных исследований

Основываясь на рекомендациях использования и эффективности применения способов планирования вычислений в гетерогенных кла-

стерных системах [1], для исследования были отобраны три широко известных списочных способа планирования (Heterogeneous Earliest Finish Time – HEFT [13], Levelized Min Time – LMT [14], Critical Path On a Processor – CPOP [13]) и два наиболее эффективных способа планирования с дублированием (Heterogeneous Duplication-Based Scheduling – HDBS [10], Levelized Duplication-Based Scheduling – LDBS [11]).

Для сравнения вышеупомянутых способов планирования с предлагаемым подходом HETS разработан программный комплекс Owl.

Граф топологии гетерогенной кластерной системы для проведения сравнения способов планирования представлен на рис.5.

При проведении данных исследований оценивается качество планирования различных способов планирования на множестве случайно сгенерированных графов задач. Для этого используется генератор, описанный в работе[13], со следующими входными параметрами:

- v = {30, 40, 50, 60, 70, 80, 90, 100} – количество подзадач графа.
- a = {0.5, 1.0, 2.0} – форма графа.
- *out_degree* = {1, 2, 3, 4, 5} – выходная степень подзадачи.
- *CCR* = {0.1, 0.5, 1.0, 5.0, 10.0} – соотношение суммарной коммутационной

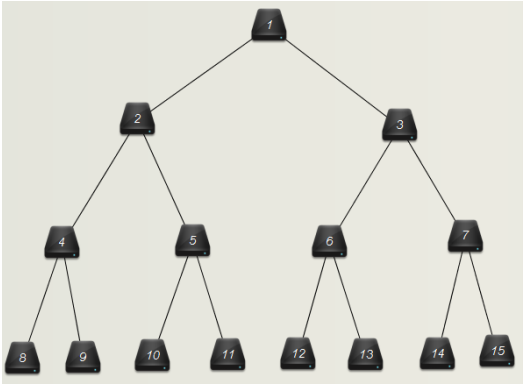


Рис. 5. Граф топологии кластерной системы для проведения исследований

стоимости к суммарной вычислительной стоимости графа задач.

- β = {0.25, 0.5, 0.75, 1.0} – диапазон отношений вычислительных стоимостей процессоров к самому производительному из них.

Все возможные комбинации этих входных параметров включают 2400 типов DAG. Кроме этого, для каждого из типов генерируется 10 графов для усреднения результатов планирования по каждому типу DAG. Таким образом, для сравнения способов планирования всего сгенерировано 24000 графов задач.

Результаты проведенных экспериментов представлены на рис.6-9 и в табл.1.

На основании результатов, приведенных на рис.6, можно отметить следующее:

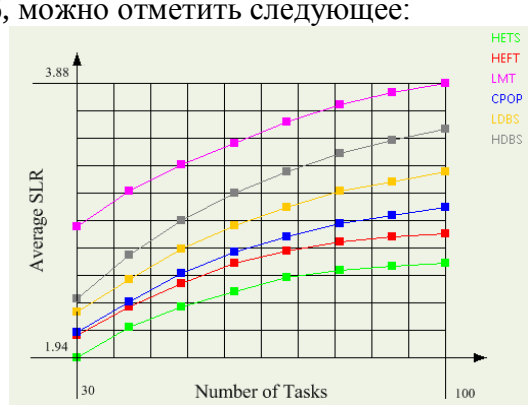


Рис.6. Зависимость среднего SLR от количества вершин графа задач

- По результатам проведенного эксперимента лучшим способом планирования, обеспечивающим минимальное среднее значение SLR из диапазона количества задач графа {30 - 100}, является HETS.
- По результатам сравнения средних значений *SLR*, исследуемые способы планирования можно расположить в порядке ухудшения характеристики *SLR* следующим образом: {HETS, HEFT, CPOP, LDBS, HDBS, LMT}.

Анализируя результаты, приведенные на рис.7, можно отметить следующее:

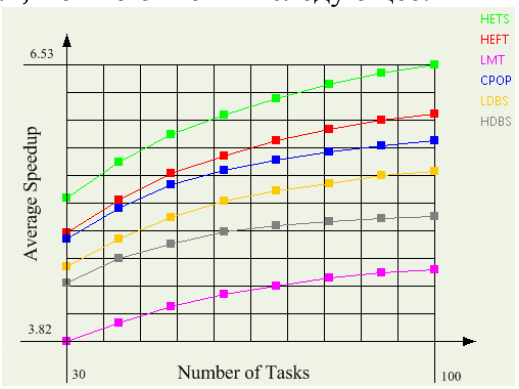


Рис. 7. Зависимость среднего SU от количества вершин графа задач

- По результатам проведенного теста лучшим способом планирования, обеспечивающим максимальное среднее значение *SU*, является HETS.
- По результатам сравнения значений среднего *SU*, исследуемые способы планирования можно расположить в порядке ухудшения характеристики *SU* следующим образом: {HETS, HEFT, CPOP, LDBS, HDBS, LMT}.

Анализируя зависимости среднего *SLR* от *CCR* (рис.8), можно отметить следующее:

- По результатам проведенного теста лучшим способом планирования является HETS, который обеспечивает минимальное

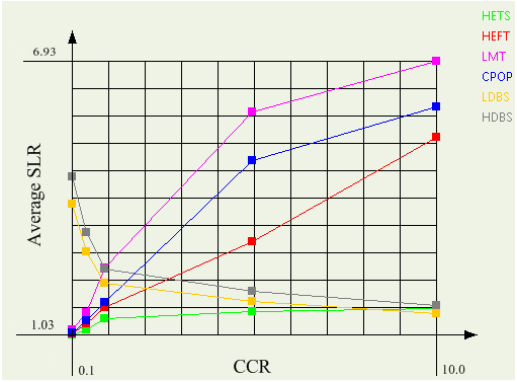


Рис. 8. Зависимость среднего SLR от CCR

среднее значение *SLR* практически на всем диапазона *CCR* {0.1 - 10.0} за исключением, когда *CCR* = {8.0, 9.0, 10.0}, где HETS уступает способу планирования LDBS.

Анализируя полученные зависимости среднего *RT* от количества вершин графа задач (рис.9), можно отметить следующее:

- По результатам проведенного теста лучшим способом планирования с минимальным средним значением *RT* является HEFT.
- По результатам сравнения средних значений *RT*, исследуемые способы планирования можно расположить в порядке ухудшения характеристики *RT* следующим образом: {HEFT, CPOP, HETS, LMT, LDBS, HDBS}.

Попарно сравнивая средние значения *SL* исследуемых способов планирования между собой (табл.1), можно отметить следующие:

- Способ планирования HETS для 91% графов задач обеспечил минимальное значение *SL*, что является лучшим результатом среди всех исследуемых способов планирования.

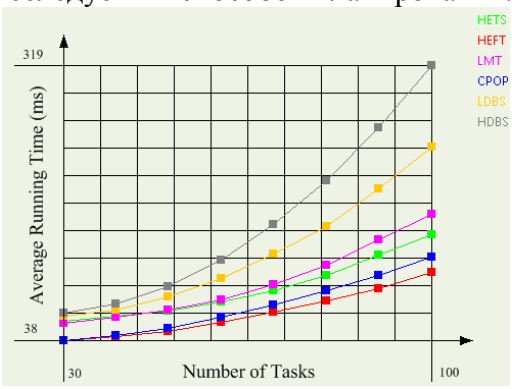


Рис. 9. Зависимость среднего RT от количества вершин графа задачи

- По результатам сравнения значений *SL*, исследуемые способы планирования можно расположить в порядке ухудшения характеристики *NOBQS* следующим образом: {HETS, HEFT, CPOP, LDBS, HDBS, LMT}.

Среднее значение *SLR* на 24000 сгенерированных графах задач для HETS = 2.39, для HEFT = 2.57, для LMT = 3.48, для CPOP = 2.68, для LDBS = 2.87, для HDBS = 3.09.

Таким образом, исследуемые способы планирования, в порядке убывания эффективности планирования по результатам экспериментов, можно расположить следующим образом: {HETS, HEFT, CPOP, LDBS, HDBS, LMT}. При этом среднее значение *SLR* у HETS лучше: на 8%, чем у HEFT; на 12%, чем у CPOP; на 20%, чем у LDBS; на 30%, чем у HDBS; на 45%, чем у LMT.

Табл.1. Результаты попарного сравнения значений SL способов планирования (NOBQS) на множестве случайно сгенерированных графов задач

		HETS	**HEFT**	**LMT**	**CPOP**	**HDBS**	**LDBS**	
HETS	Лучше		20657	23776	22153	21671	21092	91%
	Равно	*	197	114	378	684	575	2%
	Хуже		3146	110	1469	1645	2333	7%
HEFT	Лучше	3146		21939	19067	17167	13912	63%
	Равно	197	*	362	601	454	623	2%
	Хуже	20657		1699	4332	6379	9465	35%
LMT	Лучше	110	1699		2152	10061	6639	17%
	Равно	114	362	*	340	459	613	2%
	Хуже	23776	21939		21508	13480	16748	81%
CPOP	Лучше	1469	4332	21508		16043	12716	47%
	Равно	378	601	340	*	458	513	2%
	Хуже	22153	19067	2152		7499	10771	51%
HDBS	Лучше	1645	6379	13480	7499	*	8293	31%

	Равно	684	454	459	458		719	2%
	Хуже	21671	17167	10061	16043		14988	67%
LDBS	Лучше	2333	9465	16748	10771	14988		45%
	Равно	575	623	613	513	719	*	3%
	Хуже	21092	13912	6639	12716	8293		52%

7. Выводы

В статье предложен новый эффективный способ планирования вычислений HETS для гетерогенных кластерных систем с произвольной топологией. Данный подход является комбинацией списочного метода и метода, использующего дублирование. HETS позволяет объединить преимущества указанных методов и минимизировать их недостатки. Разработана программная модель для реализации предлагаемого и наиболее известных способов планирования вычислений для гетерогенных кластерных систем. Представлены результаты исследований, которые подтверждают более высокую эффективность подхода HETS по сравнению с известными способами.

Предложенный способ планирования вычислений HETS является универсальным и может быть использован для повышения реальной производительности, как масштабируемых систем с массовым параллелизмом, так и гетерогенных кластерных систем.

Список литературы

1. Young Choon Lee, «Problem-Centric Scheduling for Heterogeneous Computing Systems», September 2007.
2. Y. K. Kwok and I. Ahmad, «Benchmarking the Task Graph Scheduling Algorithms», Proc. First Merged Int'l Parallel Symposium/Symposium on Parallel and Distributed Processing (IPPS/SPDP '98), pp. 531–537, 1998.
3. G. Loutsky, O. Rusanova, «Scheduling Problems on the Parallel and Distributed Systems - an Overwiew» – Poland, Rzeszow, tom 1, pp.101-105 (Engl), 2000.
4. E. S. H. Hou, N. Ansari, and H. Ren, «A genetic algorithm for multiprocessor scheduling», IEEE Trans. Parallel and Distributed Systems, vol. 5, no. 2, pp. 113–120, 1994.
5. R. L. King, S. H. Russ, A. B. Lambert, and D. S. Reese, «An artificial immune system model for intelligent agents», Future Generation Computer Systems, vol. 17, no. 4, pp. 335–343, 2001.
6. M. K. Dhodhi, I. Ahmad, A. Yatama, «An integrated technique for task matching and scheduling onto distributed heterogeneous computing systems», J. Parallel and Distributed Computing, 62: 1338-1361, 2002.
7. S. C. Kim and S. Lee, «Push-pull: Guided search DAG scheduling for heterogeneous clusters», Proc. Intl. Conf. Parallel Processing (ICPP'05), 2005.
8. L. Wang, H. J. Siegel, V. P. Rowchoudhry and A. A. Maciejewski, «Task matching and scheduling in heterogeneous computing environments using a genetic algorithm-based approach», J. Parallel and Distributed Computing, 47: 8-22, 1997.
9. Cristina Boeres, Jos´e Viterbo Filho and Vinod E. F. Rebello, «A cluster-based strategy for scheduling task on heterogeneous processors», Proc. 16th Symp. on Computer Architecture and High Performance Computing (SBAC-PAD), 2004.
10. Y. K. Kwok, «Parallel program execution on a heterogeneous PC cluster using task duplication», Proc. 9th Heterogeneous Computing Workshop (HCW), pp. 364–374, May 2000.
11. A. Dogan and R. Ozguner «LDBS: A Duplication Based Scheduling Algorithm for Heterogeneous Computing Systems», Proc. Int'l Conf. Parallel Processing, pp. 352–359, August 2002.
12. D. Bozdag, U. Catalyurek and F. Ozguner, «A task duplication based bottom-up scheduling algorithm for heterogeneous environments», Proc. Int'l Parallel and Distributed Processing Symp., April 2005.
13. H. Topcuoglu, S. Hariri and M. Wu, «Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing», IEEE Trans. Paralleland Distributed Systems, vol. 13, no. 3, pp. 260–274, March 2002.
14. M. Iverson, F. Ozguner and G. Follen, «Parallelizing existing applications in distributed heterogeneous environments», Proc. Heterogeneous Computing Workshop, pp: 93-100, 1995.
15. H. EI-Rewini and T. G. Lewis, «Scheduling parallel program tasks onto arbitrary target machines», J. Parallel and Distributed Computing, 9: 138-153, 1990.
16. O.V.Rusanova, A.P.Shevelo, «List Scheduling Algorithm Modification for MPP Systems». Вісник НТУУ «КПІ» Інформатика, управління та обчислювальна техніка, Київ 2006 р.№45.-238 с.- С101-111.

УДК 519.687.1

ТЕЛЕНИК С.Ф.,
РОЛИК А.И.,
САВЧЕНКО П.С.

АДАПТИВНЫЙ ГЕНЕТИЧЕСКИЙ АЛГОРИТМ ДЛЯ РЕШЕНИЯ КЛАССА ЗАДАЧ РАСПЕРЕДЕЛЕНИЯ РЕСУРСОВ ЦОД

Предложен адаптивный вариант генетического алгоритма, основанный на циклическом применении двух генетических алгоритмов, решающих задачи параметрической и алгоритмической адаптации, что позволяет определить стратегию выбора параметров генетических алгоритмов и вероятность их применения. Приведена последовательность этапов работы алгоритма, предложены модифицированные операторы мутации и кроссовера, введены понятия наград и производительности, которые позволили регламентировать последовательности применения генетических операторов и корректировать вероятность их применения. Эффективность предлагаемого алгоритма продемонстрирована на примере решения задачи распределения виртуальных машин в центре обработки данных.

An adaptive version of the genetic algorithm is proposed. Algorithm is based on the simultaneous use of two genetic algorithms that solve the problem of parametric and algorithmic adaptation. This allows you quickly select and adjust the strategy selection parameters for genetic algorithms and the likelihood of their use. Sequential steps of the algorithm are given, introduced the modified parameters of mutation and crossover, introduced the concepts of rewards and performance parameters that are allowed to regulate the sequence of genetic operators, and to adjust the probability of their use.

Введение

Для эффективного использования дорогостоящих вычислительных ресурсов центров обработки данных (ЦОД), хранилищ данных и других информационных и телекоммуникационных ресурсов необходимо, чтобы соответствующие системы управления ИТ-инфраструктурой обеспечивали рациональное распределение виртуальных машин (ВМ), осуществляли управление нагрузкой и решали множество других подобных задач. В [1] показаны преимущества применения генетических алгоритмов (ГА) по сравнению с эвристическими методами для решения задач такого рода. В тоже время требует прояснения ряд фундаментальных проблем, обусловленных спецификой этих алгоритмов, поскольку ГА одновременно используют несколько типов генетических операторов: унарных, бинарных и множественных. Трудно подобрать стратегию генерации значений вероятностей использования отдельных операторов таким образом, чтобы их применение давало положительный результат на протяжении всего времени работы ГА. Кроме того, каждый из операторов имеет множество параметров, оказывающих влияние на результаты работы алгоритма, и найти оптимальные значения этих параметров достаточно сложная задача. Решение этих проблем ГА в общем виде не представляется возможным, поэтому необходимо разработать эффективную стратегию подбора параметров операторов и определения вероятностей применения этих операторов в процессе эволюции ГА. Решению такой задачи и посвящена данная статья.

Анализ исследований и публикаций

В работах [2, 3] показано, что применение ГА для решения задач распределения вычислительных ресурсов позволяет получать достаточно производительные решения. В этих исследованиях сформулирован список проблем, которые необходимо выяснить при использовании ГА для решения задач различной размерности и ограничений.

В [4, 11] предложена идея адаптации генетических операторов и подстройки вероятностей их использования. Недостатком этих работ является адаптация только одного оператора — кроссовера и использование возможных подходов к адаптации независимо друг от друга.

В [1] предложен управляемый генетический алгоритм (УГА), позволяющий корректировать параметры работы алгоритма на всех этапах

решения задачи. Помимо этого УГА решает проблемы классического ГА: вырождение популяции, попадание в локальные экстремумы и т. д. Главными недостатками УГА являются необходимость участия администратора и привязка к узкому классу задач.

Для решения этих проблем предлагается использовать разработанный в данной работе адаптивный генетический алгоритм (АГА). АГА является обобщенным ГА, способным динамически изменять вероятности использования и значения параметров каждого из операторов.

Целью работы является разработка разновидности генетического алгоритма, способного самостоятельно изменять вероятности использования операторов ГА и значения параметров этих операторов в зависимости от результатов, получаемых в ходе решения задачи, и характера решаемых задач, а также проверка эффективности работы предлагаемого АГА на примере задачи распределения ресурсов ЦОД.

Основные положения

В задачах распределения ресурсов часто необходимо использовать целевые функции, являющиеся нелинейными и не унимодальными, ограничения – нелинейные и невыпуклые, при этом переменные могут быть булевыми, целыми, непрерывными либо смешанными. Во многих случаях реализация традиционных стратегий требует огромных объемов вычислений, а «время жизни» полученного результата зачастую невелико, поскольку ситуация безостановочно изменяется и постоянно требует новых решений. В таких случаях особый интерес представляет применение генетических алгоритмов, имеющих ряд привлекательных свойств, обусловленных их природой, которые могут дать хорошие результаты при решении задач с такими свойствами.

Идея, реализуемая в этой работе, довольно проста: можно ли придать генетическим алгоритмам свойства классических алгоритмов поиска, которые были популярными при решении инженерных задач в 70-80-х годах прошлого столетия? Генетический алгоритм, «стреляя» по области решений и отбирая лучшие популяции, постоянно улучшает решение. То же самое делают и алгоритмы поиска. Однако последние используют некоторую дополнительную информацию для увеличения скорости сходимости. Нельзя ли наделить алгоритмы поиска свойствами генетических алгоритмов? В [1] был предложен УГА, использование которого позволяет получать хорошие результаты. Скорость сходимости увеличивается за счет подбора вероятностей основных операций ГА на основе результатов, полученных на предыдущих шагах работы. В УГА специальные правила, используя параметры давления отбора, скорость сходимости и коэффициенты прироста популяции позволяют изменять вероятности применения генетических операторов и тем самым увеличить скорость сходимости.

Следующим естественным шагом должна быть адаптация ГА путем отбора тех же вероятностей, но на основе некоторого критерия оценки популяции. Таким критерием может служить вероятность получения оптимального решения на основе популяции либо скорости сходимости.

Поиск эффективной стратегии выбора частоты применения генетических операторов и параметров этих операторов для решения задач, возникающих при управлении ИТ-инфраструктурой, является прямой задачей адаптации генетического алгоритма.

Выделяют три типа адаптации: структурную, параметрическую и адаптацию алгоритма [13]. Структурная адаптация предполагает изменение структуры объекта. Для класса задач, решаемых в данной работе, такой возможности не существует, поэтому будут использованы только два оставшихся типа адаптации. Параметрическая адаптация позволяет корректировать параметры работы в процессе поиска решения, в то время как адаптация алгоритма подразумевает внесение изменений в ход алгоритма.

Общая постановка задачи адаптации ГА сводится к минимизации некоторой функции F, которая служит критерием адаптации для ГА и зависит от таких параметров как: типы операторов, которые следует использовать в эволюционном процессе, частота применения этих операторов и значения параметров, с которыми применяются операторы. Определим функцию адаптации следующим образом

$$F(M, C, \alpha_M, \alpha_C, \beta_M, \beta_C),$$

где M, C – параметры, регламентирующие применение операторов мутации и кроссовера, и принимающие значения из множества {0, 1}, причем равенство параметра нулю означает, что данный оператор не участвует в эволюционном процессе, а единица означает, что опера-

тор принимает участие в процессе эволюции; α_M, α_C – частоты использования операторов, принимающие значения из интервала [0;1]; β_M, β_C – множества возможных значений операторов мутации и кроссовера.

Нахождение явного вида функции F даже для простых случаев требует огромного количества вычислений, что сведет на нет все преимущества ГА. Использование структурной адаптации в рамках решения задач управления ИТ-инфраструктурой невозможно, поэтому при осуществлении адаптации предлагается использовать некоторую информацию касательно свойств функции F . Такой информацией будут типы операторов и параметры этих операторов, с которыми они принимают участие в эволюционном процессе.

В данной работе решение задачи создания АГА осуществляется последовательным циклическим использование двух ГА. Первый применяется с целью параметрической адаптации и используется для настройки параметров генетических операторов, увеличивающих скорость сходимости. Второй служит для алгоритмической адаптации и осуществляет подстройку ГА в зависимости от результатов, получаемых в ходе решения задачи. Таким образом, в работе используются два механизма адаптации операторов АГА, один из которых обеспечивает динамическое изменение значений вероятностей применения операторов, а второй подстраивает значения его параметров. В первом случае операторы, которые в процессе работы позволили улучшить решение, чаще применяются при генерации следующих поколений. Во втором — с момента запуска АГА контролируются и изменяются значения параметров операторов.

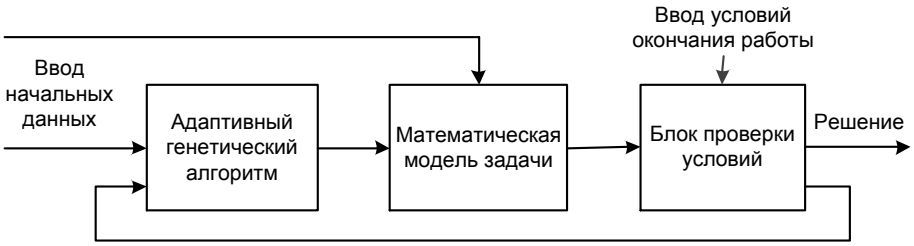


Рис. 1. Схема работы адаптивного генетического алгоритма

Схема работы предлагаемого АГА представлена на рис. 1. Математической моделью задачи являются задачи линейного программирования. К этому классу задач относится большинство задач, возникающих при управлении ИТ-инфраструктурой, например, задача распределения ресурсов, задача эффективного размещения файлов на серверах хранилища данных и т. п. Блок проверки условий служит для задания критериев остановки работы АГА. Это могут быть ограничения по времени работы либо по количеству эпох эволюционирования. Блок АГА – программная реализация предлагаемого алгоритма.

Как правило, для адаптации ГА [8] под решаемые задачи используется некоторый параметр управления [9]. Эффективность детерминистического управления была доказана для некоторых задач [4], однако его обобщение является проблематичным. Адаптивное управление [5] использует обратную связь, чтобы определить, как должны измениться параметры. Под адаптивным управлением [11] понимается введение дополнительной информации, позволяющей корректировать поведение операторов.

На сегодняшний день основные усилия исследователей ГА направлены на одновременную адаптацию только одного параметра. Наиболее полное сочетание форм управления [9] было представлено в работе [7], адаптация производилась одновременно по трем параметрам: вероятности мутации, кроссовера и размера популяции. Остальные формы адаптивных алгоритмов представлены в работах [6, 10, 12].

В данной работе предлагается обобщенная схема ГА, способного по ходу своей работы решать задачи параметрической и алгоритмической адаптации – динамически адаптировать как вероятность использования, так и значения параметров каждого из операторов. Вероятности применения операторов адаптированы детерминистическим способом. В этом случае все операторы применяются поочередно, а оператор, который в процессе работы позволил улучшить решение, автоматически будет использован в следующих эпохах. Таким образом, операторы, способствующие сходимости задачи, при генерации следующих поколений применяются чаще.

Приведем базовые определения и введем некоторые обобщения для классических операторов.

Кодирование данных. Каждая хромосома представляет собой последовательность бит. Все хромосомы $\text{chrom} = (b_1, ..., b_i, ..., b_n)$, $n = 2^k$, $k > 0$, $b_i \in \{0,1\}$, для всех $i = \overline{1, n}$, имеют длину 2^k, где k константа.

Мутация. Стандартный оператор мутации изменяет гены, путем инвертирования. Этот тип оператора имеет параметр *alter_rate*, который определяет количество генов в хромосоме, подлежащих инвертированию.

Предлагается традиционный алгоритм мутации усовершенствовать таким образом, чтобы вместо решения задачи для каждого гена о необходимости его инвертирования, производить инвертирование значения гена, равного 1, в 0 с вероятностью p_{10}, а 0 замещать 1 с вероятностью p_{01}. Такой подход позволяет применять ГА для подбора наилучших значений параметров генетических операторов. Например, если в родительской хромосоме некоторые значения вероятностей p_{10} и p_{01} не дали производительного решения, то в хромосоме потомка эти параметры обмениваются значениями, что может привести к улучшению результата.

Кроссовер. В работе предлагается использовать модифицированный оператор кроссовера, который генерирует c, $1 \le c < n$, различных точек кроссовера, делящих хромосомы на c+1 участков. При этом одно потомство наследует четные участки хромосомы первого родителя и нечетные участки второго родителя, второе потомство получается противоположным образом.

Суть предлагаемого адаптивного генетического алгоритма

Основным отличием предлагаемого алгоритма является использование двух стратегий адаптации, в связи с чем процесс получения решения происходит путем циклического повторения двух этапов. На первом этапе с помощью параметрической адаптации выбираются наиболее производительные значения параметров для каждого из используемых типов операторов. На втором этапе оценивается параметр производительности и с учетом результатов алгоритмической адаптации выбирается наиболее производительный тип оператора. Если полученное в результате решение не удовлетворяет заданным критериям, процесс повторяется с первого этапа.

Генетические алгоритмы используют несколько типов операторов, такие как унарные (мутация), бинарные (кроссовер), множественные (многоточечный кроссовер). Каждый тип оператора имеет набор параметров, влияющих на его поведение, а работа ГА начинается с определенными значениями параметров.

Пусть задано множество типов операторов $T = \{t^1, ..., t^j, ..., t^J\}$, где $j = \overline{1, J}$ — количество типов операторов, применяемых в ГА. Например, модифицированный оператор кроссовера или модифицированный оператор мутации. Каждый тип оператора t^j, $j = \overline{1, J}$, имеет набор параметров $P(t^j) = (p_{j,1}, ..., p_{j,k}, ..., p_{j,K})$, $k = \overline{1, K}$. Например, оператор модифицированной мутации имеет два параметра, которым соответствуют вероятности p_{10} и p_{01}, а кроссовер только один параметр – количество точек кроссовера.

Для генерации новых поколений ГА использует генетические операторы с параметрами, соответствующими его типу, причем параметры принимают значения из некоторого множества или интервала. Так, для модифицированного оператора мутации значения его параметров будут выбираться из интервала (0, 1), тогда как параметр оператора кроссовера может принимать любое положительное целочисленное значение, меньшее $(n-1)$.

Для увеличения шансов на выживание и размножение операторов со значениями параметров, показавших лучшие результаты, применяется параметрическая адаптация. Рассмотрим АГА, который использует типы операторов из множества T. Для каждого оператора типа $t^j \in T$, $j = \overline{1, J}$ строится популяция значений его параметров. Например, если для АГА определены два оператора: модифицированный оператор мутации и модифицированный оператор кроссовера, то фиксированным набором экземпляров операторов могут быть – для мутации $(alter\_rate, p_{10}, p_{01}) \in \{(0,1; 0,1; 0,9), (0,7; 0,8; 0), (0,4; 0,1; 0,1)\}$ для кроссовера – $(c) \in \{(1), (2), (n-1)\}$.

Обозначим набор используемых операторов АГА как $O = \{op_1, ..., op_h, ..., op_H\}$, $h = \overline{1, H}$, где H – количество операторов АГА.

Поведение каждого оператора корректируется изменением значений его параметров с использованием для каждого оператора эволюционной процедуры, функционирующей в пространстве возможных значений оператора. Для каждого оператора с момента запуска АГА производятся изменения значений его параметров. Поскольку оптимизируется относительно небольшое пространство поиска, то для поиска набора значений операторов, приводящих к улучшению результатов работы АГА, для каждого отдельного оператора также используется ГА. ГА, осуществляющий поиск лучших значений операторов, обозначим ОГА.

Популяция для каждого оператора будет состоять из хромосом, которые представляют собой набор возможных значений параметров этого оператора. Для генерации новых популяций используется одноточечный кроссовер. ОГА, применяемый для поиска в пространстве значений отдельного типа оператора наилучших значений, запускается как процедура ГА, обозначаемого далее ГГА и работающего над непосредственным поиском решения задачи. Полученное в результате использования ОГА решение служит исходными данными для ГГА. При этом АГА рассматривается как совокупность циклов ОГА и ГГА.

На этапе работы ГГА решается задача алгоритмической адаптации – увеличение вероятностей использования операторов, которые дают лучшие решения. Для оценки работы операторов введем следующие понятия.

Событие – применение конкретного генетического оператора. *Абсолютное улучшение* – событие, когда значение целевой функции нового поколения больше, чем значение целевой функции предшествующих поколений. *Улучшение* – новое поколение имеет значение целевой функции лучшее, чем у родителей. *Стабилизация решения* – значение целевой функции нового поколения незначительно отличается от родительского на протяжении ряда эпох. *Вырождение* – новое поколение имеет худшее значение целевой функции, чем родители, и ни одно из поколений не лучше, чем родительское.

Введем параметр производительности, показывающий эффективность использования оператора в текущем поколении и используемый в качестве обратной связи для эволюционного процесса. Основываясь на значениях показателей производительности, АГА корректирует значения вероятностей использования операторов. Параметр производительности для оператора op_h, $h = \overline{1, H}$ определим как функцию четырех переменных $\pi_{op_h}(ae, e, pw, w)$, где ae – количество абсолютных улучшений, e – количество улучшений, pw – количество стабилизированных решений, w – количество вырождений. Общее количество событий на шаге работы АГА определяется как $N = ae + e + pw + w$.

Параметр производительности введен с целью определения, какой из операторов дает лучшее решение на данном шаге работы АГА. Частота использования операторов учитывается относительными частотами появления событий определенного типа. При $N > 0$ относительные частоты рассчитываются так: для абсолютного улучшения – $\varphi_{ae} = ae / N$; улучшения – $\varphi_e = e / N$; стабилизированного решения – $\varphi_{pw} = pw / N$; вырождения – $\varphi_w = w / N$.

В данной работе принят следующий порядок сравнения параметров производительности, который на примере двух операторов будет формулироваться следующим образом. Более производительным будет оператор, параметр производительности которого характеризируется большим числом абсолютных улучшений. Если количество абсолютных улучшений одинаково, сравнение производится по количеству улучшений. Если количество улучшений также одинаково, сравниваются количества плоских событий. Если последнее не позволяет выделить лучшего оператора, то более производительным будет тот оператор, у которого меньше количество вырождений. При равном количестве вырождений используется мутация.

Для определения порядка использования операторов при работе АГА введено понятие награда. Каждому оператору $op_h \in O$, $h = \overline{1, H}$ приписывается награда $\rho(op_h)$, которая увеличивается в результате накопления положительного опыта. В первую очередь используется оператор, который имеет наибольшую награду. Значение параметра $\rho(op_h)$, $h = \overline{1, H}$ постоянно обновляется, а опыт, приобретенный при последних испытаниях, рассматривается как более актуальный.

Пусть $\chi(op_h)$, $h = \overline{1, H}$ ранг оператора op_h, присваиваемый в зависимости от производительности таким образом, что наивысший ранг получает наиболее производительный тип оператора. Причем присвоение ранга производится после завершения работы ОГА. Награды обновляются в конце каждой эпохи АГА согласно формуле

$$\rho(op_h) = \delta\rho(op_h) + \beta + \gamma\chi(op_h), h = \overline{1, H}$$

где δ – коэффициент ослабления, являющийся константой из множества {0, 1}. Причем $\delta = 0$ для оператора, только вступившего в работу, а $\delta = 1$ означает, что весь предыдущий опыт оператора полностью учтен. Коэффициент усиления γ служит для награждения лучших операторов. Коэффициент β, обычно имеющий значение меньше γ, используется для гарантии того, что оператор будет обязательно использован на этапе АГА.

Ниже приведен пример предлагаемого АГА для двух операторов – кроссовера и мутации, а также ограничений на продолжительность выполнения в виде количества эпох работы.

Шаг 1. Задать условия остановки алгоритма, которыми могут быть количество эпох эволюционирования генетического алгоритма или время работы алгоритма. Шаг 2. Инициализировать начальную популяцию случайным образом с учетом ограничений (1)–(3) и следующего правила: каждый раз при обращении к популяции для решения задачи ее особи отсортировываются в порядке убывания значения целевой функции. Наилучший представитель популяции – хранимое решение задачи. Лучшая из популяций запоминается как хранимая популяция. На начальном этапе хранимой является первичная популяция.

Шаг 3. Случайным образом инициализировать популяцию значений операторов с учетом ограничений, накладываемых для каждого типа операторов. Например, для параметров мутации и кроссовера могут использоваться следующие значения $(alter\_rate, p_{10}, p_{01}) \in \{(0,1; 0,1; 0,1), (0,2; 0,2; 0,2), (0,3; 0,3; 0,3)\}$ и $(c) \in \{(1), (2), (3)\}$ соответственно.

Шаг 4. Использовать каждое из значений соответствующих типов операторов для генерации промежуточных популяций. С помощью целевой функции выбрать значения, позволившие достичь наибольших показателей производительности для каждого из типов операторов. Например, для мутации это может быть $(alter\_rate, p_{10}, p_{01}) = (0,1; 0,1; 0,1)$, для кроссовера – $(c) = (1)$. В случае улучшения полученных результатов изменить хранимую популяцию, используя результаты, полученные посредством самого производительного их двух операторов, и перезаписать хранимое решение задачи. При равенстве операторов по производительности выбрать оператор мутации. Если не удалось улучшить начальное решение, перейти к шагу 3 и использовать ОГА для корректировки значений параметров применением оператора кроссовера. Если не удалось улучшить параметры операторов путем кроссовера, применить мутацию во избежание возможного попадания в локальные экстремумы. Применение параметра мутации при корректировке значений параметров операторов производится по циклу с помощью следующих правил: для кроссовера – увеличить на единицу количество точек кроссовера, а если количество точек равно $(n-1)$, принять следующее количество точек кроссовера равным 1; для мутации – увеличить каждое из значений параметров мутации на 0,1, а если значение любого из параметров равно 0,9, принять следующее значение равным 0,1. Если решение не улучшается, закончить работу и считать полученное решение наиболее производительным.

Шаг 5. Присвоить ранги операторам, рассчитать награды, применить операторы M раз (общее количество этапов работы ГГА будет равно $2 \times M$ на первой эпохе и $3 \times M$ на всех последующих) в порядке уменьшения величины награды. Если применение оператора улучшило решение, то использовать решение в качестве нового хранимого решения задачи, после чего продолжить работу с улучшенной популяцией.

Шаг 6. По истечении заданного числа этапов ГГА обновить значение параметров производительности и выбрать наиболее производительного оператора. Например, если $(alter\_rate, p_{10}, p_{01}) = (0,1; 0,1; 0,1)$ оказался наиболее производительным, то необходимо сохранить его и использовать в следующих эпохах. Перезаписать хранимое решение задачи.

Шаг 7. Если условия остановки АГА не выполнены, то перейти на шаг 3 и начать новую эпоху. На этапе работы ОГА выбрать наиболее производительные значения для всех типов операторов. Например, если

$(alter\_rate, p_{10}, p_{01}) = (0,2; 0,2; 0,2)$ и $(c) = (1)$ для мутации и кроссовера соответственно, то на этапе ГГА работа над популяцией будет проводиться для трех операторов (оператор мутации $(alter\_rate, p_{10}, p_{01}) = (0,1; 0,1; 0,1)$ переходит из предыдущей эпохи). По окончании работы ГГА пересчитать и сравнить параметры производительности, выбрать лучшего из трех операторов и перейти к следующей эпохе (шаг 3).

Шаг 8. Если условия остановки АГА выполняются, то завершить работу, использовать текущее хранимое решение в качестве решения задачи, иначе перейти на шаг 3 и продолжать работу до выполнения условия остановки.

Пример применения предлагаемого АГА

Пример применения предлагаемого алгоритма рассмотрим на задаче распределения ВМ в ЦОД.

Математическую модель задачи распределения ВМ по серверам представим следующим образом.

ЦОД содержит упорядоченное множество серверов $N = \{N_1, \ldots, N_n\}$, где n – количество физических серверов; подлежит распределению упорядоченное множество виртуальных машин $K = \{K_1, \ldots, K_m\}$, где m – количество ВМ; каждый сервер N_i, $i = 1, \ldots, n$, характеризуется двумя параметрами, определяющими его вычислительную мощность: Ω_i – мощность процессора сервера N_i; Γ_i – емкость ОЗУ сервера N_i; каждая ВМ K_j, j=1, …, m, имеет потребности в вычислительных ресурсах: ω_j – в процессорном времени; γ_j – в ОЗУ; матрица R распределения ВМ по серверам, $R = \|r_{ji}\|$ размера $m \times n$, где

$$r_{ji} = \begin{cases} 1, \text{если ВМ } K_j \text{ располагается на сервере } N_i, \\ 0, \text{в противном случае.} \end{cases}$$

Матрица R является решением задачи и определяет распределение множества K ВМ на множестве N физических серверов.

Считаем, что все серверы множества N имеют идентичные технические характеристики и, следовательно, одинаковые вычислительные ресурсы, поэтому полагаем, что $\Omega_i = 1$ и $\Gamma_i = 1$ для всех $i = 1, \ldots, n$, т. е.

$$\{\Omega_i, \Gamma_i\}|_{Ni} = \{1, 1\}, \text{ для всех } i = 1, \ldots, n. \quad (1)$$

Таким способом делаем переход от измерения вычислительных ресурсов серверов в абсолютных единицах, когда память измеряется в ГБ, а частота процессора в ГГц, к относительным.

Тогда потребности ВМ определяются как части от ресурсов сервера, нормированные относительно максимально возможной величины, равной единице.

Считаем, что потребности каждой ВМ в ресурсах не превышают возможностей сервера

$$\omega_j \le 1 \text{ и } \gamma_j \le 1, \text{ для всех } j = 1, \ldots, m \quad (2)$$

При решении задачи распределения ВМ для всех серверов множества N должно выполняться следующие ресурсное ограничение

$$\sum_{j=1}^{m} r_{ji}\omega_j \le 1 \text{ и } \sum_{j=1}^{m} r_{ji}\gamma_j \le 1, \text{ для } i = 1, \ldots, n. \quad (3)$$

Введем вектор $\vec{y} = \langle y_i \rangle$, $i = 1, \ldots, n$, где

$$y_i = \begin{cases} 1, \text{если на сервере } N_i \text{ располагается} \\ \quad \text{хотя бы одна ВМ}, \\ 0, \text{в противном случае.} \end{cases}$$

Тогда критерием оптимальности при решении задачи размещения ВМ на физических серверах будет

$$min \sum_{i=1}^{n} y_i, \quad (4)$$

т. е. сервера должны заполняться так, чтобы было задействовано минимальное их количество.

При выполнении критерия (4) будут минимизированы суммарные затраты S на обслуживание и энергоснабжение парка серверов ЦОД.

Целевую функцию можно представить следующим образом

$$S = \sum_{i=1}^{n} s_i y_i \quad (5)$$

где s_i – затраты на обслуживание и энергоснабжение i-го сервера.

В случае, когда сервера в ЦОД имеют идентичные технические характеристики, выражение (5) примет вид

$$S = s\sum_{i=1}^{n} y_i \quad (6)$$

где s – стоимость поддержания и затраты на энергоснабжение для одного сервера.

С учетом вышеизложенного задачу распределения множества K ВМ можно сформулировать следующим образом: необходимо размещать ВМ на серверах ЦОД таким образом, чтобы выражение (5) или (6) достигало минимального значения.

Результаты экспериментальных исследований

В работе произведены экспериментальные исследования на примере решения задачи распределения ресурсов ЦОД с помощью трех алгоритмов: классического ГА, управляемого генетического алгоритма [1], предлагаемого адаптивного генетического алгоритма.

По аналогии с [1–3] в данной работе исследования проводятся для различных вариантов соотношения ВМ и ресурсов серверов, на которых размещаются ВМ. Для исследований рассматриваются три варианта соотношений ресурсов, наиболее приближенные к реальным ситуациям:

— случай непропорциональных потребностей в ресурсах, когда в относительных единицах запрашиваемая величина процессорного времени (ЦП) значительно превышает запрашиваемый объем ОЗУ, т. е. $\omega_j >> \gamma_j$, для всех $j = 1, \ldots, m$. Подобный тип задачи возникает при наличии большого количества приложений, требующих сложный вычислений;

— также случай непропорциональных потребностей в ресурсах, когда запрашиваемая величина ЦП значительно меньше запрашиваемого объема ОЗУ, т. е. $\omega_j << \gamma_j$, для всех $j = 1, \ldots, m$. Такая задача возникает, когда на серверах ЦОД размещаются ВМ с приложениями, требующими обработки больших объемов данных;

— наиболее распространенный на практике случай, когда количество запрашиваемых ЦП и ОЗУ для всех ВМ распределено случайным образом в диапазоне [0,05; 0,6].

Пределы, в которых изменяются запрашиваемые ресурсы ВМ для различных экспериментов, представлены в таблице 1.

Табл. 1. Характеристики ВМ

№ эксперимента	Ограничения, ОЗУ	Ограничения, ЦП
1	0,3—0,45	0,05—0,15
2	0,45—0,6	0,05—0,15
3	0,05—0,15	0,3—0,45
4	0,05—0,15	0,45—0,6
5	0,05—0,6	0,05—0,6

В [1] доказано, что при количестве ВМ меньше пятидесяти эвристические и генетические алгоритмы дают приблизительно одинаковые результаты, а с увеличением количества ВМ выигрыш ГА становится все более ощутимым.

Отображение результатов работы КГА, УГА и АГА приводится с помощью количества высвобожденных серверов. Предполагается, что изначально для развертывания каждой ВМ выделяется отдельный сервер. Далее с помощью исследуемых алгоритмов производится оптимизация размещения ВМ на серверах с оценкой максимального количества высвобожденных серверов для каждого из алгоритмов. На рис. 2 представлен график зависимости количества высвобожденных серверов от размерности задачи (количества ВМ) для случая, когда количество запрашиваемых ЦП и ОЗУ для всех ВМ распределено случайным образом в диапазоне [0,05; 0,6]. По оси абсцисс откладывается количество ВМ, по оси ординат – количество высвобожденных серверов.

Для сравнения результатов работы УГА и АГА предлагается ввести понятие дополнительно высвобожденных серверов N_B. Значение N_B определяется как разность между количеством серверов $N_{КГА}$, высвобожденных в результате работы классического ГА, и количеством серверов $N_{УГА}$ и $N_{АГА}$, высвобожденных в результате применения УГА и АГА соответственно. Таким образом, на графиках рис. 3 показан выигрыш УГА и АГА относительно ГА в виде зависимости количества дополнительно высвобожденных серверов N_B от размерности задачи для различных вариантов соотношений запрашиваемых ресурсов. По оси абсцисс откладывается количество ВМ, которые необходимо расположить на серверах ЦОД. По оси ординат – количество дополнительно высвобожденных серверов для каждого из алгоритмов.

Данные для экспериментов генерировались случайным образом с равномерным законом распределения. Результаты экспериментальных исследований представлены на рис. 3.

Анализ графиков на рис. 3 позволяет сделать следующие выводы:

— использование УГА и АГА эффективнее, чем использование классического ГА;

— предлагаемый АГА является однозначным лидером независимо от условий эксперимента;

— при наличии ВМ с большими требованиями к ОЗУ либо ЦП (графики рис. 3,*б*, и рис. 3,*г*) эффективность использования УГА и АГА выравнивается, однако преимущество остается за АГА.

— для случая разброса требований в широком диапазоне [0,05; 0,6] (график на рис. 3,*д*), использование АГА наиболее эффективно.

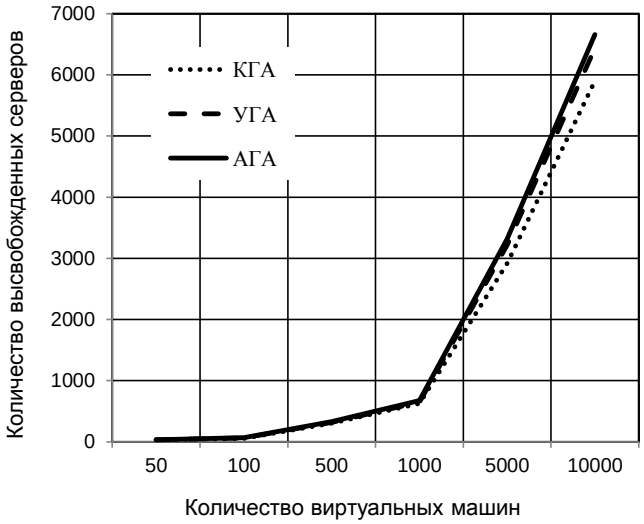
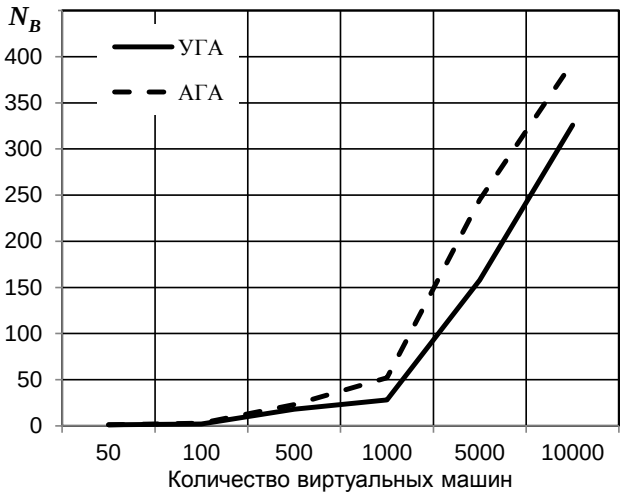
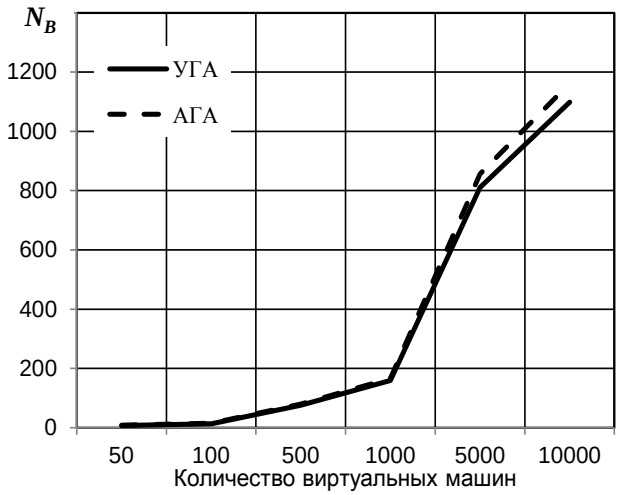


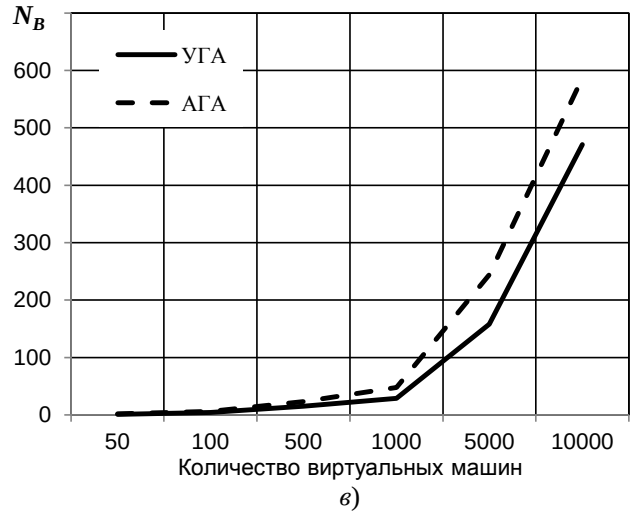
Рис. 2. Зависимость количества высвобожденных серверов от размерности задачи



а)



б)



в)

г)

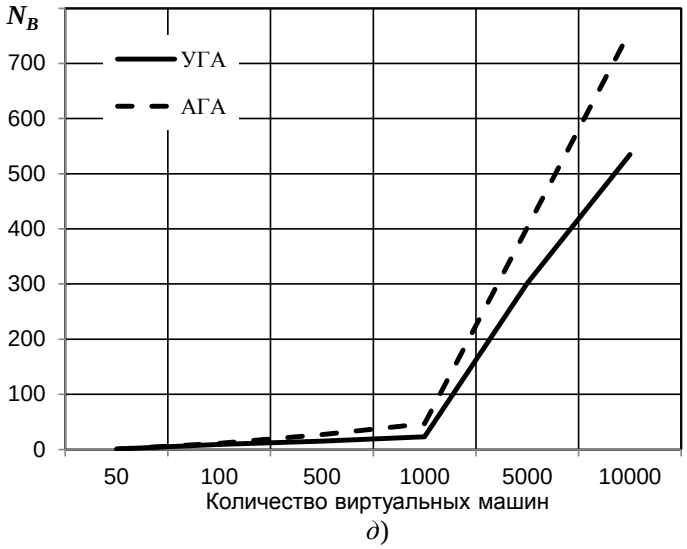


Рис. 3. Зависимость количества дополнительно высвобожденных серверов от размерности задачи, когда требования к запрашиваемым ресурсам лежат в диапазонах: а) к ЦП – [0,05; 0,15], к ОЗУ – [0,3; 0,45]; б) к ЦП – [0,05; 0,15], к ОЗУ – [0,45; 0,6]; в) к ЦП – [0,05; 0,15], к ОЗУ – [0,3; 0.45]; г) к ЦП – [0,05; 0,15], к ОЗУ – [0,45; 0,6]; д) к ЦП – [0,05; 0,6], к ОЗУ – [0,05; 0,6]

Выводы

Разработан адаптивный генетический алгоритм, позволивший решить проблему подбора значений для параметров генетических операторов и вероятностей применения этих операторов в процессе эволюции генетического алгоритма. Фундаментальной особенностью предложенного адаптивного генетического алгоритма является одновременное применение параметрической и алгоритмической адаптации. При проектировании адаптивного генетического алгоритма введены следующие понятия: модифицированные операторы кроссовера и мутации, параметрическая и алгоритмическая адаптация генетического алгоритма, сформированы правила сравнения операторов и понятия параметра производительность генетического оператора. На примере решения задачи распределения ресурсов доказана работоспособность и эффективность предлагаемого адаптивного генетического алгоритма. Доказано, что предлагаемый алгоритм позволяет за равное количество эпох получать лучшие результаты по сравнению с классическим и управляемым вариантом генетического алгоритма.

В следующих публикациях планируется развитие положений адаптивного генетического алгоритма, составление рекомендаций по настройке параметров алгоритма для конкретных задач, проведение экспериментальных исследований для других задач ИТ-сферы.

Литература

1. Теленик С.Ф. Управляемый генетический алгоритм в задачах распределения виртуальных машин в ЦОД / С.Ф. Теленик, А.И. Ролик, П.С. Савченко, М.Е. Боданюк // Вісник ЧДТУ. — 2011. — № 2. — С. 104—113.
2. Теленик С. Управління навантаженням і ресурсами центрів оброблення даних при віртуальному хостингу / С. Теленик, О. Ролік, М. Букасов, С. Андросов, Р. Римар // Вісн. Тернопільського держ. техн. ун-ту. — 2009. — Том 14. — № 4. — С. 198–210.
3. Теленик С.Ф. Генетичні алгоритми вирішення задач управління ресурсами і навантаженням центрів оброблення даних / С.Ф. Теленик, О.І. Ролік, М.М. Букасов, С.А. Андросов // Автоматика. Автоматизація. Електротехнічні комплекси та системи. — 2010. — №1 (25). — С. 106–120.
4. Back T. Optimal Mutation Rates in Genetic Search / T. Back // Fifth International Conference on Genetic Algorithms: University of Illinois at Urbana-Champaign, July 17–21, 1993. – P. 2–8.
5. Julstrom B. A. What Have You Done for me Lately? Adapting Operator Probabilities in a Steady-State Genetic Algorithm / B. A. Julstrom // Proc. of the Sixth International Conference on Genetic Algorithms: University of Pittsburgh, July 15–19, 1995. – P. 81–87.

6. Kosorukoff A. Using incremental evaluation and adaptive choice of operators in a genetic algorithm / A. Kosorukoff // Proc. of the Genetic and Evolutionary Computation Conference: (GECCO-2002), New York, USA, July 9–13, 2002. – P. 688.
7. Lis J. Self-adapting Parallel Genetic Algorithms with the Dynamic Mutation Probability, Crossover Rate and Population Size / J. Lis, M. Lis // Proc. of the 1st Polish National Conference on Evolutionary Computation. Oficina Wydawnica Politechniki, Warszawskiej. – 1996. – P. 324–329.
8. Michalewich Z. Genetic Algorithms + Data Structures = Evolution Programs / Z. Michalewich. – Berlin: Springer, 1996. – 387 p.
9. Michalewich Z. How to Solve It: Modern Heuristics / Z. Michalewich, D. Fogel. – Berlin: Springer, 2002. – 483 p.
10. Pelikan M. Combining the strengths of Bayesian optimization algorithm and adaptive evolution strategies / M. Pelikan, D.E. Goldberg, S. Tsutsui // Genetic and Evolutionary Computation Conference: (GECCO-2002), New York, USA, July 9–13, 2002. – P. 512–519.
11. Spears W.M. Adapting Crossover in Evolutionary Algorithms / W.M. Spears // 4th Annual Conference on Evolutionary Programming: San Diego, California, March 1–3, 1995. – P. 367–384.
12. Thierens D. Adaptive mutation rate control schemes in genetic algorithms / D. Thierens // Congress on Evolutionary Computation: CEC'02, Honolulu, Hawaii, May 12–17, 2002. – P. 980–985.
13. Растригин Л. А. Адаптация сложных систем / Л. А. Растригин // Изв. АН ЛатвССР. – 1978. – №5 (370). – С. 38—45.

УДК 004.27

КЛИМЕНКО І.А.,
ЖАБІНА В.В.,
ЗВОЛИНСЬКИЙ В.В.

МОДЕЛЮВАННЯ ВІДМОВОСТІЙКОЇ ПОТОКОВОЇ ОБЧИСЛЮВАЛЬНОЇ СИСТЕМИ НА ПЛІС

Розглянуто можливу реалізацію та виконано аналіз характеристик відмовостійкої потокової обчислювальної системи, що реалізована за технологією система-на-кристалі. Показано, що застосування методу автоматичної реконфігурації дозволяє виконувати відновлення системи при відмові обчислювальних модулів без суттєвого збільшення часу обчислень.

The possible implementation was considered and the analysis of fault tolerant characteristics was performed for the stream computing system implemented by the technology system-on-chip. It is shown that application of the method of automatic reconfiguration allows to recovery the system by the failure of computational modules without significantly increase of the computation time.

Вступ

Для систем управління в реальному часі важливими характеристиками є швидкодія, надійність і достовірність одержаних результатів. Можливим підходом до прискорення обробки інформації є розпаралелювання обчислювальних процесів на різних рівнях. У системах реального часу виникає необхідність реалізації алгоритмів із дрібнозернистою структурою. Ефективним підходом до прискорення реалізації таких алгоритмів є використання динамічних засобів розподілу операцій між обчислювальними вузлами, заснованих на потоковій моделі обчислень [1]. Відповідно до цієї моделі під час підготовки завдання немає необхідності в прямій формі описувати процедури синхронізації процесів і розподілу пам'яті. Програмування задач не потребує врахування числа обчислювальних модулів у системі та опису послідовності обчислень. Програми при цьому менш громіздкі і більш прості в налагодженні. Усе це створює передумови для прискорення обробки інформації.

Ефективним підходом до проблеми підвищення надійності систем та достовірності результатів обробки інформації є забезпечення відмовостійкості систем шляхом введення різного роду надмірності (апаратної, інформаційної та часової). Для систем реального часу ефективними є програмно-апаратні методи динамічної реконфігурації систем при відмові обладнання [1].

Вдосконалювання елементної бази завжди було одним з ефективних методів підвищення продуктивності обчислювальних систем. Технологія систем-на-кристалі, в першу чергу, сприяє значному підвищенню продуктивності, зменшенню вартості цифрових пристроїв, виконаних на одному кристалі, швидкості розробки та гнучкості проектування.

Таким чином, задача підвищення продуктивності і забезпечення відмовостійкості обчислювальних потокових систем, в яких забезпечується динамічний розподіл завдань між обчислювальними вузлами і динамічна реконфігурація під час відмов обладнання є актуальною. Для досягнення цього необхідна розробка нових методів виконання операцій і організації систем з використанням сучасної елементної бази – програмованих логічних інтегральних схем.

Постановка задачі

У роботі [2] запропоновані методи автоматичної реконфігурації обчислювальних систем, що управляються потоком даних, з різною архітектурою середовищ розподілу команд. Показана можливість реконфігурації системи на апаратному рівні без додаткових витрат команд. В потокових системах підготування команд здійснюється без врахування кількості обчислювальних модулів (ОМ). Це створює передумови у випадку відмови ОМ продовжувати обчислення доки в системі не залишиться хоча б один дієздатний ОМ. Для зберігання команди, що може бути загублена під час відмови ОМ використовується додатковий регістр команд, де протягом всього циклу виконання зберігається копія команди. У випадку відмови ОМ команда повторно виконується у іншому праце-

здатному ОМ. У формат кожної команди запропоновано [2, 3] ввести поле максимального часу очікування виконання даної команди. Визначений час виконання команди завантажується в блок таймеру, який запускається завантаженням слова команди у ОМ. Це дозволяє зменшити час затримки під час реконфігурації системи за рахунок індивідуального налаштування блоків таймерів для кожної операції.

У даній роботі авторами поставлена задача проаналізувати функціональні характеристики запропонованої в [2, 3] обчислювальної системи за реалізації її на ПЛІС, проаналізувати час виконання команд та ефективність засобів автоматичної реконфігурації.

Обґрунтування вибору елементної бази та засобів для створення проекту

Для виконання досліджень використовувалась ПЛІС *EP2C35F672C6N* сімейства *Cyclone* II фірми *Altera*. Мікросхема займає значне місце на ринку мікроелектроніки, доступна широкому колу розробників, маючи невисоку вартість разом з високою логічною ємністю та продуктивністю [4]. В табл. 1 зведені основні характеристики мікросхеми.

Табл. 1. Ресурси ПЛІС Cyclone II EP2C35F672C6N

Логічна ємність та швидкодія	Кількість логічних елементів	33216
	Обсяг вбудованого ОЗП (Кбіт)	484
	Кількість блоків вбудованого ОЗП M4K (4 Кбіт + 512 бітів парності)	105
Особливості	Кількість вбудованих помножувачів 18 x 18 бітів / 9 x 9 бітів	35/70
	Регістри вводу-виводу в елементах вводу-виводу	+
	Блоки двохпортового ОЗП	+
	Кількість глобальних і локальних ланцюгів тактування	16
Підсистема вводу-виводу	Підтримувані рівні напруги вводу-виводу (В)	1.5, 1.8, 2.5, 3.3
Підтримка інтерфейсів зовнішньої пам'яті	Підтримувані інтерфейси зовнішньої пам'яті	QDRII, DDR2, DDR, SDR

Для проектування, розробки та налагоджування розроблюваних пристроїв на ПЛІС, застосовувалась система автоматизації проектування (САПР) *Qwartus* II, яку компанія *Altera* поставляє з усіма своїми продуктами. САПР *Qwartus* II, забезпечує повний цикл проектування та розробки електронних пристроїв на кристалах ПЛІС, включаючи розробку проекту, синтез та моделювання, трасування, завантаження в кристал, тестування та налагоджування, в тому числі внутрісистемне. Проект синтезований з використанням мови опису апаратури *VHDL*.

Визначення часу моделювання основних операцій

Структура та алгоритми роботи реалізованої на ПЛІС системи (рис. 1) відповідають обчислювальній системі, що запропонована в роботі [3].

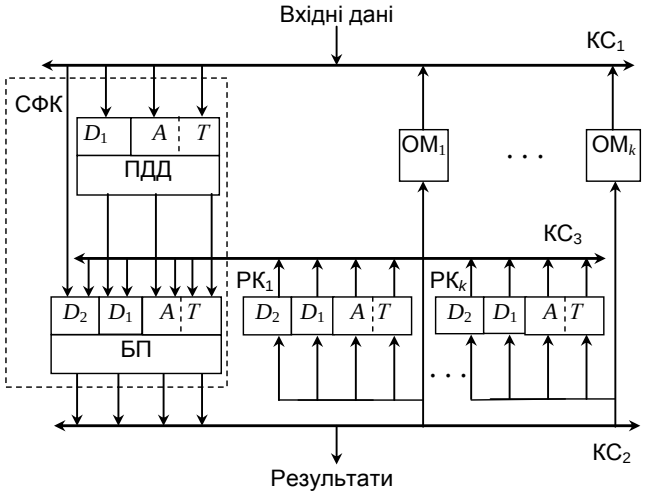


Рис. 1. Організація системи: СФК – схема формування команд, ППД – пам'ять з довільним доступом, БП – буферна пам'ять, КС – комутаційне середовище, РК – регістр команд, D_1 і D_2 – поля даних, A – поле актора, T – поле таймера

Під час проведення тестового моделювання експериментально визначено період часу, протягом якого всі управляючі сигнали встигають надійти до входів відповідних блоків обчислювальної системи в очікуванні наступного перепаду синхросигнала. Цей час прийнятий в якості тривалості синхросигналу і дорівнює 20 нс. Також експериментально визначені тривалості значень часу моделювання основних операції, які представлені в табл. 2.

Значення максимального часу виконання операцій необхідні для контролю виконання команди в ОМ. Одержані значення заносяться у таймер для кожного ОМ перед початком вико-

нання команди. Якщо операція не виконана за визначений час (спрацював таймер), то вважається, що ОМ відмовив.

Табл. 2. Час виконання основних операцій в системі на ПЛІС

Операція	Час виконання, нс
Додавання	6
Віднімання	6
Множення	7
Ділення	12
Зведення в квадрат	7
Вилучення кореня	8
Порівняння $a = b$	6
Порівняння $a > b$	6
Порівняння $a < b$	6
Порівняння $a \geq b$	6
Порівняння $a \leq b$	6
Вентиль *a if b*	6
Вентиль *a if not b*	6
2-розмножувач	3
N-розмножувач	$N \times 2 + 2$
Повторювач *a*	6
Повторювач *b*	6

Оцінка задіяних ресурсів ПЛІС

Принциповими обмеженнями побудови мультипроцесорних систем на ПЛІС є кількість логічних ресурсів, зокрема загальна кількість логічних вентилів, об'єм убудованої пам'яті, внутрішні канали передачі даних та виводи мікросхеми [5]. В цьому зв'язку в роботі оцінена максимальна обчислювальна потужність розроблюваної системи, яку дозволить реалізувати використання одного кристалу ПЛІС заданого класу.

За результатами моделювання роботи обчислювальної системи отримані наступні значення задіяних ресурсів мікросхеми для різної кількості ОМ (табл. 3). Врахувавши фактичні ресурси мікросхеми *Cyclone* II *EP*2*C*35*F*672*C*6*N* (табл. 1), визначено, що максимальна кількість ОМ дорівнює одинадцяти. В табл. 3 показані дані про сумарні задіяні ресурси мікросхеми під час моделювання обчислювальної системи з різною кількістю обчислювальних модулів, тобто враховані також середовище формування і розподілу команд та пам'ять.

Табл. 3. Оцінка задіяних ресурсів ПЛІС

Кількість АОМ	Логічні елементи	Біти пам'яті	Вбудовані помножувачі 9 x 9 бітів
1	2871	176896	6
2	5468	176896	12
3	8015	176896	18
11	29224	176896	66

Дослідження характеристик системи без відмов устаткування

В якості досліджуваних критеріїв оберемо коефіцієнт прискорення обчислень та коефіцієнт ефективності використання різної кількості ОМ, вважаючи, що їх максимальна кількість дорівнює одинадцяти. Обчислення виконуються за наступними формулами:

$$K_{\text{П}} = \frac{T_1}{T_P},\ K_{\text{Е}} = \frac{K_{\text{П}}}{P},$$

де: T_1 – час розв'язування задачі з одним ОМ; T_P – час розв'язування задачі в з P ОМ; $K_{\text{П}}$ – коефіцієнт прискорення; $K_{\text{Е}}$ – коефіцієнт ефективності; P – кількість ОМ.

Задача 1 полягає в обчисленні функції, що має наступний вигляд:

$$F = \frac{\left(y^2 + b \cdot y + 2 \cdot a \cdot b + 4\right) \cdot \left(\left(a^2 + b^2 + c^2 + d^2\right) \cdot a \cdot b\right)}{a - b - c \cdot a + 12 \cdot \sqrt{b^2 + a^2}}.$$

Граф задачі на модифікованій графічній мові Деніса [1] представлений на рис. 2.

З результатів моделювання (табл. 4) видно, що для задачі, яка розглядається, немає необхідності використовувати більш трьох ОМ. Збільшення кількості ОМ не дозволяє зменшити час обчислень, але впливає на надійність відмовостійкої обчислювальної системи, що є основною метою даного дослідження. В системі обчислення можуть продовжуватися поки існує хоча б один дієздатний ОМ. Таку можливість забезпечують засоби автоматичної реконфігурації системи під час відмови ОМ.

Табл. 4. Результати моделювання системи на ПЛІС

Кількість АОМ	Час розв'язування задачі, мкс	Коефіцієнт прискорення	Коефіцієнт ефективності
1	9.45	1	1
2	7.91	1.1947	0.5973
3	7.83	1.2069	0.4023
4	7.83	1.2069	0.3017
5	7.83	1.2069	0.2414
6	7.83	1.2069	0.2011
7	7.83	1.2069	0.1724
8	7.83	1.2069	0.1509
9	7.83	1.2069	0.1341
10	7.83	1.2069	0.1207
11	7.83	1.2069	0.1097

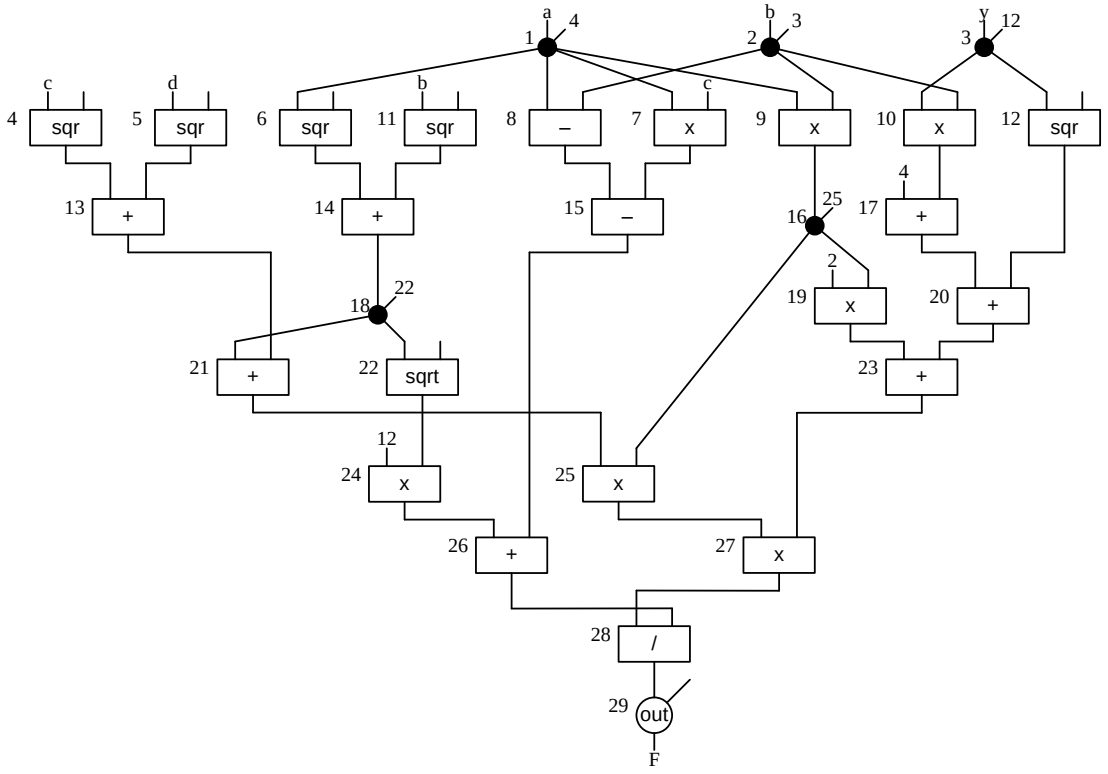


Рис. 2. Граф задачі 1

Моделювання та дослідження характеристик системи під час відмови обчислювальних модулів

В процесі обчислень готовий до виконання команди ОМ формує сигнал готовності. Чергова команда передається для виконання в один з готових для цього ОМ. Під час функціонування обчислювальної системи можуть виникнути збої в роботі одного або декількох ОМ. Збої можуть виникнути до початку виконання команди (не завершено процес пересилання команди в ОМ) або після того, як ОМ почав виконувати команду [4].

Під час досліджень виконано моделювання обох випадків для ряду задач різної складності, в тому числі, для задачі 1 (рис.2) та задачі 2 (рис.3), яка містить цикли.

В якості досліджуваного параметру розглядається час виконання програми з урахуванням часу реконфігурації системи під час відмови визначеної кількості ОМ.

Розглянемо випадок1, коли збої в роботі ОМ виникають в процесі пересилання команди в ОМ. Результати зведені у табл. 5 і табл. 6.

З результатів моделювання видно, що середня затримка часу розв'язування задачі під час відмови ОМ в даному випадку досить мала, наприклад, 0 мкс за відмови до чотирьох ОМ.

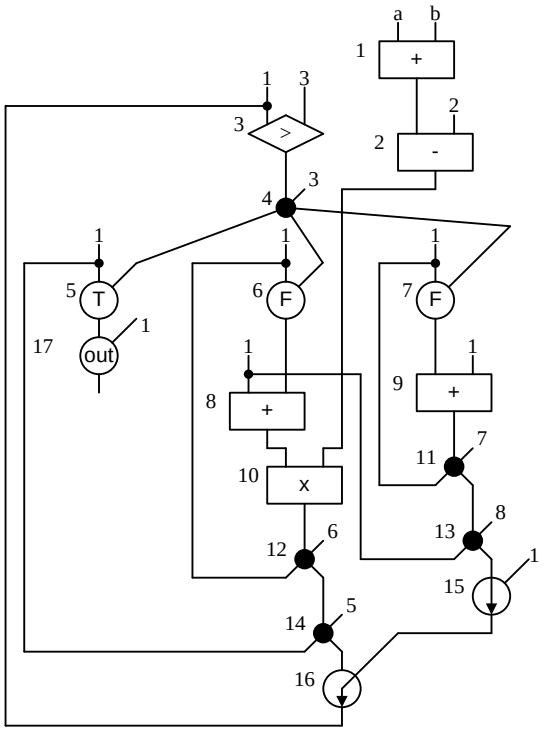


Рис. 3. Граф задачі 2

Такі значення пояснюються тим, що час реконфігурації системи менше, ніж час формування наступної команди, тому затримка непомітна. За відмови п’яти і більше ОМ затримка збільшується, бо збільшується час реконфігурації, особливо коли в системі залишається один працюючий ОМ.

Розглянемо випадок 2, коли збої виникають після того, як ОМ почав виконувати команду. Результати зведені у табл. 7 і табл. 8.

Середня затримка часу розв'язування задачі під час відмови ОМ зростає з кількістю ОМ, що відмовили. Це пояснюється тим, що має пройти певний час, на протязі якого буде визначена відмова ОМ, а цей час залежить від максимального часу виконання операції, який визначається таймером для кожної команди. Крім цього, час обчислень залежить від стану керуючих автоматів в момент відмови ОМ, тому затримка обчислень в табл. 8 зростає нерівномірно.

Табл.5. Час розв'язування задачі 1 (випадок 1), мкс

Кількість ОМ	Кількість ОМ, що відмовили (починаючи з 1-го)									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
11	7.83	7.83	7.83	7.83	7.83	7.83	7.83	7.83	7.85	10.21

Табл. 6. Час розв'язування задачі 2 (випадок 1), мкс

Кількість ОМ	Кількість ОМ, що відмовили (починаючи з 1-го)									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
11	12.15	12.15	12.15	12.15	12.19	12.33	12.33	12.33	12.45	14.25

Табл. 7. Час розв'язування задачі 1 (випадок 2), мкс

Кількість ОМ	Кількість ОМ, що відмовили (починаючи з 1-го)									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
11	7.95	7.91	7.95	8.07	8.03	8.05	8.11	8.15	8.27	10.37

Табл. 8. Час розв'язування задачі 2 (випадок 2), мкс

Кількість ОМ	Кількість ОМ, що відмовили (починаючи з 1-го)									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
11	12.27	12.23	12.49	12.49	12.59	12.57	12.67	12.59	13.01	14.83

Висновки

В роботі виконано моделювання потокової відмовостійкої обчислювальної ситеми на ПЛІС *EP*2*C*35*F*672*C*6*N* сімейства *Cyclone* II фірми *Altera*, засобами САПР *Quartus* II.

Показано, що витрати часу на реконфігурацію системи під час відмови ОМ є незначними щодо загального часу виконання заданих обчислень. Це пояснюється тим, що відновлення системи за відмови ОМ реалізується на апаратному рівні без витрати додаткових команд. Реконфігурація системи в основному зводиться до відключення ОМ, що відмовив, тобто не потребує великих витрат часу. За наявності однотипних ОМ такий підхід дає можливість продовжувати обчислення доки в системі не залишиться хоча б один дієздатний модуль. Це підвищує вірогідність результатів обчислень і забезпечує умови для підвищення надійності системи в цілому.

Під час побудови систем може бути використаний базовий принцип побудови відмовостійких систем – принцип високонадійного ядра, яке контролює іншу апаратуру [6]. В якості захищеного ядра може виступати середовище формування команд. Даний компонент системи реалізований на основі запам’ятовуючих пристроїв, для яких добре розроблені апаратні методи підвищення надійності [7].

Таким чином, для реалізації алгоритмів з дрібнозернистою структурою можуть бути ефективно використані потокові обчислювані системи на ПЛІС, які забезпечують автоматичне розподілення операцій між обчислюваними модулями і автоматичну реконфігурацію системи в разі відмови обладнання.

Перелік літератури

1. Жабин В.И. Архитектура вычислительных систем реального времени / В.И.Жабин. – К.: ВЕК +, 2003. – 176 с.
2. Клименко И.А. Обеспечение отказоустойчивости потоковых систем на однотипных вычислительных модулях / И.А.Клименко, В.В.Жабина // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Видавництво «ВЄК +», 2009. – №51. – С. 166 – 171.
3. Пат. № 59112 Україна, МПК G06F 15/16 (2006.01). Обчислювальний пристрій / І.А. Клименко, В.В. Жабина; Заявник і патентовласник: Національний авіаційний університет, Київ. – Заявлений 06.08.2010. – Опубл. 10.05.2011. – Бюл. №9.
4. Cyclone II Device Handbook [Електронний ресурс]. – Altera Corporation, 2008. – Режим доступу: http://www.altera.com/devices/fpga/cyclone2/cy2-index.jsp.
5. Клименко І.А. Тенденції застосування сучасної елементної бази для побудови високопродуктивних обчислювальних систем / І.А. Клименко // Проблеми інформатизації та управління: Зб.наук.пр.– К.: Вид-во нац. авіац. ун-ту «НАУ-друк», 2010.– Вип.1(29). – С 90 – 103.
6. Авиженис А. Отказоустойчивость – свойство, обеспечивающее постоянную работоспособность цифровых систем / А. Авиженис // ТИИЭР (пер. с англ.). – 1978. – Т. 66, №10. – С. 5 – 25.
7. Огнев И.В. Надежность запоминающих устройств / И.В. Огнев, К.Ф. Сарычев. – М.: Радио и связь, 1988. – 223 с.

УДК 004.056.2

МАРКОВСЬКИЙ О.П.,
ФЕДОРЕЧКО О.І.,
САЇДРЕЗА МАХМАЛІ

ТЕХНОЛОГІЯ ЦИФРОВОГО ПІДПИСУ DSA НА ОСНОВІ АРИФМЕТИКИ ПОЛІВ ГАЛУА

Запропоновано модифікацію алгоритму формування цифрового підпису DSA, що базується на новому використанні арифметики кінцевих полів. Наведено математичне обґрунтування запропонованого підходу. Описано технології генерації ключів, формування цифрового підпису та його перевірки. Для всіх цих процедур наведено числові приклади. Доведено, що використання арифметики кінцевих полів дозволяє помітно прискорити роботу з цифровим підписом. Запропонована модифікація алгоритму формування цифрового підпису DSA орієнтована на апаратну реалізацію.

The modification of DSA techniques based on novel application of arithmetic of finite fields are presented. The mathematical background of the proposed approach is first presented. The techniques of public and secret keys generation, forming and verification of signature based of finite fields arithmetic are described. A numerical example for all mentioned procedures is given. It has been showed that using of finite fields arithmetic may greatly accelerate the processing of DSA signatures. The proposed DSA modification is oriented for hardware implementation.

Вступ

Сучасний етап розвитку інформаційних технологій все більше підтверджує відому тезу академіка В.М.Глушкова про настання ери безпаперової звітності. Фактично, вже сьогодні зростаючий рівень динаміки ділових відносин диктує такий рівень оперативності документообігу, що може бути досягнутим лише використанням сучасних комп'ютерних та мережових технологій. Особливо рельєфно ця тенденція виявляється в банківській справі. За умов переходу до безпаперового документообігу великої ваги набуває забезпечення автентичності документів: тобто має гарантуватися авторство документу, а також те, що при передачі по відкритим мережам його не змінено.

Для забезпечення цілісності та автентичності документів з середини 80-х років активно використовуються криптографічні механізми цифрового підпису. В середині 90-х в більшості країн було прийнято криптографічні стандарти цифрового підпису. Всі вони базуються на математичних операціях модулярного експоненціювання над числами великої розрядності. Відповідно, обчислювальна реалізація існуючих механізмів цифрового підпису потребує помітних витрат часу. Зростання обчислювальних потужностей, які потенційно можуть бути використані зловмисниками для підробки цифрового підпису диктує необхідність постійного збільшення розрядності чисел, що застосовуються при формуванні цифрового підпису. Так, в сучасних умовах для забезпечення потрібного рівня захищеності, розрядність чисел становить 2048 з перспективою збільшення в найближчі роки до 4096. При реалізації операцій модулярного експоненціювання збільшення розрядності чисел має наслідком експоненційне зростання часу обчислень. Причому, темпи зростання об'єму обчислень випереджають збільшення швидкодії комп'ютерних систем. Особливо гостро проблема часу формування цифрового підпису стоїть для термінальних портативних малопотужних обчислювальних пристроїв, які підтримують мережові протоколи захисту інформації.

Таким чином, проблема зменшення обчислювальної складності процесів формування цифрового підпису і, відповідно, прискорення контролю автентичності документів є на сьогоднішній день актуальною.

Аналіз існуючих алгоритмів цифрового підпису

Цифровий підпис інформаційного повідомлення – це криптографічний механізм, який гарантує по-перше, цілісність – тобто те, що повідомлення не зазнало змін, а по-друге: авторство, тобто те, що повідомлення підписано певним автором [1].

Початок практичному використанню цифрового підпису банківськими установами було

покладено створенням механізмів несиметричної криптографії. Перші механізми цифрового підпису мали за основу відомий алгоритм несиметричного шифрування RSA. Згодом стало зрозумілим, що використання алгоритмів несиметричного шифрування неефективне, оскільки пов'язане зі надлишковими витратами обчислювальних ресурсів. Розпочалися роботи зі створення ефективних спеціалізованих механізмів цифрового підпису. Етапним стало прийняття в серпні 1991 року стандарту цифрового підпису DSS (Digital Signature Standard). Цей стандарт передбачає процедуру перевірки авторства та цілісності повідомлення згідно з алгоритмом DSA (Digital Signature Algorithm) [2]. Згодом, в Росії було прийнято в якості стандарту практично ідентичний за математичними принципами алгоритм формування цифрового підпису ГОСТ Р.34.10-94 [1].

Сутність алгоритму DSA полягає в слідуючому. При генерації ключів алгоритму виконується наступна послідовність дій.

1) Генерується просте l-розрядне число p. Наприклад, при l=6 p= 43 = 101011_2.

2) Знаходиться число q фіксованої розрядності $b \approx l/2$, що є подільником числа p-1. Наприклад, якщо p= 43, то p-1= 42 = 6·7 і тоді q=7, b=3.

3) Довільним чином вибирається число h, таке, що $h<p$-1 і $h^{(p-1)/q} \bmod p > 1$. Наприклад, якщо вибрати h = 5, то $5^6 \bmod 43 = 16$.

4) Обчислюється $g = h^{(p-1)/q} \bmod p$. Наприклад при p=43, q=7 та h=5 g=16.

5) Вибирається число $x<q$, наприклад, x = 5.

6) Обчислюється $y = g^x \bmod p$. В рамках поточного прикладу $y=16^5 \bmod 43 = 21$.

Згенеровані описаним способом числа p, q g та y – являються компонентами відкритого ключа, а x – закритий ключ.

Процедура формування цифрового підпису лицем, що знає закритий ключ x полягає в наступному.

При передачі документу m формується його хеш-сигнатура $H(m)$ фіксованої розрядності. Стандартом для формування передбачено використання хеш-алгоритму SHA-1 [3]. Нехай, для прикладу, $H(m)$=37. Лице, що підписує повідомлення виконує таку послідовність дій:

1) Довільним чином вибирається k таке, що $k<q$; наприклад k = 3.

2) Обчислює $r = (g^k \bmod p) \bmod q$. В рамках прикладу, що розглядається, значення r обчислюється як: $r = (16^3 \bmod 43) \bmod 7 = 11 \bmod 7 = 4$.

3) Визначається мультиплікативна інверсія k^{-1} так, що $k \cdot k^{-1} \bmod q = 1$. Для приклада, при k = 3 значення k^{-1}= 5, оскільки 3·5 mod 7 =15 mod 7 =1.

4) Обчислюється $s = (k^{-1} \cdot (H(m) + x \cdot r) \bmod q$. В рамках прикладу, що розглядається значення s =(5·(37+5·4))mod 7 = 285 mod 7 = 5.

5) Цифровий підпис, який складається з двох компонентів r та s відсилається приймачеві.

Одержувач документу проводить перевірку цифрового підпису наступним чином.

1) Для отриманого документу m' за встановленим хеш-алгоритмом обчислюється хеш-сигнатура $H(m')$. Якщо документ не змінено, то $m=m'$ і відповідно $H(m)=H(m')$. Наприклад, якщо $H(m)$=37 і документ не змінено при передачі, то $H(m')$ =37.

2) Знаходиться мультиплікативна інверсія s^{-1} така, що $s \cdot s^{-1} \bmod q = 1$. Якщо q=7 і s=5 тоs^{-1}=3.

3) Обчислюється $u_1 = (H(m') \cdot s^{-1}) \bmod q$. Для прикладу, що розглядається $u_1 = (37 \cdot 3) \bmod 7 = 111 \bmod 7 = 6$.

4) Обчислюється $u_2 = (r \cdot s^{-1}) \bmod q$, зокрема для прикладу $u_2 = (4 \cdot 3) \bmod 7 = 12 \bmod 7 = 5$.

5) Обчислює $v = ((g^{u_1} \cdot y^{u_2}) \bmod p) \bmod q$. Для прикладу, що розглядається $v = ((16^6 \cdot 21^5) \bmod 43) \bmod 7 = 11 \bmod 7 = 4$

6) Якщо $v = r$, то вважається, що документ не зазнав змін при передачі і від підписаний лицем, яке знає закритий ключ x.

Математичний сенс процедури DSA полягає в тому, що знаючи x, автор формує $s = (k^{-1} \cdot (H(m) + x \cdot r) \bmod q$. Зрозуміло, що $(s \cdot k) \bmod q = (k \cdot k^{-1} \cdot (H(m)+x \cdot r)) \bmod q = (H(m)+x \cdot r) \bmod q$. Якщо $H(m)=H(m')$, то $H(m')+x \cdot r = s \cdot k$. З урахуванням наведеного, за умови того, що прийнятий документ є автентичним справедливо:

$$v = ((g^{u_1} \cdot y^{u_2}) \bmod p) \bmod q =$$

$$= ((g^{H(m') \cdot s^{-1}} \cdot (g^x)^{r \cdot s^{-1}}) \bmod p) \bmod q =$$

$$= (g^{s^{-1} \cdot (H(m')+x \cdot r)} \bmod p) \bmod q =$$

$$= (g^{s^{-1} \cdot k \cdot s} \bmod p) \bmod q = (g^k \bmod p) \bmod q = r$$

Якщо $H(m) \neq H(m')$, то $H(m')+x \cdot r \neq s \cdot k$, і, відповідно, $v \neq r$. Якщо підпис документу зробила інша особа з використанням ключа $x' \neq x$, то $y \neq g^{x'}$ і $v \neq r$. Якщо припустити, що підпис перехоплений зловмисником, то його ціллю є змінити текст документу на $m'' \neq m$ та підібрати таке $x'' \neq x$ щоб $(k^{-1} \cdot (H(m'') + x'' \cdot r) \bmod q = s$. Це зробити важно, так як значення k не передається і, від-

повідно, зловмиснику не відоме. Фактично, потрібно підбирати два b-розрядні числа x'' так, що якщо вважати, що $b \approx l/2$, то об'єм перебору становить 2^l.

Очевидним недоліком DSA є значна обчислювальна складність. В процесі формування цифрового підпису виконується одна операція модулярного експоненціювання b-розрядних чисел, віднаходження мультиплікативної b-розрядної інверсії k^{-1}

Для зменшення обчислювальної складності віднаходження мільтиплікативної інверсії в алгоритмі DSA застосовано зменшення вдвічі розрядності при виконанні цієї операції, оскільки розрядність q приблизно в два рази менша розрядності p.

Для зменшення обчислювальної складності операцій молярного експоненціювання найчастіше використовується метод Монтгомері, який дозволяє звести складну в обчислювальному плані операцію модулярної редукції до зсувів. Застосування для зменшення об'єму обчислень результатів передобчислень неефективне в силу того, що код експоненти змінюється при кожному формуванні цифрового підпису.

Ціллю досліджень є підвищення продуктивності формування та перевірки цифрового підпису.

Алгоритм цифрового підпису на основі експоненціювання на полях Галуа

Операція модулярного експоненціювання над числами, довжина яких значно перевищує розрядність процесора, виконується фрагментами. При виконанні арифметичної операції над кожним із фрагментів потрібно враховувати можливість переносу в наступний фрагмент. Виходячи з цього, одним з напрямків підвищення продуктивності реалізації модулярного експоненціювання є заміна арифметичних операцій на логічні, при виконання яких не потрібно враховувати перенос. Фактично мова йде при заміну операції арифметичного модулярного експоненціювання на аналогічну операцію на полях Галуа.

В плані захищеності використання операцій арифметичного модулярного експоненціювання та експоненціювання на полях Галуа практично є ідентичним. Дійсно, при використанні модулярного експоненціювання $A^h \bmod M = \xi$ порушення захисту в математичному сенсі є ідентичним віднаходженню мінімального h з рівняння $A^h = j \cdot M + \xi$ при відомих значеннях A, M та ξ і не відомому значенні j. Практично це рівняння може бути розв'язане лише шляхом перебору, об'єм якого при великій розрядності перевищує технічні можливості комп'ютерних систем. Якщо через символ $\otimes$ позначити операцію множення без переносів, через $A|^p$ – експоненту степені p на основі множення без переносів, тобто $A|^h = \overset{h}{\underset{j=1}{\otimes}} A$, а через $D \operatorname{rem} M$ – остачу від поліноміального ділення D на M, то операція експоненціювання на полі Галуа з утворюючим поліномом M може бути позначена як $A|^h \operatorname{rem} M$. При застосуванні експоненціювання на полях Галуа для захисту даних, його порушення в математичному сенсі означає віднаходження мінімального h, що задовольняє рівнянню $A|^h = j \otimes M \oplus \xi$ в якому заданими є значення A, M та ξ. Цілком очевидним є те, що єдиним шляхом розв'язання цього рівняння є також перебір.

Аналіз математичних принципів, покладених в основу DSA показує, що в ньому використовується два простих числа p і q, для яких існує g таке, що:

$$g^q \bmod p = 1. \qquad (1)$$

Відповідно, за умови (1) для будь-яких цілих x та n виконується $(g^{x+n \cdot q} \bmod p) \bmod q = (g^x \bmod p) \bmod q$.

Аналогічно, в модифікації DSA на основі експоненціювання на полях Галуа потрібно також застосування 3-х компонент: l-розрядного числа p, b-розрядного числа q та числа g для яких (1) трансформується в:

$$g|^q \operatorname{rem} p = 1. \qquad (2)$$

Число p є простим в алгебрі множення без переносів, тобто його не можна представити у вигляді добутку без переносів двох чисел. Прикладом 9-розрядного (l=9) простого в алгебрі множення без переносів є число $p=299_{10} = 100101011_2$. Число q має бути простим в арифметичному сенсі подільником $2^{l-1}-1$. Наприклад, якщо l=9, то $2^{l-1}-1 = 255 = 3 \cdot 5 \cdot 17$, відповідно, можливим варіантом може бути q=17. Умові (2) задовольняє g=29.

Процес генерації ключів модифікації DSA на основі експоненціювання на полях Галуа полягає в наступному:

1) Генерується просте в алгебрі множення без переносів l-розрядне число p. На практиці число l має бути не меншим 1025. Наприклад, вибирається 9-розрядне число p=299.

2) Знаходиться число q – один із простих, в арифметичному сенсі, подільників числа 2^{l-1}-1 та число g, що задовольняє (2). Якщо p=299, то можливими значеннями q та g є q=17 і g=29.

3) Вибирається випадкове x, наприклад x=10.

4) Обчислюється x=10 = $g|^x$ rem p. Якщо p=299, g=29, а x=10, то $y = 29|^{10}$ rem 299 = 114.

Згенеровані описаним способом числа p, q g та y – являють собою відкритий ключ, тоді як x – закритий ключ.

Процедура формування цифрового підпису лицем, що знає закритий ключ x полягає в наступному.

При передачі документу m формується його хеш-сигнатура $H(m)$ фіксованої розрядності. Нехай, для прикладу, $H(m)$=134. Лице, що підписує повідомлення виконує таку послідовність дій:

1) Довільним чином вибирається k таке, що $k<q$; наприклад k = 14.

2) Обчислюється $r = (g|^k$ rem $p)$ mod q. Якщо p=299, g =29, q=17, а k=14, то $y = (29|^{14}$ rem 299)mod 17 = 88 mod 17 = 3.

3) Визначається мультиплікативна інверсія k^{-1} така, що $k \cdot k^{-1}$ mod q = 1. При q=17 і k =14 значення k^{-1} = 11, оскільки 11·14 mod 17 = 154 mod 17 =1.

4) Обчислюється $s = (k^{-1} \cdot (H(m) + x \cdot r) \bmod q$. В рамках прикладу, що розглядається значення s =(11·(134 + 10·3)) mod 17 = 1804 mod 17 = 2.

5) Цифровий підпис, який складається з двох компонентів r та s відсилається одержувачу документу. Для наведеного вище прикладу одержувачу відсилається пара r =3 і s =2.

Одержувач документу отримує текст документу m' та дві компоненти цифрового підпису (r та s). Перевірка цифрового підпису проводиться виконанням наступної послідовності дій:

1) За допомогою встановленого хеш-алгоритму обчислюється хеш-сигнатура прийнятого документу $H(m')$.

2) Обчислюється мультиплікативна інверсія s^{-1} компоненти s цифрового підпису, тобто знаходиться таке s^{-1}, що добуток $s \cdot s^{-1}$ mod q = 1. Для s=2 s^{-1} = 9 оскільки 18 mod 17 = 1.

3) Обчислюється $u_1 = (H(m') \cdot s^{-1}) \bmod q$. Для прикладу, що розглядається u_1 = (134·9) mod 17 = 1206 rem 17 = 16.

4) Обчислюється $u_2 = (r \cdot s^{-1}) \bmod q$. Для поточного прикладу u_2 = (3·9) mod 17=27 mod 17 =10.

5) Обчислюється $v = ((g|^{u_1} \otimes y|^{u_2})$ rem $p)$mod q. Для прикладу, що розглядається, чисельне значення v=((29|^{16}⊗114|^{10}) rem 299) mod 17 = ((51⊗130)rem299)mod 17=(6630rem 299)mod17= = 88 mod 17 = 3.

6) Якщо $v = r$, то вважається, що документ не зазнав змін при передачі, а також він підписаний лицем, яке знає закритий ключ x. Для прикладу, що ілюструє виклад, v=3=r, тобто цілісність та автентичність документу вважається доведеною.

Оцінка ефективності

Головною перевагою запропонованого способу формування та перевірки цифрового підпису на основі експоненціювання на полях Галуа є менша обчислювальна та часова складність у порівнянні з модулярним експоненціюванням, що використовується в стандарті DSA.

Базовою операцією формування та перевірки цифрового підпису є модулярне експоненціювання, тобто обчислення $a|^E$ rem p, де a та p - суть l-розрядні числа, а E являє собою b-розрядне число. За класичним алгоритмом експоненціювання, ця операція включає в себе b циклів, по числу розрядів експоненти, в кожному з яких виконується, в середньому, 1.5 операцій множення l-розрядних чисел.

В свою чергу, кожна операція множення складається з власне операції множення та редукції. При модулярному множенні редукція полягає в віднаходженні залишку від арифметичного ділення коду добутку на модуль, а при множенні на полі Галуа редукція полягає в обчисленні залишку поліноміального ділення добутку на утворюючий поліном поля Галуа.

Якщо вважати що всі розряди множника дорівнюють одиниці, то формування молодшого розряду добутку не потребує арифметичних операцій, для обчислення другого розряду потрібно виконати одну операцію додавання. Для формування третього розряду потрібно реалізувати дві операції додавання та, зі ймовірністю 0.5, ще одну операцію додавання переносу. При цьому середня кількість переносів в наступний розряд, четвертий розряд, становить 1.25. Узагальнюючи, можна говорити, що формування i-го розряду добутку ($i \le l$) потребує i-1 бітових операцій додавання та, в середньому, $\xi(i) = \sum_{j=2\ldots i} 0.5^{j-1} \cdot (i - j)$ додавань, для врахування переносів. Аналогічно, формування m-го ро-

зряду добутку ($l<m<2\cdot l$) потребує $2\cdot l-m$ бітових операцій додавання та, в середньому, $\xi(2\cdot l-m)$ додавань, для врахування переносів. В цілому, приймаючи до уваги, що лише половина розрядів множника, в середньому, дорівнює одиниці, сумарна кількість D_a бітових при виконанні арифметичного множення становить:

$$D_a = \sum_{i=2}^{l} ((i-1) + \xi(i)) \approx l^2. \qquad (3)$$

При виконанні множення без переносів середня кількість D_g бітових операцій додавання становить:

$$D_g = \sum_{i=2}^{l} (i-1) \approx \frac{l^2}{2}. \qquad (4)$$

Порівняння формул (2) та (3) показує, що обчислювальна складність множення без переносів практично вдвічі менша арифметичного множення. Аналогічний аналіз проведений щодо складності редукції показує, що арифметична редукція має вдвічі більше обчислювальну складність у порівнянні з віднаходженням залишку поліноміального ділення.

Таким чином, перехід від традиційної арифметики до арифметики на кінцевих полях при виконанні базових для технологій цифрового підпису операцій модулярного експоненціювання дозволяє вдвічі зменшити обчислювальну складність.

Суттєво більший виграш досягається при порівнянні часової складності. Як було показано вище, кількість операцій додавання l-розрядних слів при арифметичному множенні та множенні на полях практично однакова. Проте, кожен із розрядів при додаванні на полях оброблюється незалежно, в той час, як при виконанні арифметичного додавання потрібно формувати перенос.

При використанні найбільш швидкодіючих схем прискореного переносу довжина критичного шляху його формування становить $1.5\cdot\log_2 l$. Це означає часова складність формування цифрового підпису з використанням арифметики полів Галуа не менш як в $1.5\cdot\log_2 l$ разів менша ніж при застосуванні традиційної арифметики. Враховуючи, що на практиці $l\geq 1024$, виграш у часовій складності становить не менше 15 разів.

В роботі [4] запропоновано ефективний спосіб обчислення експоненти на полях Галуа непрямим шляхом, який з використанням таблиць передобчислень дозволяє на порядки прискорити виконання цієї операції.

З викладеного зрозуміло, що повною мірою переваги запропонованої модифікації DSA з використанням арифметики кінцевих полів можуть бути реалізовані в рамках апаратного виконання.

Висновки

Для підвищення продуктивності формування та перевірки цифрового підпису DSA запропонована його модифікація. Модифікація полягає в використанні при реалізації найбільш ресурсоємкої операції роботи з цифровим підписом - модулярного експоненціювання арифметики кінцевих полів, зокрема полів Галуа. Розроблено відповідні технології генерування ключів, формування та перевірки цифрового підпису.

Доведено, що використання арифметики кінцевих полів замість традиційної дозволяє помітно прискорити роботу з цифровим підписом. Запропонована модифікація алгоритму формування цифрового підпису DSA орієнтована на апаратну реалізацію.

Список літератури

1. ГОСТ Р.34.10-94. Системы обработки информации. Защита криптографическая. Алгоритм формирования цифровой подписи.
2. Digital Signature Standard (DSS).#186. US Department of commerce, National Institute of Standards and Technology, 1994.
3. Secure Hash Standard. Federal Information Processing Standard Publication #180, US Department of Commerce, National Institute of standard and technology, 1995.
4. Самофалов К.Г. Марковський О.П., Шаршаков А.С. Способ ускоренной реализации экспоненцирования на полях Галуа в системах защиты информации // Проблеми інформатизації та управління. Збірник наукових праць: Випуск 2(33).-К.,НАУ.- 2011.- С.143-151.

УДК 004.75

САЛЬНІКОВ А.О.,
БОЙКО Ю.В.

ВИРІШЕННЯ ПРОБЛЕМ ІНТЕГРАЦІЇ ВІРТУАЛЬНИХ ОРГАНІЗАЦІЙ НА ПРОВАЙДЕРАХ РЕСУРСІВ В УКРАЇНСЬКОМУ НАЦІОНАЛЬНОМУ ГРІД-СЕГМЕНТІ

В роботі проведено аналіз існуючих проблем інтеграції віртуальних організацій (ВО) в українському національному грід-сегменті (УНГ) та показано ключові обмеження провайдерів ресурсів. Запропоновано методики вирішення проблем масштабованості, гнучкості та цілісності політик, відокремлення ВО. Виконано успішне впровадження методик на трьох кластерах УНГ.

Contemporary virtual organizations (VO) integration problems in Ukrainian National Grid (UNG) has been analyzed. Key limitations of UNG resource providers are shown. Techniques for solving the problems of scalability, policy flexibility and integrity as well as VO separation are proposed. Successful deployments of proposed methods are conducted on three UNG member clusters.

1. Вступ

Грід-обчислення це різновид розподілених обчислень, що орієнтовані на використання ресурсів світового масштабу, зазвичай із застосуванням високопродуктивних обчислювальних кластерів [1]. За останні роки грід набуває все більшого розповсюдження та розвитку як в світі в цілому (Європейська грід-інфраструктура – EGI [2]), так і в Україні зокрема (Український національний грід – УНГ [3]).

Побудова грід-інфраструктури, EGI чи УНГ, є лише фундаментом для вирішення ресурсоємних задач. Головною метою координованого доступу до ресурсів є вирішення реальних наукових завдань динамічними групами людей з різних установ, в так званих *віртуальних організаціях* (ВО). Дослідження кожної ВО напрямлені на вирішення певної наукової задачі.

Обмін обчислювальними потужностями чи простором зберігання даних, з боку провайдерів ресурсів має чітко визначати до яких сервісів надається доступ, кому і за яких умов. Множина користувачів та/або установ що визначається такими правилами і є віртуальною організацією [1].

Головна спільна риса, що характерна для всіх учасників віртуальної організації – вирішення одного класу проблем, а відповідно і використання одного класу обладнання та/або прикладного програмного забезпечення. Спільний клас задач диктує однакові вимоги до сервісів грід-інфраструктури що будуть використані, процесорного часу для виконання розрахунків, обсягів та критеріїв зберігання даних.

Такі вимоги можуть принципово відрізнятися для різних ВО, проте одна й та сама ВО чітко висуває вимоги до грід інфраструктури. Саме тому одиницею з якою працює провайдер ресурсів, як елемент загальної грід-інфраструктури, є віртуальна організація [4]. Цей підхід надає можливість гнучкого налаштування локального планувальника ресурсів обчислювального елементу для гарантії процесорного часу, чи конфігурації квот на елементі зберігання згідно домовленостей з тією чи іншою ВО. Відповідно, договір з провайдерами ресурсів відбувається на рівні ВО, а не окремих користувачів.

2. Проблеми інтеграції ВО на провайдерах ресурів в УНГ

Інфраструктура УНГ історично побудована децентралізованим чином без чіткої координації. Розвиваючись, нові обчислювальні кластери під'єднувались до УНГ встановлюючи базову поставку програмного забезпечення проміжного рівня Nordugrid ARC [5] доступну на час інсталяції.

Конфігурація грід на обчислювальних кластерах при таких інсталяціях обмежувалась лише базовими сервісами рівня ресурсів: обчислювальним елементом (CE) та локальною інформаційною системою грід-ресурсу (GRIS) [6]. Конфігурація обчислювального елементу також залишалася базовою, з використанням так званої *класичної авторизації*. Ключові властивості класичної авторизації це [7]:

- використання локальних облікових записів та файлів відповідності;
- механізми авторизації замикає на собі операційна система;

- грід-сервіси не проводять додаткових перевірок для розмежування прав доступу.

На більшості провайдерів УНГ застосовані тривіальні налаштування класичної авторизації – використання єдиного облікового запису для будь-якого учасника будь-якої ВО. Списки учасників ВО формуються атоматично завдяки використанню сервісу VOMS. Переважну більшість українських ВО обслуговує VOMS сервер КНУ [8], проте використання сервісу обмежено лише оновленням списків користувачів.

Використання такого підходу має ключові обмеження для інтеграції ВО:

- Масштабованість: кожна зміна політики доступу для учасника ВО потребує змін на кожному провайдері ресурсів;
- Гнучкість політик: вбудовані засоби контролю політик операційної системи можуть бути недостатньо гнучкими для підтримки внутрішньої структури ВО грід-сервісами;
- Цілісність: вбудовані засоби контролю політик операційної системи можуть відрізнятися на різних провайдерах ресурсів;
- Відокремлення ВО: прив'язка до ВО відбувається виключно за DN користувача, і не може коректно працювати в випадку участі одного користувача в декількох ВО.
- Безпека: завдання різних користувачів різних ВО мають змогу отримати доступ до файлів та процесів всіх грід-завдань через роботу від імені одного локального облікового запису.

В результаті таких обмежень конфігурації провайдерів ресурсів в УНГ, не відбувається розділення роботи різних ВО, не кажучи вже про підтримку внутрішньої структури ВО. Відповідно залишаються неврахованими особливості вимог прикладного програмного забезпечення кожної ВО, якій надаються ресурси. Це призводить до *відсутності ефективного планування завдань, що веде до простою національних обчислювальних ресурсів.*

Більш того серед близько 30-ти учасників УНГ лише одиниці слідкують за оновленнями програмного забезпечення проміжного рівня грід та проводять роботи з розгортання чи забезпечення підтримки нових сервісів. Таким чином сьогодні стоїть *проблема неінтероперабельної роботи* інфраструктури: помилки в роботі та несумісність різних версій програмного забезпечення, практично повна відсутність підтримки роботи з сервісам рівня кооперації грід.

Через збільшення активності використання інфраструктури такими ВО як moldyngrid, networkdynamics та medgrid, насьогодні проблема інтероперабольності гостро стоїть перед УНГ. Роботи з вирішення проблем координованої роботи інфраструктури закладено в державну цільову науково-технічну програму впровадження і застосування грід-технологій на 2009–2013 роки. В рамках програми створено координаційний комітету, який проводить роботи по встановленню регламенту роботи як провайдерів ресурсів так і віртуальних організацій в Україні.

Проте навіть за інтероперабельної роботи, для вирішення проблем масштабованості, гнучкості, цілісності та відокремлення роботи ВО на провайдерах ресурсів необхідні методики, що дозволять систематизувати підхід до інтеграції ВО. Розробка та впровадження таких методик є завданням даної роботи.

3. Засвідчення участі в ВО

Класична авторизація не враховує фактор участі користувача в ВО. Ідентифікація користувача вказує тільки на його унікальне ім'я (DN) за яким неможливо однозначно визначити приналежність до ВО [7]. В той час ефективне розділення ресурсів (резервацій чи приорітезації обчислювальних потужностей, квот елементів зберігання) потребує механізмів визначення такої інформації.

Методом, що дозволяє провести ідентифікацію приналежності до ВО є засвідчення за допомогою сервера VOMS [9]. На сьогодні роботу з VOMS підтримують всі проекти по розробці програмного забезпечення проміжного рівня грід. Служба VOMS є невід'ємною складовою EMI та IGE.

Методика засвідчення участі в ВО будується на розширенні проксі-серти-фікатів делегації користувача. Учасник ВО, за допомогою утиліти для створення делегації, звертається до серверу VOMS з запитом щодо засвідчення параметрів участі в ВО. VOMS сервер створює сертифікат атрибутів (Attribute Certificate – AC) [10], який містить перелік прав доступу учасника відповідно до інформації в базі даних. Такий AC засвідчується цифровим підписом серверу VOMS та передається клієнту користувача, що

додає АС як розширення до базової делегації [9]. Грід-сервіси, що підтримують таке розширення зможуть працювати з параметрами доступу ВО, в той час як інші реалізації будуть ігнорувати таке розширення і працювати спираючись на класичну авторизацію.

Параметри участі в ВО відображено в повному імені атрибутів користувача (Fully Qualified Attribute Name, FQAN). В АС міститься перелік FQAN всіх атрибутів, засвідчення яких вимагалось в запиті користувача. FQAN має наступний формат:

```
/VO[/group[/subgroup(s)]][/Role=role][
/Capability=cap]
```

4. Методики інтеграції ВО на провайдерах ресурсів

Метод засвідчення участі в ВО за допомогою сертифікату атрибутів дозволяє грід-сервісам на провайдері ресурсів отримати інформацію як про участь ВО, так і особливості участі відповідно до внутрішної структури.

Права доступу учасника відображені виключно в FQAN, та можуть змінюватись адміністратором ВО через інтерфейс керування VOMS [8]. Сервіси що не потребують запуску окремих процесів від імені користувача (наприклад елемент зберігання чи каталог файлів) для забезпечення виконання політики доступу використовують FQAN безпосередньо. Якщо необхідно виконати запуск процесів (наприклад обчислювальний елемент чи інтерактивний вхід) необхідне надання відповідності користувача ВО локальному обліковому запису ОС від імені якого буде запущено цільовий процес.

Програмне забезпечення проміжного рівня Nodrugrid ARC, що використовується в УНГ, не містить штатних засобів надання локальних облікових записів в залежності від FQAN учасника в ВО [5].

З поставкою програмного забезпечення gLite проекту EGI використовується інфраструктура Site Access Control (SAC) [11].

Використання бібліотек SAC для Nodrugrid ARC дозволить визначити єдині політики доступу у випадку одночасної роботи сервісів різного програмного забезпечення проміжного рівня (в тому числі авторизації на елементах зберігання, тощо).

Розглянемо методики інтеграції ВО на провайдерах ресурсів з використанням інфраструктури SAC в застосуванні до Nodrugrid ARC.

4.1. Відокремлення ВО. Виходячи з проведеного вище аналізу випливає, що для розділення доступу різних ВО, відповідні їм облікові записи мають бути різними. Більш того, для підвищення ефективності роботи провайдера ресурсів є необхідність розділяти роботу різних учасників однієї ВО для:

- відокремлення роботи (виконання задачі одного учасника не може випадково чи навмисно завадити виконанню завдання іншого);
- надання різних прав доступу в залежності від FQAN учасника;
- забезпечення “справедливого” (fair share) розділення ресурсів (сервіси операційної системи не пов’язані з роботою грід повинні розрізняти завдання різних учасників, щоб уникнути монопольного використання ресурсів).

Відокремлення ВО є особливо важливим для роботи віртуальних організацій, які орієнтовані на обробку захищених приватних даних. За статичного підходу до відповідності користувачів, що використовується в Nodrugrid ARC, при реалізації відокремлення виникає проблема масштабування: учасники додаються та видаляються із віртуальних організацій незалежно від провайдерів ресурсів, що ускладнює процес керування відповідністю до системних облікових записів.

Для вирішення проблеми масштабування необхідно використовувати динамічно розподілені ресурси облікових записів – *пули користувачів* (від англ. pool – сукупність об’єктів). Локальний обліковий запис ставиться у відповідність до ідентифікації користувача динамічно, за допомогою механізму оренди [12]. Для ідентифікації використовується DN учасника разом з VOMS FQAN. Таким чином при засвідченні участі в ВО, користувачеві ставиться у відповідність обліковий запис із пулу, що відповідає даній ВО. Для однієї ВО можуть існувати декілька пулів, а відповідність обліковим записам відбувається в залежності від групи, ролі чи атрибутів учасника. Розміри пулів визначаються кількістю та функціями ресурсів провайдера та локальними політиками використання ресурсів.

Забезпечуючи розподіл за обліковими записами на рівні операційної системи, пули не обмежують архітектуру грід-сервісів, які в свою чергу можуть додатково визначати права доступу безпосередньо використовуючи VOMS AC делегації.

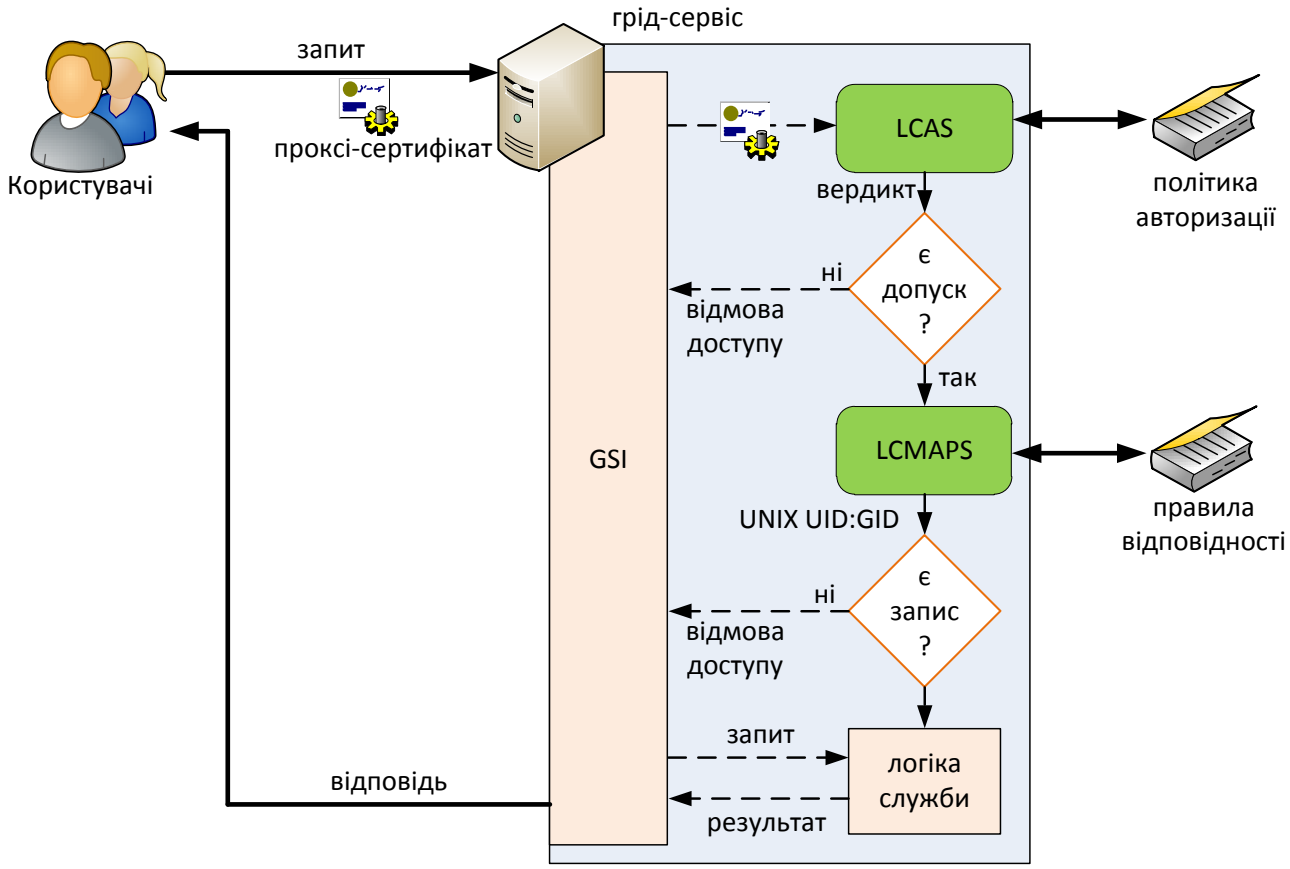


Рис. 1.Схема алгоритму взаємодії GSI з бібліотеками LCAS/LCMAPS

4.2. Цілісність та гнучкість політик за допомогою SAC. Site Access Control – це інфраструктура бібліотек та сервісів локальної авторизації, що надає механізми реалізації політик доступу до грід-ресурсу та забезпечує відповідність грід-користувачів локальним обліковим записам [**Ошибка! Источник ссылки не найден.**].

Бібліотеки LCAS та LCMAPS інфраструктури SAC представлені набором незалежних підключених програм.

Local Centre Authorization Service (LCAS) – бібліотека для реалізації механізмів контролю доступу на сервісах грід. Вона являє собою набір методик прийняття рішення про успішну авторизацію чи заборону доступу.

Local Credential MAPping Service (LCMAPS) – бібліотека для реалізації механізмів присвоєння локальних повноважень (таких як UID облікового запису) для грід-завдань, що виконуються на локальних фабрикатах.

На рисунку 1 зображено схему алгоритму взаємодії GSI з бібліотеками LCAS/LCMAPS. Використання інструментарію LCAS для забезпечення політики авторизації разом з LCMAPS для створення правил відповідності забезпечує локальну авторизацію грід-сервісів та дозволяє ефективно працювати з сертифікатами атрибутів VOMS завдяки відповідним підключеним програмам SAC.

Всі підключені програми LCMAPS розділяють на два типи: *модулі присвоєння (acquisition)* та *модулі забезпечення виконання (enforcement)*. Модулі присвоєння призначені для збору інформації щодо надання повноважень для кожного запиту, які присвоюються модулями забезпечення виконання [12].

Для Nodrugrid ARC було створено зовнішні застосування що викликають LCAS та LCMAPS. Відповідно до алгоритму роботи (рис. 1) LCAS приймає бінарне рішення щодо надання чи відхилення доступу. Зовнішньому застосуванню передається DN користувача та шлях до файлу з проксі-сертифікатом. LCAS перевіряє цифровий підпис серверу VOMS та використовує перевірку доступу відповідно до конфігурації. Код повернення використовується для прийняття рішення.

Приклад конфігурації LCAS для реалізації контролю доступу за VOMS AC наведено на рис. 2.

```
pluginname=lcas_voms.mod,\
pluginargs="-vomsdir /etc/grid-security/vomsdir/ \
     -certdir /etc/grid-security/certificates/ \
     -authfile /etc/grid-security/voms-user-mapfile
```

Рис.2. Лістинг конфігураційного файлу LCAS для роботи з VOMS AC

```
"/moldyngrid/modelling/Role=production" .mdgmodprod
"/moldyngrid/Role=VO-Admin" .mdgadm
"/testbed.univ.kiev.ua" .tb
```

Рис.3. Фрагмент лістингу файлу voms-user-mapfile

Директорія `vomsdir` містить публічні сертифікати серверів VOMS які обслуговують ВО, що підтримуються; `certificates` містить сертифікати довірених кореневих центрів сертифікації грід. Файл `voms-user-mapfile` містить список FQAN, які допущені до використання ресурсу (рис. 3).

В наведеній конфігурації будуть допущені всі засвідчені учасники ВО testbed.univ.kiev.ua та учасники групи modelling з роллю production чи роллю VO-Admin ВО moldyngrid. Другий параметр не використовується LCAS, проте необхідний для використання цього ж файлу бібліотекою LCMAPS.

Винесення рішення щодо авторизації на ресурсі в LCAS дозволяє змінювати політику доступу без переконфігурації самого сервісу.

Nodrugrid ARC виконує присвоєння локальних повноважень (операції setuid та setgid) за допомогою вбудованих засобів, проте для присвоєння використовуються лише механізми класичної авторизації [5]. Виклик LCMAPS з окремої підключеної програми не може виконати зміну повноважень в батьківському процесі. Тому для роботи з Nodrugrid ARC необхідно забезпечити передачу визначених повноважень від LCMAPS.

Аналогічно до виклику LCAS, виконуваній програмі передається DN користувача та шлях до файлу з проксі-сертифікатом. Програма виконує виклик LCMAPS, конфігурація якого не повинна використовувати модулі забезпечення виконання. Отримані значення локальних повноважень передаються до Nodrugrid ARC який і виконує необхідне присвоєння.

Приклад конфігурації LCMAPS для Nordugrid ARC наведено на рис. 4.

Конфігураційний файл визначає наступні операції:

- good – викликає модуль забезпечення виконання, що нічого не робить;
- vomsextract – викликати базовий модуль роботи з VOMS, що обробляє VOMS AC розширення делегації користувача з використанням VOMS API та зберігає ідентифікацію VOMS в структурах даних LCMAPS для роботи інших підключених програм;
- vomspoolaccount – викликати модуль, який повертає значення UID локального облікового запису з заданого пулу користувачів відповідно до участі в ВО;
- vomslocalgroup – аналогічно до vomspoolaccount, але повернути значення GID;

Відповідно до визначених операцій формується політика присвоєння локальних повноважень:

- виклик vomsextract: якщо знайдено валідне AC розширення викликається модуль vomspoolaccount (визначення UID), інакше робота завершується без забезпечення виконання (good);
- Якщо визначення UID виконано успішно (vomspoolaccount) визначається GID, в протилежному випадку робота LCMAPS завершується (good).
- Після визначення GID (vomslocalgroup) робота LCMAPS завершується (good).

Файли та директорії вказані в конфігурації-містять інформацію щодо прив'язки FQAN до пулу користувачів. Ім'я локального аккаунту з пулу містить номер (наприклад, для ВО moldyngrid імена аккаунтів представлені множиною: `mdg01, mdg02,…, mdg50`). Ім'я складається з базового імені до якого додається порядковий номер. В конфігурації для посилання на пул використовується нотація крапки – зазначається лише базове ім'я, якому передує символ «крапка» (наприклад для посилання на пул для ВО moldyngrid використовується `.mdg`).

```
good = "lcmaps_dummy_good.mod"

vomsextract = "lcmaps_voms.mod"
" -vomsdir /etc/grid-security/vomsdir"
" -certdir /etc/grid-security/certificates"

vomspoolaccount = "lcmaps_voms_poolaccount.mod"
" -override_inconsistency"
" -max_mappings_per_credential 1"
" -do_not_use_secondary_gids"
" -gridmapfile /etc/grid-security/voms-user-mapfile"
" -gridmapdir /etc/grid-security/gridmapdir"

vomslocalgroup = "lcmaps_voms_localgroup.mod"
" -groupmapfile /etc/grid-security/voms-group-mapfile"
" -mapmin 1"

voms:
vomsextract -> vomspoolaccount | good
vomspoolaccount -> vomslocalgroup | good
vomslocalgroup -> good
```

Рис.4. Приклад файлу конфігурації LCMAPS для роботи з Nordugrid ARC

5. Впровадження методик

Описані вище методики були застосовані до обчислювальних кластерів Київського національного університету імені Тараса Шевченка (КНУ), Інституту молекулярної біології і генетики (ІМБіГ) НАН України та Національного наукового центру з медико-біотехнічних проблем (ННЦМБП) при президії НАН України.

Апаратна та програмна архітектура обчислювальних кластерів істотно відрізняється, проте використання описаних методик інтеграції ВО є універсальним механізмом, що дозволяє проводити подальше регулювання ефективного планування відповідно до особливостей програмної архітектури та домовленостей про резервації з кожною ВО. Разом з тим вирішуються зазначені проблеми класичної авторизації.

На кожному провайдері ресурсів, налаштовано роботу Nordugrid ARC з зовнішніми програмами виклику LCAS та LCMAPS. Впроваджено конфігурацію бібліотек яка використовує інформацію з VOMS AC.

На обчислювальному кластері КНУ також функціонують сервіси програмного забезпечення gLite, що використовуються для роботи в інфраструктурі EGI. Ці сервіси також використовують LCAS та LCMAPS, що забезпечує інтероперабельну роботу для ВО незалежно від точки входу.

В створеній конфігурації, для обслуговування нової ВО провайдером необхідно:

- створити новий пул користувачів (або декілька пулів за необхідності підтримки складної внутрішньої структури ВО);
- додати до списків допущених FQAN нові записи відповідно до інформації наданої ВО;
- додати інформацію про сертифікат серверу VOMS, що обслуговує ВО.

Разом з методиками авторизації, запропонований підхід забезпечив індикацію класу виконуваних задач за ідентифікатором групи (GID) пулу користувачів. Це дозволило налаштувати резервації та використовувати прогнозування часу обрахунку локального планувальника, зменшуючи час простою обчислювальних ресурсів зазначених кластерів.

6. Висновки

Було проаналізовано існуючі проблеми інтеграції ВО на провайдерах ресурсів УНГ під керуванням Nordugrid ARC, та сформовано ключові обмеження існуючого підходу: відсутність масштабованості, гнучкості та цілісності політик доступу, разом з проблемами відокремлення роботи різних ВО.

Ефективне планування локальних ресурсів можливе лише за наявності детермінованих вимог до процесорного часу, обсягів та критеріїв зберігання даних, які диктує прикладне програмне забезпечення, що використовується. Тому необхідним є розділення доступу для різного класу задач, а відповідно і для різних віртуальних організацій.

Проблему масштабованості конфігурації вирішено шляхом використання серверу VOMS для динамічного керування FQAN учасників ВО.

Проблему відокремлення роботи різних ВО вирішено за допомогою використання пулів локальних облікових записів, прив'язка до яких відбувається у відповідності до FQAN учасника, який засвідчено сервером VOMS за допомогою використання сертифікату атрибутів.

Цілісність та гнучкість політик досягнуто винесенням рішень авторизації та відповідності локальним обліковим засобам в інфраструктуру SAC, а саме використання бібліотек LCAS та LCMAPS. Взаємодія з сервісами Nordugrid ARC відбувається за рахунок конфігурації без забезпечення виконання.

Методики дозволяють додавати нові ВО чи проводити модифікацію конфігурації та політики доступу без переривання роботи та зміни конфігурації самих грід-сервісів.

Методики були застосовані до обчислювальних кластерів Київського національного університету імені Тараса Шевченка, Інституту молекулярної біології і генетики (ІМБіГ) НАН України та Національного наукового центру з медико-біотехнічних проблем (ННЦМБП) при президії НАН України та дозволили гнучко налаштувати локальне планування ресурсів та підтримку віртуальних організацій УНГ.

Перелік посилань

1. Foster, Ian. The Anatomy of the Grid - Enabling Scalable Virtual Organizations / Ian Foster, Carl Kesselman, Steven Tuecke // International Journal of Supercomputer Applications. – 2001. –Vol. 15. –P. 2001.
2. Infrastructure, European Grid. Towards to sustainable grid infrastructure [online] –http://www.egi.eu/. –2011.
3. Український академічний Грід: досвід створення й перші результати експлуатації / Ю.В.Бойко, М.Г.Зинов'єв, О.О.Судаков, С.Я.Свістунов // Математичні машині і системи. –2008. –Vol. 1. –Pp. 67-84.
4. Foster, Ian. The physiology of the grid: An open grid services architecture for distributed systems integration / Ian Foster. – 2002.
5. Ellert, M. Advanced Resource Connector middleware for lightweight computational Grids / M. Ellert et al. // Future Gener. Comput. Syst. –2007. –Vol. 23, no. 1. –Pp. 219–240.
6. Ukrainian Grid Infrastructure: Practical Experience / Mykhaylo Zynovyev, Sergiy Svistunov, Oleksandr Sudakov, Yuriy Boyko // Proceedings of the 4-th IEEE Workshop IDAACS 2007. – 2007. – September. – Pp. 165-169. – 6-8 September 2007, Dortmund, Germany.
7. The Globus Security Team. Globus Toolkit Version 4 Grid Security Infrastructure: A Standards Perspective [online]. – http://www-unix.globus.org/toolkit/docs/5.0/5.0.0/security/GT4-GSI-Overview.pdf. – 2005. – September. – Version 4.
8. PHP VOMS-Admin project development [online]. – http://grid.org.ua/development/pva/. – 2011.
9. Alfieri R. An Authorization System for Virtual Organizations / R. Alfieri, R. Cecchini, V. Ciaschini et al. // In Proceedings of the 1st European Across Grids Conference, Santiago de Compostela. –2003. –Pp. 13–14.
10. Farrell, S. An Internet Attribute Certificate Profile for Authorization. – RFC 5755 (Proposed Standard). [online] – 2010. – January. http://www.ietf.org/rfc/rfc5755.txt.
11. Site Access Control [online]. – http://www.nikhef.nl/pub/projects/grid/gridwiki/ – 2011.
12. Local Credential MAPping Service [online]. – http://www.nikhef.nl/grid/lcaslcmaps/lcmaps-apidoc/html/ – 2005.

УДК 004.021

ГАЕВСКАЯ Е.А.,
СТИРЕНКО С.Г.

МЕТОДЫ ОБРАБОТКИ ИЗОБРАЖЕНИЙ, ПОЛУЧЕННЫХ С ПОМОЩЬЮ ТЕХНОЛОГИИ ФАЗОВОГО КОНТРАСТА

В данной статье рассмотрены методы для обработки изображений такие, как метод Фурье-преобразований, методы обратной проекции, а также итерационные методы. Также проведен анализ и сравнение данных методов с точки зрения эффективности, контрастности полученного изображения, а также скорости обработки.

This article describes methods for image processing, such as method of Fourier transform, back projection method and iterative methods. These methods were analyzed and compared in terms of efficiency, the contrast of the result image and processing speed.

Введение

Метод фазового контраста позволяет получить более контрастные изображения, нежели в традиционной рентгенографии, что позволяет повысить качество медицинского обследования. Чувствительность данного метода основывается на получении и обработке информации о фазовых сдвигах волн при прохождении через объект [1, 2, 3]. В связи с тем, что реализацию данной технологии труднее выполнять экспериментально, фазовый контраст еще не получил широкого распространения, но над этой технологии работают ведущие мировые научные физические центры Франции, Германии, США и Росии, имеющие электронные ускорители – синхротроны, которые являются источниками мощного рентгеновского излучения; а разработками вычислительных систем для аппаратов фазового контраста заняты научные организации Австралии, Франции, США и другие.

Аппараты, использующие технологию фазового контраста, позволяют получить информацию о сдвигах в фазах волн при прохождении через исследуемый объект. Сдвиг фазы определяет проекцию объекта при определенном положении источника рентгеновского излучения. Далее по полученной информации нужно восстановить срезы объекта. Методы, существующие для этого приведены ниже.

Методы для восстановления трехмерной функции объекта, которые получили наибольшее распространение в различных медицинских и других приложениях, можно разделить на два основных класса: аналитические и итерационные (рис. 1).

Аналитические методы основаны на точных математических решениях уравнений восстановления изображения. В основе большинства из них используются аппарат преобразования Фурье и преобразования Радона.

Все аналитические методы реконструкции изображения теоретически эквивалентны, однако отличаются процедурой реализации. К данному классу методов относятся двухмерное восстановление Фурье, метод обратного проецирования, а также метод обратной проекции с фильтрацией (рис. 2).

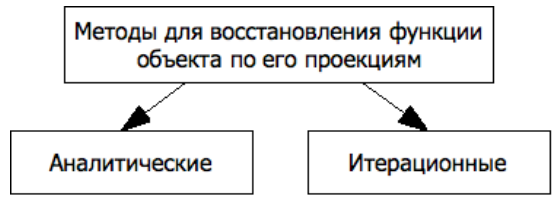


Рис. 1. Классификация методов для восстановления функции объекта по его проекциям

Рис. 2. Виды аналитических методов

Итерационные методы восстановления изображения используют аппроксимацию восстанавливаемого объекта массивом ячеек равной плотности, представляющих собой неизвестные величины, связанные системой линейных алгебраических уравнений, свободными членами которых являются отсчеты на проекции. Решаются системы уравнений итерационными методами, что и дало название данному классу ме-

тодов восстановления. В настоящее время известно несколько итерационных методов восстановления изображения (рис. 3). Отличаются они в основном последовательностью внесения поправок во время итерации.

Целью данной статьи является анализ методов восстановления.

Постановка задачи

Целью данной статьи является выделение наиболее эффективных методов обработки изображений по таким характеристикам, как контрастность изображения, а также скорость получения результата. Для этого проведен анализ существующих методов восстановления изображения по его проекциям, а также моделирование работы алгоритмов, основанных на данных методах, с последующим анализом.

1. Задача восстановления изображения

Для получения трехмерной функции $f(x, y, z)$ исследуемого объекта используют одномерные проекции ее двумерных сечений $f(x, y)$ при фиксированном z. Двумерное сечение исследуемого объекта описывается функцией плотности $f(x, y)$. За пределами сечения плотность предполагается равной нулю [4].

Аппарат фазового контраста генерирует плоскую волну в плоскости исследуемого сечения объекта, которая может быть представлена множеством плоских лучей. После прохождения рентгеновских лучей через сечение, измененные лучи воспринимаются детекторами [3]. Система трубка-детекторы вращается вокруг исследуемого сечения. С системой трубка – детектор связана подвижная система координат (u, v), центр которой совпадает с центром системы (x, y), а оси повернуты относительно неподвижной системы на угол θ. Направление лучей совпадает с координатой u (рис. 3).

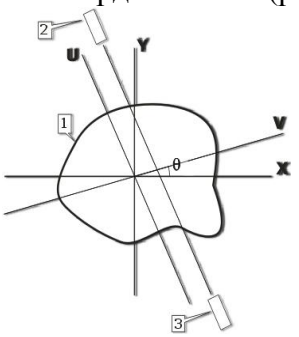


Рис. 3. Расположение объекта в системе: 1 – объект; 2 – источник излучения; 3 – приемник

Лучи определяются их расстоянием от начала координат (*x, y*) и углом поворота θ относительно неподвижной системы координат (*x, y*). Любая точка функции $f(x, y)$ может быть задана как в системе (*x, y*), так и в системе (u, v), при этом координаты связаны уравнениями преобразования (1, 2) .

$$x = v \cdot \cos(\theta) - u \cdot \sin(\theta) \qquad (1)$$
$$y = v \cdot \sin(\theta) + u \cdot \cos(\theta)$$

и наоборот:

$$v = x \cdot \cos(\theta) + y \cdot \sin(\theta) \qquad (2)$$
$$u = -x \cdot \sin(\theta) + y \cdot \cos(\theta)$$

Проекция, в общем случае, – это отображение *N*-мерной функции в (*N*-1)-мерную функцию, получаемое путем ее интегрирования в заданном направлении. В рассматриваемом двухмерном случае за направление интегрирования выбирается ось *u*, совпадающая с направлением лучей рентгеновского источника. Тогда проекция p функции f будет определена как:

$$p_\theta(v) = \int f(x, y)du \qquad (3)$$

или для подвижной системы координат, при различных θ получим набор проекций (4):

$$p_\theta(v) = \int f(u, v)du \qquad (4)$$

где θ – угол, под которым получена проекция.

Таким образом, исходным материалом для задачи восстановления является набор проекций $p_\theta(v)$, полученных под разными углами θ. В любом практическом случае количество проекций ограничено, тем не менее, при определенных допущениях восстановление можно выполнить по конечному числу проекций.

Задача восстановления, практически заключающаяся в решении интегрального уравнения (4), может быть сформулирована следующим образом: по конечному числу проекций $p_\theta(v)$, измеренных под разными углами и заданных в свою очередь дискретно, требуется восстановить значение функции *f*(*u*,*v*). Количество проекций существенно влияет на точность восстанавливаемого изображения.

Все многочисленные вычислительные алгоритмы, применяемые для решения этой задачи, чтобы найти в некотором отношении оптимальную оценку решения в заданном классе искомых функций *f*.

2. Аналитические методы

Методы, основанные на преобразовании Фурье, являются самыми естественными и мощными способами для решения поставленной задачи. В этом методе выполняется переход от функции $f(x,y)$ к ее Фурье образу $F(\omega_x,\omega_y)$ и соответственно от $f(u,v)$ к Фурье образу $F(U,V)$, где $U=\omega_x, V=\omega_y$ – повернутые на угол координаты в Фурье-плоскости.

Одним из основных свойств преобразования Фурье является его линейность, из чего следует, что если исходная функция поворачивается на угол θ, то соответствующее преобразование Фурье тоже поворачивается на угол θ. Данный метод основывается на теореме о проекциях и сечениях, которая заключается в том, что Фурье-образ искомой функции в точках Фурье-плоскости, лежащих на прямой, есть одномерный Фурье-образ соответствующей проекции. Это так называемая теорема о проекциях и сечениях (5) [7].

$$F(\omega_x,\omega_y)=F(|V|\cdot\cos(\theta),|V|\cdot\sin(\theta))=P_\theta(V) \qquad (5)$$

Теорема позволяет вычислить Фурье образ искомой функции во всех точках Фурье плоскости. Вычисляя Фурье образ проекции при разных θ, а затем по формуле обратного преобразования Фурье (6) восстановить саму функцию.

$$f(x,y)=\frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int F(\omega_x,\omega_y)\exp(-\omega_x x+i\omega_y y)d\omega_x d\omega_y \qquad (6)$$

При переходе в полярные координаты можно получить эквивалентную формулу (7), на которой построен ряд конкретных вычислительных алгоритмов.

$$f(x,y)=\frac{1}{4\pi^2}\int_0^{\pi}\Phi^{-1}\{\Phi\{p_\theta(v)\}|V|\}d\theta \qquad (7)$$

Достоинством данного метода является возможность аналитически оценить детальность получаемого изображения, а также сравнительно небольшой объем вычислений.

Основной недостаток данного метода состоит в его неустойчивости к помехам.

3. Метод обратной проекции

Данный метод оценивает плотность $f(x,y)$ в любой точке сечения посредством сложения лучевых сумм для всех лучей, проходящих через искомую точку.

Восстановление $f(x,y)$ производится путём обратного проектирования каждой проекции через плоскость, т.е. величина сигнала, соответствующая данной лучевой сумме, прикладывается ко всем точкам, которые образуют этот луч. После того, как это сделано для всех проекций, получается приближённая аппроксимация исходного объекта. Для каждой точки изображения восстановленная плотность является суммой всех лучевых проекций, которые проходят через эту точку. Поэтому метод обратной проекции иногда называется методом суммирования или линейной суперпозиции. Математическое описание метода обратной проекции может быть представлено следующим образом:

если в (7) принять $|V|=1$, тогда

$$f(x,y)=\int_0^{\pi}p_\theta(v)d\theta \qquad (8)$$

В случае измерения конечного числа проекций:

$$f(x,y)=\sum_i^M p_\theta(x\cdot\cos(\theta_i)+y\cdot\cos(\theta_i))\Delta\theta_i \qquad (9)$$

где суммирование производится по всем углам проекции:

$$v=(x\cdot\cos(\theta_i)+y\cdot\cos(\theta_i)) \qquad (10)$$

Аргумент соответствует только тем лучам, которые проходят через точку, коэффициент представляет угловое расстояние между соседними проекциями, M – количество проекций. К сожалению, величина, полученная с помощью уравнения (9), не идентична истинной плотности, за счет того, что количество проекций ограничено.

Для того чтобы обозначить причину невысокой точности метода обратной проекции, рассмотрим пример: предположим, что восстанавливаемый объект состоит из одной точки. Тогда результат восстановления по проекциям методом суммирования будет представлять собой не точку, а многолучевую звезду, центр которой находится в восстанавливаемой точке (рис. 4).

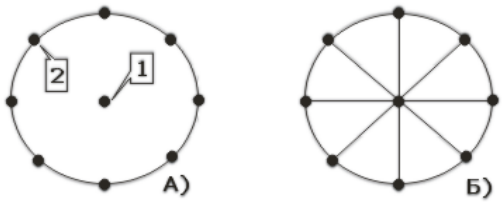


Рис. 4. Восстановление точечного объекта методом обратной проекции. а) исследуемый объект и его проекции. б) результат после обратного проецирования

Достоинством метода является высокая скорость восстановления, в чем можно убедится при моделировании.

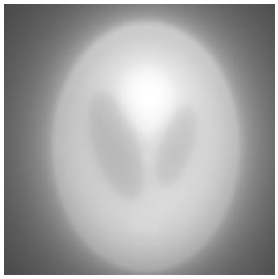


Рис. 5. Результат восстановления фантома с помощью метода обратной проекции (90 снимков)

Однако это достоинство ведет за собой недостатки: невысокая точность полученного результата (Рис. 5), которая возникает за счет недостаточного числа дискретных проекций, усечением области частот Фурье-преобразования, а также ошибки интерполяции в частотной области.

4. Метод обратной проекции с фильтрацией свертки

Данный метод аналогичен методу обратной проекции, но дает более точные результаты восстановления за счет того, что профили до обратного проецирования модифицируются или фильтруются, т.е. в (7) $|V|\neq 1$. Это преобразование позволяет вывести эффект затемнения, присущий методу обратной проекции. Именно модификация профилей достаточно точно компенсирует ложный сигнал, создающий нерезкое изображение при обратном проецировании.

Математическое обоснование данного метода приведено ниже: предположим, что есть Фурье-образ некоторой функции,

$$K(v) = \Phi^{-1}\{|V|\} = \int_{-\infty}^{\infty} |V| \exp(ivV)\,dV \qquad (11)$$

тогда произведение на Фурье-образ проекции можно рассматривать как Фурье-образ свертки самой проекции и

$$f(x,y) = \frac{1}{4\pi^2}\int_0^{\pi} \Phi^{-1}\{\Phi\{p_\theta(v) \otimes K(v)\}\}d\theta \qquad (12)$$

из (7) следует, что выражение, описывающее метод обратного проецирования с фильтрацией свёрткой при измерении конечного числа проекций будет иметь вид:

$$f(x,y) = \sum_i^M G_\theta(v)(x\cdot\cos(\theta_i) + y\cdot\cos(\theta_i))\Delta\theta_i \qquad (13)$$

где $G_\theta(v)$ – результат свертки измерений проекции с некоторой функцией, называемой ядром свертки. Как и в методе обратного проецирования – суммирование идёт по M – количеству проекций, а коэффициент представляет угловое расстояние между соседними проекциями. Так как ядро существенно влияет на качество восстанавливаемого изображения, его выбор является предметом тщательного исследования с учётом особенностей объекта, подлежащего восстановлению.

Достоинствами этого метода являются: увеличение качества и точности изображения в 15-20 раз (в этом можно убедиться, сравнив рис. 5 и 6); процесс вычислений для восстановления функции $f(x,y)$ может идти почти одновременно с получением проекции (как только получена новая проекция, осуществляется ее свертка с ядром и производятся последующие вычисления).

Рис. 6. Результат восстановления фантома с помощью метода обратной проекции с фильтрацией (90 снимков)

Недостаток заключается в более трудоемких вычислениях, а также существенно то, что данный метод не является помехоустойчивым.

5. Итерационные методы восстановления

Данный класс методов использует последовательные приближения, при которых изна-

чально выбирается произвольное изображение и рассчитываются для него проекции. Далее в изображении вводятся поправки для улучшения согласования этих проекций с проекциями полученными экспериментально. Проводится необходимое количество итераций для получения удовлетворительной сходимости.

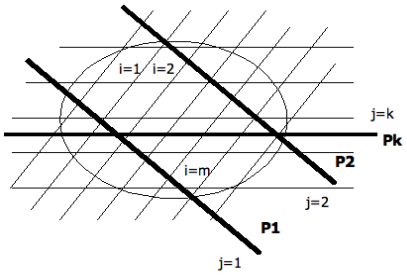


Рис. 7. Разбиение сечения на однородные участки

Алгебраические методы восстановления состоят в том, что в сечение объекта разбивается на M участков, в каждом из которых значение функции $f(x, y)$ приближенно принимается постоянным (рис. 7), можно свести интегральное уравнение (4) к системе алгебраических уравнений вида:

$$P_j = \sum_{i=1}^{M} t_{ij} \cdot f_i \qquad (13)$$

Здесь P_j – один из элементов проекции, f_i – искомое значение функции в i-ом элементе сечения, t_{ij} – геометрический коэффициент – длина пути j-го луча наблюдения в i-ом элементе сечения. t_{ij} образуют матрицу коэффициентов системы (13).

При большом числе элементов M сечения решение системы неустойчиво по отношению к погрешностям измерения P_j и даже к погрешностям вычислений. Такие системы, как правило, решают методом последовательных приближений или с использованием приемом статистической регуляризации.

Рис. 8. Результат восстановления фантома с помощью итерационного метода (90 снимков, 1 итерация)

Итерационные алгоритмы обладают следующими существенными достоинствами:

• при их построении не требуется определять обратный оператор;

• достаточно просто синтезируются нелинейные итерационные алгоритмы, учитывающие априорную информацию о восстанавливаемом изображении;

• при реализации этих методов возможна работа в интерактивном режиме, позволяющая сделать компромиссный выбор между качеством восстановления и временем обработки.

• в работах Гордона, а также Мюллера было доказано, что для восстановления изображения алгебраическими методами требуется меньшее количество проекций, нежели аналитическими [8, 9].

Однако общим недостатком итерационных алгоритмов является их низкая вычислительная эффективность, обусловленная итеративным характером вычислений. Тем не менее, ряд итерационных алгоритмов находит свое применение. К сожалению, отсутствуют формальные подходы, позволяющие определить целесообразность использования именно итерационных алгоритмов. Вопросы о том, когда их следует использовать и сколько итераций необходимо выполнить, решаются в каждом конкретном случае исходя из практического опыта.

6. Сравнение методов

В данной части рассмотрены результаты моделирования восстановления изображения с помощью наиболее оптимальных методов: итерационного метода (ART) и метода обратной проекции с фильтрацией свертки.

Моделирование выполнялось с помощью интерактивного пакета learnCT, который позволяет выполнить восстановление изображения по его проекциям различными методами с применением различных фильтров. Объектом исследований был выбран фантом Шеппа-Логана (Рис. 9).

Рис. 9. Фантом Шепп-Логан

Скорость выполнения восстановления для обоих методов приведена на графике 1.

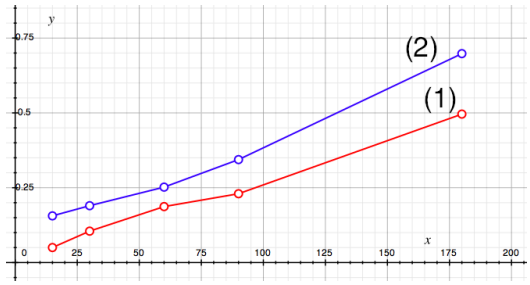
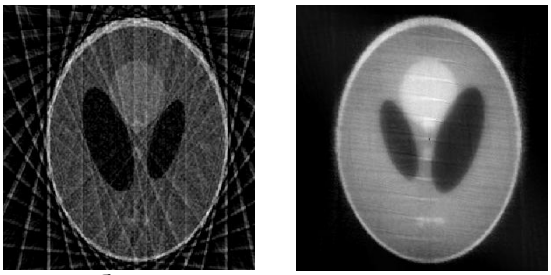


Рис. 8. График зависимости скорости восстановления от количества проекций для методов обратной проекции с фильтрацией (1) и итерационного метода (2)

Можно заметить, что скорость итерационного метода ниже скорости метода обратной проекции в 1,5-2 раза. Однако на рис. 10 видно, что итерационный метод дает более точные изображения при малом количестве проекций, нежели метод обратной проекции.



а) метод обратной проекции с фильтрацией

б) итерационный метод

Рис. 10. Результаты восстановления для 15 проекций

Заключение

Среди рассмотренных выше методов, наименее используемым методом является Фурье-метод, за счет потерь данных (при интерполировании их из полярных координат в декартовы), а также трудоемких обратных преобразований Фурье.

Метод обратной проекции может быть использован для увеличения скорости восстановления, но в случае его использования качество полученного изображения будет ниже. Повысить качество изображения в 15-20 раз можно за счет использования метода обратной проекции с фильтрацией свертки, также на качество использования данного метода будет влиять выбор ядра.

Итерационные методы более трудоемки за счет множества итераций, однако они позволяют получить более качественные изображения при малом числе снимков, нежели аналитические методы. Также итерационные методы можно оптимизировать, что увеличит скорость восстановления. Достоинство этих методов заключается в том, что при обработке в интерактивном режиме можно сделать выбор между качеством восстановления и временем обработки. Итерационные методы более легки в реализации, нежели аналитические методы, за счет того, что в последних присутствуют Фурье-преобразования. Определение же целесообразности использования итерационных алгоритмов более сложная задача.

Список литературы

1. Bech M. X-ray imaging with a grating interferometer / Martin Bech // Journal of Synchrotron Radiation. – 2009. – № 6. – С. 1-116.
2. Davis T. Phase contrast imaging of weakly absorbing materials using hard x-rays / T. Davis, D. Gao, T. Gureyev, A. Stevenson, S. Wilkins // Nature. – 1995. – № 373. – С. 595-598.
3. Bronnikov A.V. Theory of quantitative phase-contrast computed tomography / A.V. Bronnikov // Journal of the Optical Society of America. – 2002. – № 19. – С. 472-480.
4. Хермен Г. Восстановление изображений по проекциям. Основы реконструктивной томографии. / Хермен Г. – Москва: Мир, 1983. – 349 с. – (ТИИЭР; т.71 №3).
5. Цифровая обработка изображений в информационных системах / [И.С. Грузман, В.С. Киричук, В.П. Косых, Г.И. Перетягин, А.А. Спектор] – Новосибирск: Изд-во НГТУ, 2002. . – 352 с.
6. Тихонов А.Н. Математические задачи компьютерной томографии. / Тихонов А.Н., Арсенин В.Я., Тимонов А.А. – Москва: Наука, 1987. – 160 с.
7. Kak A.C. Principles of Computerized Tomographic Imaging / A.C. Kak, Malcolm Slaney // Engineering. – 1988. – № 33.1. – С. 327.
8. Wojciech Chlewicki. 3D Simultaneous Algebraic Reconstruction Technique for Cone-Beam Projections: Master of Science Thesis / Wojciech Chlewicki. – UNIVERSITY OF PATRAS, 2001. – С. 57.
9. Guan, H. Computed tomography using algebraic reconstruction techniques (ARTs) with different projection access schemes: a comparison study under practical situations. / Guan H., Gordon R. // Physics in Medicine and Biology. – 1996. – № 9. – С. 1727-1743.

УДК 004.724.4(045)

КУЛАКОВ А.Ю. ,
БОГАТЫРЧУК И.И.,
ГНАТЮК О.С.

СПОСОБ ИЕРАРХИЧЕСКОГО ПЛАНИРОВАНИЯ ЗАДАЧ В GRID-СИСТЕМАХ

Предложен усовершенствованный способ иерархического планирования. Повышение эффективности планирования достигается за счет физических каналов связи у процессоров вычислительного узла и, возможности поддержки дуплексного режима передачи информации.

The improved method for hierarchical planning is proposed. Efficiency increasing of planning is achieved by physical links of processor's compute node and ability to support duplex communication.

Введение

В связи с расширением сферы использования и увеличением нагрузки на Grid-системы [1] актуальным становиться задача повышения эффективности их функционирования. Специфическая проблема, которая лежит в основе концепции Grid-системы, – это скоординированное совместное использование разнороных, динамически меняющихся ресурсов [2]. Эффективность решения данной задачи в значительной степени зависит от способа планирования и распределения задач в системе [3].

В зависимости от характера решаемых задач и типа Grid-систем в них используются как статические [4,5] , так и динамические алгоритмы планирования [6,7,8]. При этом используются три основных способа планирования: *централизованный, децентрализованный* и *иерархический* [9].

Централизованный способ планирования плохо масштабируется с увеличением количества ресурсов, поэтому этот способ пригоден лишь для Grid-систем с ограниченным числом узлов.

Децентрализованный способ планирования обеспечивает лучшую отказоустойчивость и надежность, по сравнению с централизованным способом, однако, отсутствие метапланировщика, который обладает информацией обо всех приложениях и ресурсах, снижает эффективность процесса планирования.

Следует отметить, что задача планирования задач в Grid-системе является NP-полной [10], сложность которой является нелинейной функцией от размерности системы, то есть от количества ее узлов. С целью уменьшения сложности задачи планирования в Grid системах большой размерности используются иерархические схемы планирования [11].

Модифицированный способ планирования задач

В работе [12] предложен иерархический способ планирования, особенностью которого является улучшенная возможность адаптации к определенным условиям, таким как структура пользовательского приложения и состав вычислительных ресурсов. К недостаткам данного подхода следует отнести то, что не учитывается наличие у процессоров узла множества физических каналов связи с другими процессорами. Это естественно сказывается на эффективности процесса планирования задач.

В общем случае задача планирования заключается в следующем. Пусть имеется Grid система, состоящая из K узлов. Каждый i–й узел состоит из Pi процессоров (компьютеров). Задано приложение из m подзадач с помощью DAG графа. Известны также топологии каждого из K узлов системы, описанные с помощью неориентированных графов. Требуется найти i–й узел Grid системы, который обеспечивает минимальное общее время выполнения заданного приложения $\{T_i \,|i=1,...,K\}$.

$$T_i = \sum_{i=1}^{m} \sum_{l=1}^{P_i} t_{jli} \times X_{jli} + S_i + \max\{ Tr , Tf_i \} \quad ,$$

где: t_{jli} – время выполнения j–й подзадачи в l–м процессоре i–го узла Grid системы;

S_i – время доставки входных данных и результатов приложения в (из) i–го узла Grid системы;

Tr – время готовности приложения к выполнению в узлах системы;

Tf_i – время освобождения i–го узла Grid системы для выполнения заданного приложения в эксклюзивном режиме;

$X_{jli} = 1$, если j–й подзадача выполняется в l–м процессоре i–го узла;

$X_{jli} = 0$ в противном случае.

В данной работе в качестве основного показателя эффективности используется коэффициент ускорения выполнения задачи

$$K_y = T_1/T_n,$$

где: T_1 – время выполнения задачи на одном процессоре узла Grid системы, T_n – время выполнения задачи на процессорах узла Grid системы.

При этом коэффициент эффективности функционирования системы равен:

$$K_{эс} = \frac{K_y}{N},$$

где: K_y – коэффициент ускорения, N – количество процессоров в узле Grid системы.

Эффективность работы планировщика определяется по формуле:

$$K_{эа} = \frac{\max(T_{кр}; \frac{\sum w_i}{N})}{T_n},$$

где: N – количество процессоров, $T_{кр}$ – критическое время задачи, w_i-вес i-й вершины, T_n– время выполнения задачи на заданной вычислительной системе.

С целью повышения эффективности планирования приложений в вычислительных Grid системах предлагается модификация иерархического способа за счет учета:

- количества физических каналов связи процессора узла, соединяющих его с другими процессорами;
- режима дуплекс/полудуплекс канала процессора при выполнении планирования.

Рассмотрим основные этапы предлагаемого иерархического способа планирования для вычислительных Grid систем.

1. Пользователь представляет приложение в виде графа задачи (рис.1) и формирует ресурсный запрос, с учетом заданного критерия оптимизации.

2. **Метадиспетчер имеет следующую информацию об узлах: стоимость работы каждого узла за единицу времени, а также скорость доставки данных в каждый узел.**

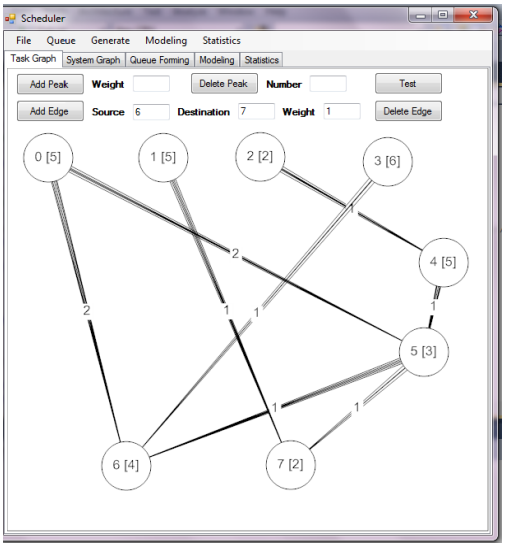


Рис.1

3. Метадиспетчер посылает локальным менеджерам граф задачи и получает в ответ возможное время запуска на каждом из них. Для определения этого времени на каждом из локальных ресурсов выполняется планирование заданного приложения.

4. Локальные менеджеры сообщают возможное время освобождения своих узлов для определения начального времени выполнения приложения.

5. В зависимости от заданного пользователем критерия оптимизации и имеющихся данных, метадиспетчер выбирает оптимальный узел Grid системы и посылает туда запрос на резервирование ресурсов, причем резервирование должно быть создано таким образом, чтобы воспользоваться зарезервированными ресурсами могло только это приложение.

6. Если резервирование выполнено, приложение посылается в очередь выбранного узла. В том случае если ресурсы, необходимые для запуска задания освободятся раньше времени начала резервирования, необходимо обеспечить разрешение запуска приложения.

В рамках данной работы был проведен ряд экспериментов, направленных на сравнение эффективности предложенного алгоритма планирования для однородных узлов (ПА) с эффективностью базового алгоритма (БА), описанного в работе [12].

Для сбора статистики задавались такие переменные:

- Дуплексность.
- Количество физических линков (PL).
- Количество вершин графа задачи (Nв).

Моделирование выполнялось при таких параметрах:

- Веса вершин в диапазоне 1…Nв/2.
- Веса пересылок в диапазоне 1… Nв/2.
- Связность в диапазоне 10…90% с шагом 10% (C).
- Отношение количества вершин графа задачи к количеству вершин графа системы 5 : 1, 10 : 1. (D).
- Топологии: полносвязная на 4 узла, звезда на 4 узла.
- Количество физических линков: 3, 1 (PL)

На рис. 2. приведен пример формирования очереди из графа задачи.

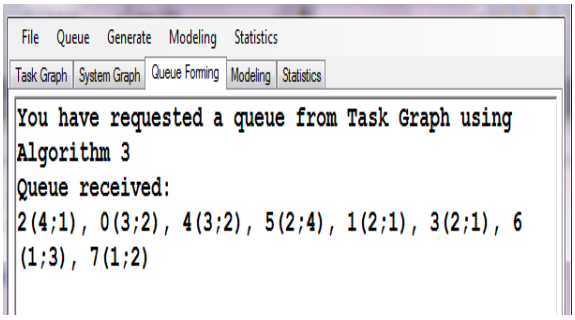


Рис. 2

Результатом погружения является модифицированная диаграмма Ганта (рис.3), в которой кроме расписаний работы каждого процессора отображено также расписание работы каждого канала связи.

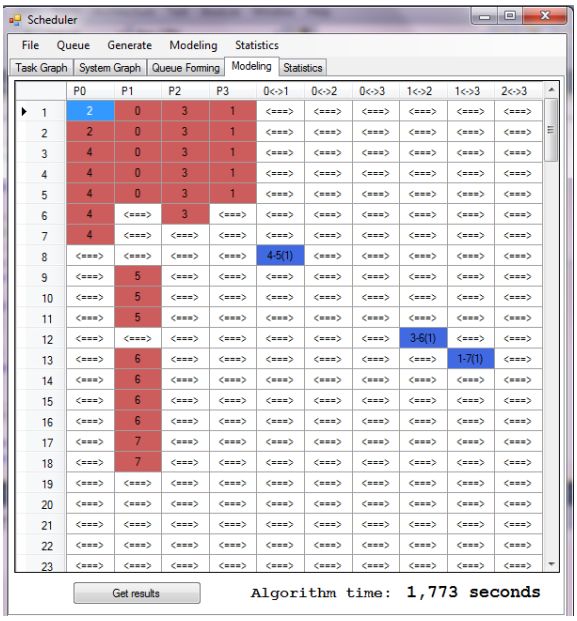


Рис. 3

В результате моделирования были получены результаты, позволяющие оценить влияние количества физических линков процессора (рис.4) и режима дуплекс на коэффициент ускорения (рис.5).

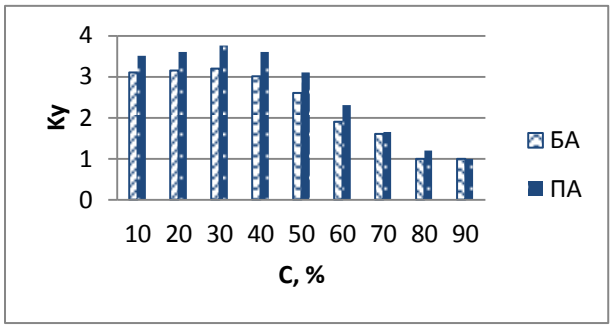


Рис. 4

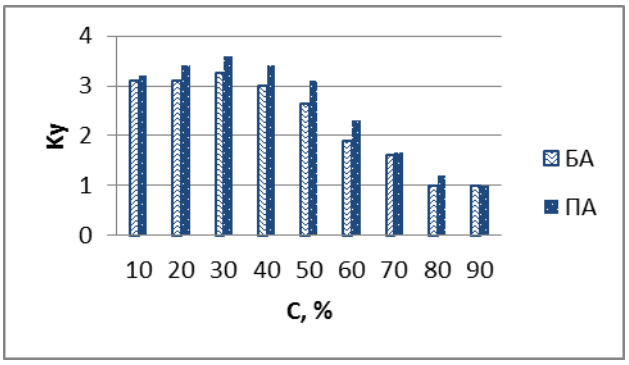


Рис. 5

На рис.6 представлена зависимость коэффициента ускорения от связности задачи с учетом физических линков и дуплексного режима передачи информации.

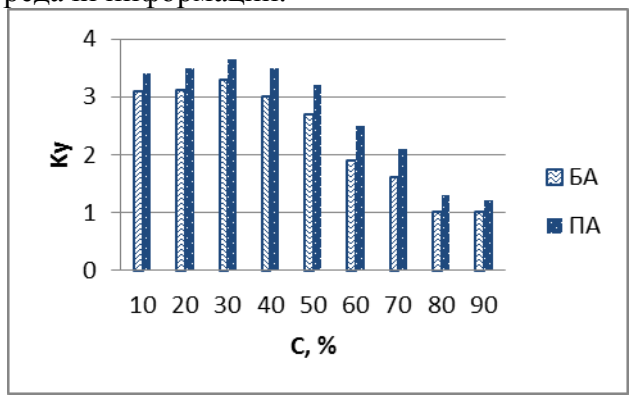


Рис. 6

Очевидно, что учет наличия трех физических линков у каждого процессора узла позволяет увеличить коэффициент ускорения в среднем на 10-20%. Такой результат достигается за счет одновременного выполнения пересылок данных по нескольким каналам.

Учет дуплексного режима канала процессора увеличивает коэффициент ускорения в среднем на 5-15%. Это обусловлено тем, что при высоких связностях задачи в графе возникает много перекрестных связей, в результате чего у процессоров возникает необходимость одновременно передавать данные друг другу. Если же учесть обе характеристики системы, то, как следует из рис. 6, прирост коэффициента ускорения по сравнению с базовым алгоритмом,

представленным в работе [12], составит в среднем 20-25%. Производительность узла убывает по мере увеличения связности задачи. Это связано с большим количеством пересылок. Наиболее близкий к оптимальному коэффициент ускорения наблюдается на отметке 30% связности задачи.

Заключение

Предложенный в данной работе усовершенствованный иерархический способ планирования задач позволяет достигнуть более высокой эффективности планирования за счет учета количества физических каналов связи у процессоров конкретного вычислительного узла, а так же учета возможности поддержки каналами связи процессоров дуплексного режима.

Выполнена практическая реализация предложенного алгоритма в виде программного продукта, который может быть использован для моделирования планирования задач в GRID-системах.

Список литературы

1. I. Foster, C. Kesselman. Computational Grids. Chapter 2 of “The Grid: Blueprint for a New Computing Infrastructure”, Morgan-Kaufman, 1999.
2. I. Foster, C. Kesselman and S. Tuecke, The Anatomy of the Grid: Enabling Scalable Virtual rganizations, in the International J. Supercomputer Applications, 15(3), pp. 200-220, fall 2001.
3. Y. Zhu, A Survey on Grid Scheduling Systems, Department of Computer Science, Hong Kong University of science and Technology, 2003.
4. S.Y. You, H.Y. Kim, D. H. Hwang, S. C. Kim, Task Scheduling Algorithm in GRID Considering Heterogeneous Environment, in Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications, PDPTA '04, pp. 240-245, Nevada, USA, June 2004.
5. Симоненко В.П., Куреньов А.С. Алгоритм статического планирования для GRID систем.-Вісник НТУУ "КПІ".Сер. Інформатика, управління та обчислювальна техніка. - 2011. Випуск 53. - С.10 – 17.
6. A.Takefusa, S. Matsuoka, H. Casanova and F. Berman, A Study of Deadline Scheduling for Client-Server Systems on the Computational Grid, in Proc. of the 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10'01), pp. 406--415, San Francisco, California USA, August 2001.
7. N. Muthuvelu, J. Liu, N. L. Soe, S.r Venugopal, A. Sulistio and R. Buyya, A Dynamic Job Grouping-Based Scheduling for Deploying Applications with Fine-Grained Tasks on Global Grids, Proceedings of the 3rd Australasian Workshop on Grid Computing and e-Research (AusGrid 2005), Newcastle, Australia, January 30 – February 4, 2005.
8. K.Kurowski, B.Ludwiczak, J.Nabrzyski, A.Oleksiak and J.Pukacki, Improving Grid Level Throughput Using Job Migration And Rescheduling, Scientific Programming vol.12, No.4, pp. 263-273, 2004
9. V. Hamscher, U. Schwiegelshohn, A. Streit, R. Yahyapour, Evaluation of Job-Scheduling Strategies for Grid Computing, in Proc. of GRID 2000 GRID 2000, First IEEE/ACM International Workshop, pp. 191-202, Bangalore, India, December 2000.
10. H. El-Rewini, T. Lewis, and H..Ali, Task Scheduling in Parallel and Distributed Systems, ISBN: 0130992356, PTR Prentice Hall, 1994.
11. T. Casavant, and J. Kuhl, A Taxonomy of Scheduling in General-purpose Distributed Computing Systems, in IEEE Trans. on Software Engineering Vol. 14, No.2, pp. 141--154, February 1988.
12. Кулаков Ю.А., Русанова О.В., Шевело А.П. Иерархический способ планирования для GRID.- систем - Вісник НТУУ "КПІ".Сер. Інформатика, управління та обчислювальна техніка. - 2009. Випуск 51. - С.57– 66.

УДК 004.75

СЛЮСАР Є.А.,
БОЙКО Ю.В.

ОРГАНІЗАЦІЯ ЦЕНТРАЛЬНОГО КЕШУЮЧОГО КАТАЛОГУ РЕСУРСІВ В УКРАЇНСЬКІЙ НАЦІОНАЛЬНІЙ ГРІД-ІНФРАСТРУКТУРІ

Проведено аналіз існуючих проблем швидкості роботи інформаційної системи при роботі з грід-порталами у інфраструктурах на базі програмного забезпечення Nordugrid ARC. Розроблено методики кешування відомостей локальних інформаційних служб грід-ресурсів для зменшення часу відгуку при опитуванні стану запущених грід-завдань. Відповідно до представленої методики реалізовано службу кешуючого каталогу ресурсів. Розроблену програмну бібліотеку для взаємодії зі службою впроваджено в грід-порталах віртуальних організацій MolDynGrid та NetworkDynamics в українській національній грід-інфраструктурі.

Analysis of existing responsiveness problems of grid information system focused on implementation of grid-portals employing Nordugrid ARC middleware is conducted. Techniques for centralized caching of data from local grid-resource information services were developed. Proposed techniques were implemented in the caching information index service. A software library developed for accessing the cache is incorporated into the grid-portals of MolDynGrid and NetworkDynamics virtual organizations of Ukrainian national grid-infrastructure.

1. Вступ

Грід-технології є потужним інструментом наукових досліджень як фундаментальних так і прикладних галузей науки і техніки. Вирішення багатьох громіздких розрахункових задач, що зазвичай постають перед цими видами діяльності та потребують потужних інформаційно-обчислювальних ресурсів, можна здійснити шляхом залучення ресурсів національних та міжнародних грід-інфраструктур. Політики використання ресурсів узгоджуються між власниками цих ресурсів та віртуальними організаціями – об'єднаннями людей для вирішення спільної наукової чи технічної задачі, що використовують грід-ресурси згідно з погодженими правилами. Будь-яка віртуальна організація має доступ до певного набору ресурсів, які надаються користувачам, що зареєстровані в ній. З іншого боку, кожен ресурс може надаватися у користування одразу декільком віртуальним організаціям. Кожна віртуальна організація самостійно встановлює правила роботи для своїх учасників, виходячи із балансу між потребами досліджень та наявними ресурсами. [1]

Грід-інфраструктури створюються на базі існуючих інформаційних та обчислювальних ресурсів, таких як обчислювальні кластери, сховища та бази даних, із використанням багаторівневої програмної архітектури, що виступає проміжним елементом між цими ресурсами та користувачами. [2] Реалізації такого програмного забезпечення проміжного рівня, визначають набір протоколів та служб і, відповідно, інтерфейси до них. Пакети програмного забезпечення, що застосовуються в українському національному грід-сегменті – пакети Nordugrid Advanced Resource Connector (ARC) [3] та gLite – надають користувачеві лише інтерфейс командного рядка для роботи із службами рівнів кооперації та ресурсів грід-архітектури. Інтерфейс командного рядку не є тривіальним у використанні і потребує додаткових знань як командної оболонки UNIX, так і специфічних для грід-середовища мов семантичного опису завдань. З метою спрощення використання грід-інфраструктур науковцями та іншими спеціалістами, розробляються та впроваджуються веб-орієнтовані інтегровані середовища роботи – грід-портали, що використовують інтерфейс веб-браузера для взаємодії з користувачем. Такі спеціалізовані веб-портали розгортаються в межах однієї чи декількох віртуальних організацій та дозволяють досліднику сфокусуватись на галузі своїх досліджень, приховуючи від нього деталі організації процесу обчислень та внутрішню реалізацію механізмів взаємодії з грід-інфраструктурою.

2. Постановка задачі

Незалежно від особливостей організації процесу обчислень на грід-порталі, він безумовно реалізує певний обмежений набір механізмів взаємодії із грід-інфраструктурою. Зокрема, серед них можна виділити такі операції:

- формування опису грід-завдання та направлення його на виконання;
- опитування стану завдань, що були направлені до грід-інфраструктури;
- керування завданням – призупинення, скасування, видалення, тощо.
- отримання результатів завдання, що завершилось;

Операції направлення завдання та отримання результатів в залежності від реалізації порталу можуть одночасно обробляти як одиничне завдання, так і цілу множину завдань. Операція отримання результатів роботи завдання може бути викликана лише після того, як надійде інформація про факт завершення цього завдання. Однією з найбільш критичних операцій є опитування стану завдань, оскільки, на відміну від інших перелічених операцій, має виконуватись періодично і стосується усього набору завдань, направлених грід-порталом на виконання. Таким чином, грід-портал може за запитом користувача або автоматичним чином направляти розрахункові завдання до грід-інфраструктури, проте подальші процедури обслуговування завдань викликаються як реакція на події зміни стану цих завдань, що генеруються підсистемою моніторингу. Отже, час реакції грід-порталу напряму залежить від оперативності отримання цих подій.

Пакет програмного забезпечення проміжного рівня gLite має у своєму складі окрему службу посередника ресурсів – систему балансування навантаження (Workload Management System, WMS), що окрім власне підбору грід-ресурсів під вимоги завдання самостійно здійснює направлення та контроль за виконанням завдань. Служба має зовнішній інтерфейс на основі відкритого протоколу SOAP, що може використовуватись грід-порталами для отримання подій моніторингу завдань. У складі глобальної грід-інфраструктури WLCG наявні інсталяції WMS, що одночасно обслуговують сотні завдань. [4]

У пакеті програмного забезпечення Nordugrid ARC окрема служба посередника ресурсів відсутня. Натомість, функція підбору обчислювальних елементів інтегрована до програмних бібліотек інтерфейсу користувача та викликається безпосередньо у контексті операції направлення завдання. [3] Програмний інтерфейс до опитування стану завдання передбачає подачу на вхід лише одного ідентифікатора завдання. Отримання поточного стану одного завдання займає декілька секунд, що не є прийнятним для масової обробки завдань грід-порталом.

З огляду на те, що в українському національному грід-сегменті більшість грід-ресурсів функціонують під управлінням саме програмного забезпечення Nordugrid ARC [5], актуальною є задача розробки методики швидкого та масового опитування стану грід-завдань для подальшого впровадження у грід-порталах українських віртуальних організацій.

3. Кешуючий каталог ресурсів.

Висока тривалість операції отримання стану завдання пов’язана із особливостями реалізації інформаційної системи, яка у грід-інфраструктурі на основі програмного забезпечення Nordugrid ARC представлена двома службами:

- система інформації грід-ресурсу, ARC Resource Information System (ARIS),
- служба каталогу інформації грід, Enhanced Grid Information Index Service (EGIIS).

ARIS – це служба рівня ресурсів, що працює у складі обчислювального елемента чи сховища даних та надає своїм клієнтам відомості про стан відповідного грід-ресурсу згідно стандартизованої інформаційної моделі за протоколом LDAP. Внутрішня реалізація служби включає в себе стандартний LDAP-сервер та сценарії-колектори, що наповнюють та періодично оновлюють базу даних цього сервера інформацією про стан грід-ресурсу, отриману через внутрішні інтерфейси постачальника інформації служб обчислювального елемента чи сховища даних. Для обчислювального елемента, зокрема, служба ARIS публікує стан усіх грід-завдань, що очікують у черзі, виконуються чи вже завершились на відповідному обчислювальному кластері. Таким чином, база даних LDAP-сервера являє собою кеш відомостей про грід-ресурс, що регулярно оновлюється під час роботи сценаріїв-колекторів.

Служба EGIIS являє собою динамічний каталог посилань на примірники служб ARIS та EGIIS, що також має зовнішній інтерфейс на основі протоколу LDAP. Грід-ресурси реєструють у каталозі посилання на власний примірник служби ARIS чи EGIIS за допомогою спеціального LDAP-запиту. Кожен запис у каталозі має обмежений час життя, що за замовчуванням становить 60 секунд, тому операція реєстрації грід-ресурсу має повторюватись із відповідною періодичністю. Така схема роботи забезпечує

актуальність відомостей у каталозі, оскільки реєстрацію підтверджують лише ті ресурси, що функціонують у штатному режимі.

Інформаційну систему грід-інфраструктури можна представити у вигляді динамічної ієрархічної структури, що може складатись з одного дерева або лісу дерев. Кореневою вершиною кожного дерева є каталог ресурсів EGIIS вищого рівня, проміжні вершини представляють каталоги ресурсів нижчих рівнів, а кінцеві вершини – зареєстровані грід-ресурси. У загальному випадку, кожен ресурс може реєструватись у більш ніж одному каталозі для забезпечення високої доступності інформаційної системи. [6]

У великих грід-інфраструктурах, таких як колаборація Nordugrid, до складу якої входить і український грід-сегмент, організовано три рівні ієрархії інформаційної системи – загальний, рівень країн та рівень ресурсів (Рис. 1). Для пошуку вільних обчислювальних кластерів із заданими характеристиками, вбудований до інтерфейсу користувача посередник ресурсів на першому кроці здійснює обхід кожного дерева із лісу дерев інформаційної системи, починаючи з кореневих каталогів вищого рівня. У процесі обходу формується список усіх адрес доступу до відповідних зареєстрованих примірників служби ARIS. На другому кроці виконується опитування кожного примірника та отримується необхідна інформація про властивості та поточний стан відповідного грід-ресурсу. Операція побудови списку грід-ресурсів із лісу дерев інформаційної системи також виконується і під час здійснення опитування стану грід-завдання. Таким чином кількість LDAP-запитів, які здійснює інтерфейс користувача, складається із загальної кількості каталогів ресурсів у інфраструктурі та власне потрібних для відповідної операції грід-ресурсів.

Зауважимо, що каталоги ресурсів містять лише посилання на інші служби, будь-яких інших додаткових відомостей в них не публікується. Тому незалежно від приналежності до віртуальної організації та типу запиту, будь-який користувач чи застосування-портал грід-інфраструктури для виконання операції із грід-завданнями виконують ідентичний набір операцій та за умови стабільності грід-інфраструктури отримують ідентичні списки ресурсів.

Загальну кількість LDAP-запитів можна зменшити до одного запиту, якщо винести операції обходу інформаційної системи та отримання відомостей із ARIS-служб грід-ресурсів до **окремого кешуючого каталогу**. Це також дозволяє за один запит отримати відомості про масив завдань, що можуть виконуватися на різних грід-ресурсах.

Проте, запровадження нової служби вимагає модифікації схем взаємодії грід-порталу із інфраструктурою. З огляду на критичність до часу виконання основних операцій грід-порталу над грід-інфраструктурою, проаналізованих вище, оптимальним є запровадження нового компоненту для масового опитування стану завдань та використання стандартних засобів пакету ARC для здійснення інших операцій. Таким чином зберігається сумісність, оскільки нові програмні засоби запроваджуються лише на грід-порталах і примірниках служби кешуючого каталогу, та не впливають на функціонування інших компонентів грід-інфраструктури.

Кешуючий каталог ресурсів доцільно реалізувати із залученням готових компонентів, що входять до внутрішньої реалізації служби ARIS. У поточних версіях пакету ARC у якості керуючого LDAP-сервера використовується Berkeley Database Information Index (BDII) версії 5 – компонент, що включає в себе засоби запуску та керування конфігурацією відкритого LDAP-серверу OpenLDAP та службу наповнення та оновлення бази даних цього сервера, яка періодично викликає зовнішні сценарії-колектори, що повертають інформацію у форматі LDIF. На кожному кроці результати роботи сценаріїв-колекторів впорядковуються та порівнюються із поточним вмістом бази даних LDAP, у результаті чого формуються запити на зміну вмісту цієї бази даних, які направляються на LDAP-сервер. Таким чином оновлення бази даних LDAP не потребує припинення обслуговування зовнішніх клієнтів служби. [7]

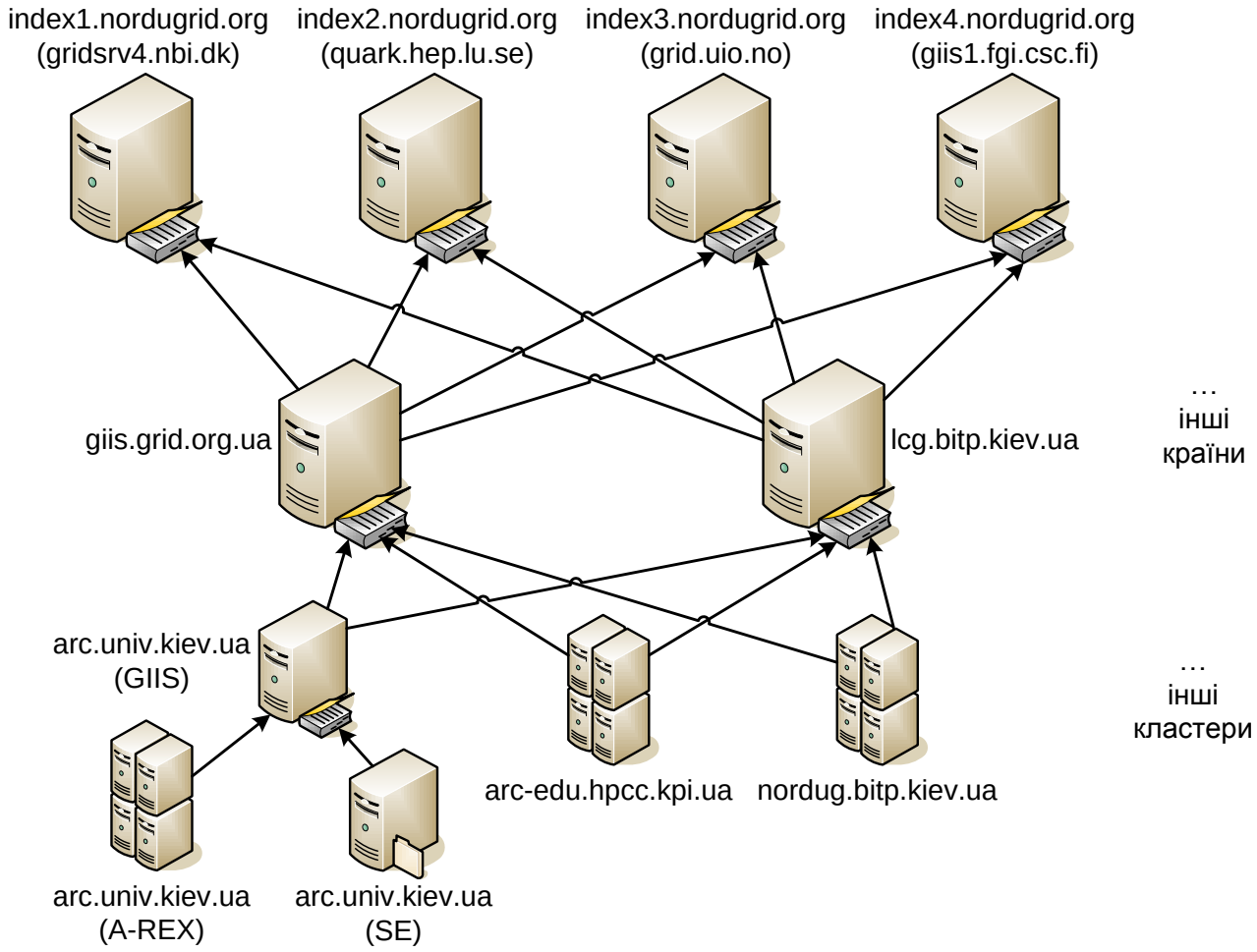


Рис. 1. Структура інформаційної системи колаборації Nordugrid

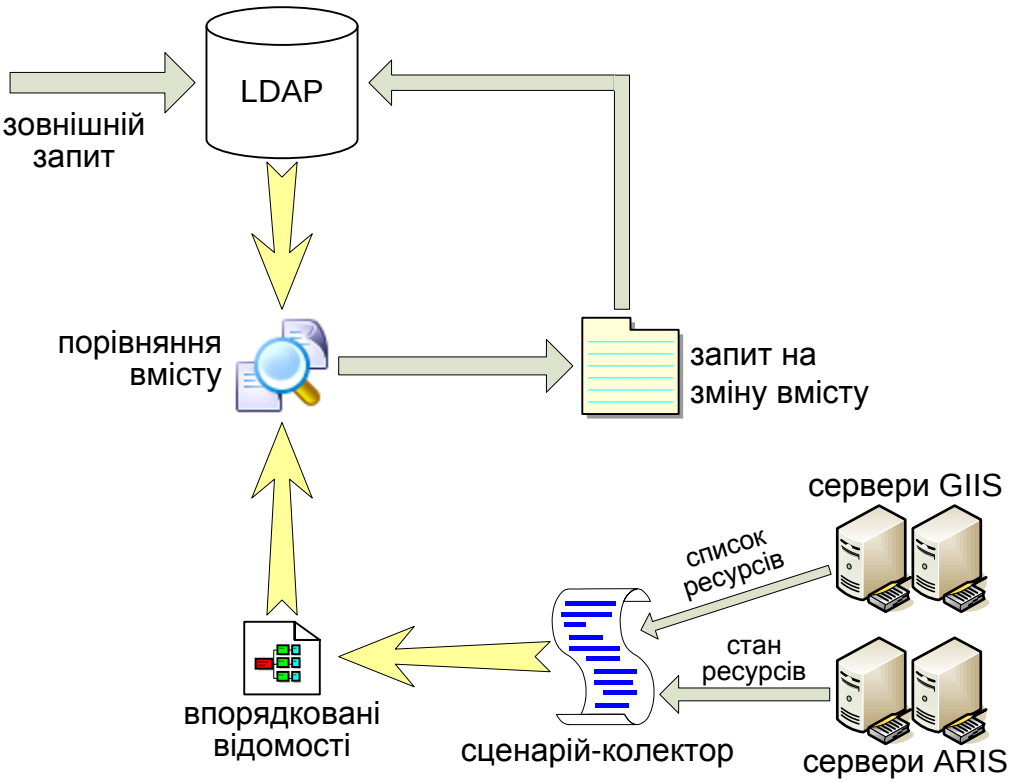


Рис.2. Взаємодія компонентів служби кешуючого каталогу ресурсів

Отже, відмінність кешуючого каталогу грід-ресурсів від стандартної служби ARIS полягає у застосуванні спеціального сценарія-колектора, який виконує обхід лісу дерев інформаційної системи грід-інфраструктури для формування списку примірників ARIS, а потім завантажує повний вміст баз даних усіх виявлених на попередньому етапі примірників та видає його у форматі LDIF на вхід системи BDII. Така реалізація дозволить клієнтам служби здійснювати будь-які запити до кешованих відомостей про грід-ресурси.

4. Реалізація

Схему взаємодії компонентів служби кешуючого каталога ресурсів наведено на Рис. 2. Для реалізації сценарію було обрано мову Perl з огляду на вбудовану підтримку регулярних виразів та гнучкі засоби керування процесами. Звернення до серверів ARIS та EGIIS виконуються засобами клієнтських утиліт OpenLDAP, при чому кожен виклик зовнішнього застосування має жорсткі обмеження за часом роботи для попередження «залипання» батьківського сценарію. Для скорочення тривалості кожного циклу оновлення бази даних опитування серверів служби ARIS реалізовано паралельно, за замовчуванням – у 10 потоків.

Для українського національного грід-сегменту, що загалом налічує близько 25 обчислювальних кластерів, середній час обходу усіх інформаційних служб сценарієм складає в середньому 4 хвилини. Така довга тривалість обумовлена тим, що деякі грід-ресурси, що підтримують свою реєстрацію у каталогах EGIIS, з технічних причин є недоступними з вузла служби кешуючого каталога, і спроби з'єднання із ними припиняються через тайм-аут. Час обслуговування зовнішнього запиту LDAP-сервером складає 1 секунду, що в 10 разів менше за час виконання стандартної утиліти `ngstat` пакету ARC, що отримує стан грід-завдання традиційним способом.

З точки зору клієнта служба кешуючого каталогу являє собою звичайний LDAP-сервер, вміст бази даних якого можна переглянути за допомогою будь-якого LDAP-браузера (Рис. 3), а програмно – за допомогою стандартних інтерфейсів доступу до LDAP-каталогів.

Для масового опитування стану грід-завдань необхідно сформувати критерії фільтрації результатів на стандартній мові опису фільтрів LDAP. Для інформаційної моделі Nordugrid Schema, що застосовується у програмному забезпеченні проміжного рівня Nordugrid ARC, такий фільтр має вигляд:

```
(|(nordugrid-job-globalid=JID1)
(nordugrid-job-globalid=JID2)(...)),
```

де замість `JID1`, `JID2`, … необхідно підставити глобальні ідентифікатори грід-завдань у тому ж вигляді, як вони були отримані після направлення завдання до грід-інфраструктури.

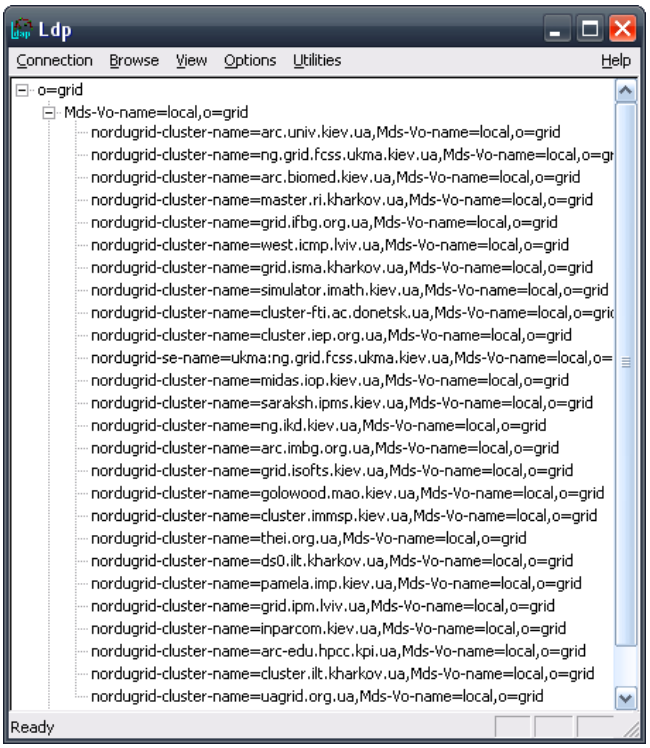


Рис. 3. Вміст бази даних LDAP-сервера служби кешуючого каталогу

Кількість ідентифікаторів завдань, що може бути передана за допомогою фільтру в межах одного запиту складає приблизно 700 одиниць для конкретного примірника служби каталогу `ldap://bdii.grid.org.ua:2170/o=grid` та визначається із внутрішнього обмеження LDAP-сервера OpenLDAP.

5. Інтеграція з Грід-порталами

Для інтеграції реалізованої служби кешуючого каталогу грід-ресурсів із грід-порталами була розроблена бібліотека для середовища PHP, яка реалізує функції масового опитування стану грід-завдань та для доступу до служби використовує стандартне розширення `php_ldap`. Бібліотека автоматично виконує розбиття масиву ідентифікаторів грід-завдань на блоки розміром у 512 елементів, формує для кожного із них відповідний опис LDAP-фільтру та здійснює запит до сервера. Результати деякі-

лькох запитів поєднуються та заповнюються у попередньо відведені комірки вихідного масиву станів.

Розроблена бібліотека була впроваджена до реалізацій грід-порталів віртуальної лабораторії в межах віртуальної організації MolDynGrid [8] та лабораторії моделювання нелінійних процесів Національного наукового центру з медико-біотехнічних проблем при президії НАН України у складі віртуальної організації NetworkDynamics [9]. Особливість останнього грід-порталу полягає в тому, що він дозволяє користувачам запускати великі серії грід-завдань математичного моделювання із розкидом параметрів, які можуть налічувати тисячі окремих грід-завдань. Використання служби кешуючого каталогу ресурсів дозволило значно розвантажити сервер грід-порталу та оперативно отримувати результати моделювання таких великих серій завдань для їх подальшого зведення та повернення користувачеві.

6. Висновки

В результаті проведеного аналізу реалізацій грід-порталів визначено, що ключові обмеження швидкості роботи пов'язані із використанням стандартних засобів інтерфейсу користувача для керування завданнями у грід-інфраструктурах, побудованих засобами програмного забезпечення проміжного рівня Nordugrid ARC. Було показано, що великий час відгуку стандартної операції опитування стану завдання пов'язаний із особливостями реалізації інформаційної системи таких грід-інфраструктур.

Розроблено методики кешування відомостей із локальних інформаційних служб грід-ресурсів для зменшення часу відгуку при опитуванні стану грід-завдань. Показано, що запровадження центрального кешуючого каталогу грід-ресурсів дозволить збільшити швидкість опитування інформаційної системи та здійснювати вибірку стану цілого масиву грід-завдань за один запит.

Представлена методика була реалізована за допомогою спеціалізованого багатопотокового сценарія-колектора мовою Perl для служби інформації ARIS. Для середовища PHP створено програмну бібліотеку масового опитування стану грід-завдань, та виконано інтеграцію служби керуючого каталогу з грід-порталами віртуальних організацій MolDynGrid та NetworkDynamics у складі української національної грід-інфраструктури.

Представлені методики та програмні рішення можуть бути використані для оптимізації інших грід-служб, що взаємодіють з інформаційною системою грід.

Перелік посилань

1. Foster, Ian. The physiology of the grid: An open grid services architecture for distributed systems integration / Ian Foster. [Online] – 2002. http://www.globus.org/research/papers/ogsa.pdf
2. Foster, Ian. The Anatomy of the Grid – Enabling Scalable Virtual Organizations / Ian Foster, Carl Kesselman, Steven Tuecke // International Journal of Supercomputer Applications. – 2001. – Vol. 15.
3. Ellert, M. Advanced Resource Connector middleware for lightweight computational Grids / M. Ellert et al. // Future Gener. Comput. Syst. – 2007. – Vol. 23, no. 1. – Pp. 219–240.
4. Andreetto, P. The gLite Workload Management System / P. Andreetto et alt. // in Proc. Computing in High Energy and Nuclear Physics (CHEP) 2007, Victoria, British Columbia (CA), Sep 2007.
5. Бойко Ю.В. Український академічний Грід: досвід створення й перші результати експлуатації / Ю.В.Бойко, М.Г.Зинов'єв, О.О.Судаков, С.Я.Свістунов // Математичні машини і системи. – 2008. – Вип. 1. – ст. 67–84.
6. Balazs Konya. The Nordugrid-ARC Information System / Balazs Konya, Daniel Johansson // NORDUGRID-TECH-4 [Online] – 16/6/2011. http://www.nordugrid.org/documents/arc_infosys.pdf
7. Field, L. Berkeley Database Information Index V5 / Laurence Field // CERN TWiki [Online] – 9 Aug 2010. https://twiki.cern.ch/twiki/bin/rdiff/EGEE/BDII
8. Salnikov Andrii. Virtual Laboratory MolDynGrid as a Part of Scientific Infrastructure for Biomolecular Simulations / A.O. Salnikov, I.A. Sliusar, O.O. Sudakov et al. // International Journal of "Computing". – 2010. – Vol. 9, no. 4.
9. Andrii Salnikov, Roman Levchenko, Oleksandr Sudakov. Integrated Grid Environment for Massive Distributed Computing in Neuroscience. Proceedings Book of the 6-th IEEE International Conference IDAACS 2011, 15-17 September 2011, Prague, Czech Republic, pp. 198-202.

УДК 004.588,
377.111.3,
377.131.11,
377.131.14

НОВІКОВ Ю.Л.

КОНЦЕПЦІЯ СТВОРЕННЯ ЄДИНОГО ІНФОРМАЦІЙНОГО РЕСУРСУ НАВЧАЛЬНИХ МАТЕРІАЛІВ

У статті представлено концепцію створення та впровадження єдиного підходу до інтеграції існуючих і новостворюваних ресурсів навчальних матеріалів в єдине освітнє середовище. Визначено універсальний механізм представлення навчального матеріалу будь-якого типу для можливості конвертації ресурсів з однієї системи дистанційного навчання у іншу з метою спільного використання напрацьованого надбання спеціалістів різних навчальних закладів. Проаналізовано існуючі рішення (веб портали) та на основі цього аналізу запропоновано методи об'єднання освітніх ресурсів у єдине освітнє середовище.

This article represents the concepr of development and implementation of a unified approach to the integration of existing and newly created resources of learning materials in a single educational environment. Defined universal mechanism for representation of any kind of educational material to be able to convert resources from one e-learning system to another to share accumulated heritage professionals from different educational institutions. The existing solutions (web portals) and based on the analysis methods of combining educational resources in a single learning environment.

Вступ

Метою статті є розробка концепції створення єдиного підходу до інтеграції існуючих і новостворюваних навчальних ресурсів в єдине освітнє середовище. Визначено універсальний механізм представлення навчального матеріалу будь-якого типу для можливості конвертації ресурсів з однієї системи дистанційного навчання у іншу з метою спільного використання напрацьованого надбання спеціалістів різних навчальних закладів. Проаналізовано існуючі рішення (веб портали) та на основі цього аналізу запропоновано методи об'єднання освітніх ресурсів у єдине освітнє середовище.

Аналіз можливостей програмних технологій і апаратних засобів для організації віддаленого доступу до інформаційних ресурсів навчального призначення і керування навчальним процесом показує, що сучасний навчальний процес забезпечується інформаційними технологіями на рівні комп'ютерних систем підтримки дистанційного навчання з інтернет-доступом до освітніх ресурсів. Такі комп'ютерні системи призначені для використання, зберігання і відображення освітніх ресурсів, організації Інтернет або Інтранет доступу до них, керування навчальним процесом і, здебільшого, є складними спеціалізованими програмно-апаратними комплексами, які отримали назву веб-порталів дистанційного навчання. Зараз існує низка систем [1], які використовуються різними навчальними закладами України та функціонують на базі різних систем дистанційного навчання (ДН). Проте проблемою є відсутність єдиного механізму взаємодії між цими системами та сумісність зі світовими стандартами в області освіти, такими як SCORM [2], IMS [3], IEEE [4]. Тому наявні освітні ресурси не можуть бути використані поза межами системи, засобами якої вони були створені. Звідси наявність великої кількості схожих ресурсів та неможливість їх зіставлення, модифікації, виявлення переваг та недоліків, використання різними навчальними закладами.

Таким чином винікає актуальне завдання розробки засобів конвертації навчальних ресурсів до єдиного формату з метою їх подальшого використання усіма освітянами.

Передумови створення єдиного освітнього веб-середовища

Дистанційне навчання покликане забезпечити досягнення сучасних вимог до навчального процесу, втілити принципи відкритості навчання та навчання протягом всього життя. Сучасні освітні тенденції вимагають великої гнучкості та динамічності в організації навчального процесу. Розвиток економіки і галузей народного господарства створює потребу в професійних кадрах нових спеціально-

стей. Навіть класичні спеціальності вимагають модернізації процесу навчання в умовах стрімкого розвитку науки і техніки. Таким чином виникає потреба в створенні систем навчання нового покоління, характерними рисами яких стає: орієнтація на індивідуальні особливості людини, що навчається; гнучкість; відкритість для модифікації і розширення; простота підготовки вихідного матеріалу.

Для підтримки навчання з використанням інформаційних технологій застосовуються різні архітектури програмно-технічного забезпечення і методологічні підходи. Важливість методів та засобів підтримки навчання у вказаному аспекті стрімко зростає у зв'язку із все більшим застосуванням різних ресурсів і сервісів е-навчання. Тому, концепція розробки ефективної програмної архітектури підтримки навчання стають досить актуальними.

Сучасний європейський досвід підтримки навчання з використанням інформаційних технологій орієнтований на створення та широке використання парадигми веб-освітнього простору. Під поняттям веб-освітнього середовища розуміється створене за рахунок сучасних інформаційних технологій спеціальне віртуальне середовище, яке поєднує у собі функції організації дистанційного керування навчанням і забезпечення веб-доступу до різних захищених освітніх веб-ресурсів, за рахунок чого створюються передумови для підвищення якості навчання. Такий доступ реалізується через Інтернет або локальну комп'ютерну мережу.

У веб-освітньому середовищі повинні бути визначені та обумовлені:

– методологія організації навчальних ресурсів;
– методологія керування навчальними ресурсами;
– методологія навчання;
– інструментарій створення навчальних ресурсів;
– освітні стандарти;
– освітні сервіси і режими.

Основною метою веб-освітнього середовища є підвищення якості навчання за рахунок використання досягнень сучасних інформаційних технологій, організації віддаленого керування навчанням і забезпечення веб-доступу до різних захищених освітніх веб-ресурсів.

Головне завдання, що має бути розв'язане створенням єдиного освітнього веб-середовища, – забезпечення інтеграції різних, в тому числі й розподілених, інформаційних ресурсів для їхнього несуперечливого використання різними користувачами через уніфікований, бажано – єдиний, інтерфейс. Для цього кожен з ресурсів повинен формуватися з урахуванням вимог освітніх стандартів, зокрема. стандарту SCORM, який визначає інфраструктуру веб-середовища з урахуванням сучасних технологій організації єдиного доступу до захищених освітніх ресурсів, наприклад, технології Shibboleth [5].

Необхідність створення й використання освітнього веб-простору викликана наступними тенденціями:

– створення навчальних ресурсів на основі банків готових до використання сертифікованих і авторизованих об'єктів навчання;
– використання єдиного авторизованого доступу до всіх компонентів освітнього простору;
– використання єдиних освітніх сервісів і режимів;
– використання єдиного авторського інструментарію зі створення, редагування і керування освітніми ресурсами;
– використання єдиних уніфікованих освітніх стандартів і рекомендацій.

У загальному вигляді структуру системи підтримки освітнього веб-середовища можна представити у вигляді взаємодії інформаційно-керуючого ядра системи з двома підсистемами керування ресурсами та веб-порталом, також які взаємодіють з користувачами (рис. 1).

Інформаційно-керуюче ядро є ключовим елементом, що забезпечує системну функціональність, і, насамперед, підтримку інтегрування розподілених електронних ресурсів у єдине освітнє веб-середовище. При взаємодії з користувачем вважається, що будь-який інформаційний компонент не тільки надає системі свій блок керування ресурсом, але й забезпечує поповнення функцій інформаційно-керуючого ядра за рахунок сервісу, який він підключає до нього.

Інформаційно-керуюче ядро здійснює взаємозв’язок з підсистемою керування ресурсами, веб-порталом і користувачами через API (прикладні програмні інтерфейси).

Рис. 1. Структура системи підтримки освітнього веб-середовища

Для забезпечення роботи користувачів у системі підтримки освітнього веб-середовища передбачене виконання звичайних функцій інформаційної системи: електронної пошти, служби новин, електронного документообігу, тематичних форумів, сховищ адміністративно-організаційних даних тощо, які покладено на підсистему веб-порталу, а також функції забезпечення міжкомпонентної взаємодії.

Веб-портал – це самостійний структурний елемент системи (рис. 2), що забезпечує інтегрований доступу до наявних і створюваних електронних ресурсів за допомогою уніфікованого веб-інтерфейсу.

Веб-портал також має забезпечувати інформаційну взаємодію системи підтримки освітнього веб-середовища із освітніми закладами, установами та організаціями, а також з іншими інформаційними системами, яка базується на використанні сучасних інформаційних технологій.

СКБД порталу керує загальним сховищем даних інформаційних ресурсів активного користування користувачів з профілями прав доступу до них.

Підсистема керування ресурсами забезпечує інформаційне наповнення системи підтримки освітнього веб-середовища, яке складається із сукупності включених у систему цифрових навчальних ресурсів, а також відповідних функцій, що забезпечують маніпулювання ними (збереження, читання, редагування, копіювання, видалення), їх упорядкування, класифікацію, пошук, їх спільне та повторне використання та взаємодію користувачів з навчальними ресурсами системи. Структуру підсистеми представлено на рис. 3.

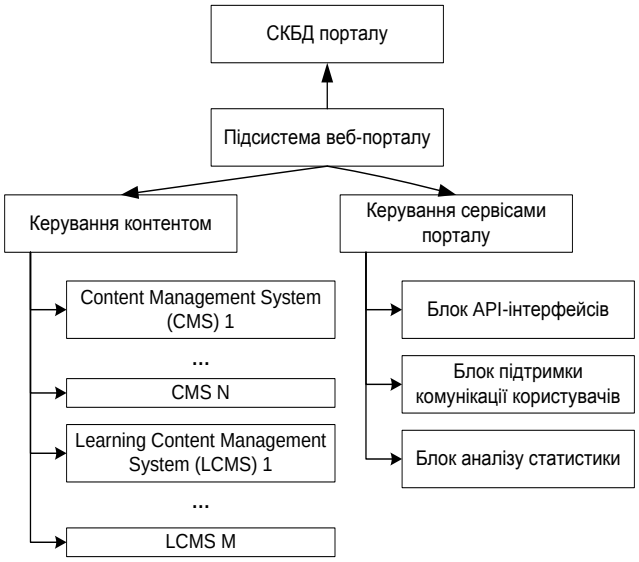


Рис. 2. Структура підсистеми веб-порталу

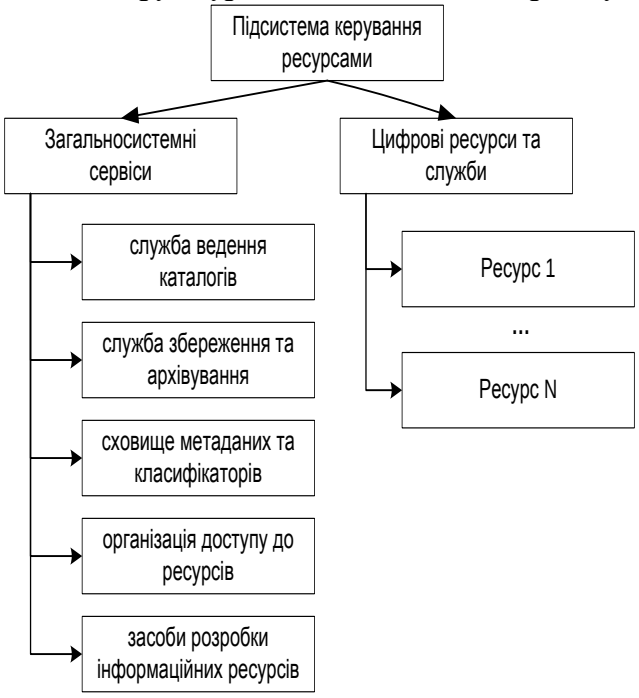


Рис. 3. Структура підсистеми керування ресурсами

Основні функції підсистеми керування ресурсами:

– формування складу цифрових ресурсів і служб;

– надання доступу до цифрових ресурсів і служб;

– забезпечення захисту цифрових ресурсів і служб;

– ведення й підтримка в актуальному стані метаданих і класифікаторів системи;

– пошук інформаційних ресурсів за збереженою метаінформацією;

– реєстрація, структурування, кодифікація та інтеграція навчальних цифрових ресурсів різної тематики і галузей знань;

– обслуговування запитів на оброблення ресурсів та інші форми керування ними.

Кожен ресурс підсистеми може містити освітню, науково-методичну та організаційно-управлінську інформацію. Із загальносистемних компонентів підсистеми можна виділити:

– службу ведення каталогів;

– службу збереження та архівування;

– сховище метаданих та класифікаторів;

– організацію доступу до ресурсів;

– засоби розробки інформаційних ресурсів.

На службу ведення каталогів покладене виконання наступних функцій:

– формування складу (додавання, виключення, типізація, систематизація і кодування) цифрових ресурсів і сервісів підсистеми керування ресурсами;

– збирання та підтримка в актуальному стані даних про сервіси та служби підсистеми;

– збирання та підтримка в актуальному стані метаданих цифрових ресурсів і класифікаторів підсистеми керування ресурсами;

– надання даних про зареєстровані цифрові ресурси.

Сховище метаданих і класифікаторів обслуговує потреби зі зберігання даних зазначених вище служб і, відповідно, теж є одним із цифрових ресурсів системи. До функцій цього ресурсу відносяться зберігання і надання метаданих, що збираються службою ведення каталогів, ведення рубрика торів, класифікаторів і тезаурусів ресурсів, ключових слів і індексів, використовуваних службою пошуку. Цей ресурс є об'єктним сховищем даних, яке забезпечує можливості гнучкого розширення складу збереженої інформації та ефективного масштабування системи у разі розширенні її складу.

Комунікація та обмін повідомленнями між веб-порталом, підсистемою керування ресурсами, ядром і користувачами здійснюється низкою компонент, що об’єднуються у механізм зв’язку, що включає наступні модулі (рис. 4):

Керуюче ядро складається з таких сервісів:

– забезпечення інформаційної безпеки;

– авторизація користувачів і сертифікація електронних підписів;

– підтримка підсистем порталу;

– керування функціональністю порталу;

– керування системними задачами керуючого ядра;

– підтримки динамічного інтерфейсу порталу;

– підтримка системних баз даних.

Модуль логістики надає наступні сервіси:

– документального супроводу надання інформаційних послуг;

– супроводу фінансових розрахунків з клієнтами.

Модуль взаємодії з користувачами надає наступні підтримки сервісів:

– комунікаційних взаємодій користувачів, відкладених у часі;

– інтерактивної взаємодії користувачів з порталом у реальному часі.

Модуль загального керування надає наступні сервіси:

– первинної реєстрації користувачів і персоналізації прав доступу;

– персоналізованого керування користувачами;

– керування наданням інформаційних послуг;

– загального супроводу навчального процесу.

Модуль роботи з інформаційними ресурсами надає наступні сервіси:

– контролю використання контенту;

– контролю використання контенту із зовнішніх джерел;

– експертного оцінювання та контролю якості контенту.

Модуль технологічного аналізу та рейтингування надає наступні сервіси:

– аналізу використання інформаційних ресурсів;

– аналізу ефективності навчального процесу;

– визначення рейтингу інформаційних ресурсів.

Підсистема моніторингу надає наступні сервіси накопичення статистики :

– використання інформаційних ресурсів;

– проходження навчального процесу.

– активносте користувачів.

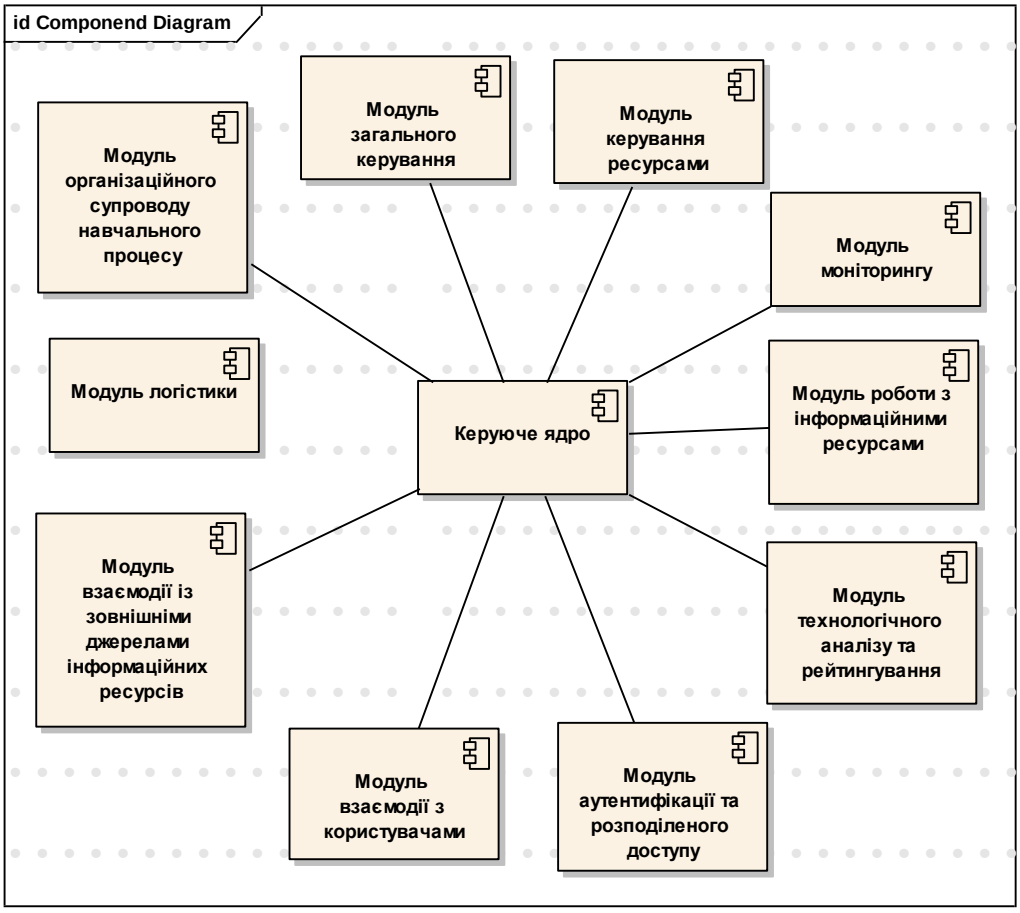


Рис. 4. Сервіси та модулі взаємозв'язку компонент єдиного веб-середовища

Класифікація та інтеграція цифрових ресурсів веб-середовища

Узагальнена категоризація цифрових ресурсів веб-середовища за ознакою їх призначення наступна:

1. Освітні інформаційні ресурси.
2. Науково-методичні інформаційні ресурси.
3. Організаційно-управлінські інформаційні ресурси.
4. Програмні ресурси.
5. Метадані та класифікатори.
6. Довідники різного призначення.

Формування й підтримка в актуальному стані всіх цифрових ресурсів освітнього веб-середовища та їхня класифікація є одною з головних функцій інформаційно-керуючого ядра системи, що забезпечуються протягом усього життєвого циклу.

До складу вже існуючих ресурсів, входять:

1. Освітні інформаційні ресурси та їх компоненти:

– електронні підручники;
– курси дистанційного навчання;
– електронні лабораторні роботи та практикуми;
– імітаційні моделі та системи моделювання різних процесів;
– тестові завдання та блоки тестів за дисциплінами;
– курсові та дипломні проекти;
– електронні конспекти лекцій;
– портфоліо учасників навчального процесу;
– чати, блоги та інша електронна переписка учасників навчального процесу;
– електронні книги;
– наукові, науково-популярні, просвітницькі статті та інші публікації, що стосуються навчання;
– електронні каталоги бібліотек;
– інформація про освітні та наукові проекти, конкурси й гранти;
– інформація про бази даних освітньо-наукового та наукового спрямування;
– посилання на освітні, наукові та інші ресурси, інформація яких не входить до складу ресурсів освітнього веб-середовища;
– інші авторизовані, сертифіковані об'єкти навчання.

2. Науково-методичні інформаційні ресурси:

– методичні рекомендації до електронних підручників, дистанційних курсів та окремих розділів дисциплін, що вивчаються;

– навчальні плани, навчальні робочі плани та програми дисциплін, за якими може бути організоване електронне навчання з використанням інформаційних ресурсів; що є у освітньому веб-середовищі;

– статті, реферати дисертацій та дисертації, а також публікації методичного та науково-методичного характеру.

3. Організаційно-управлінські інформа-ційні ресурси:

– інформація про структуру освітнього веб-середовища, його підрозділи, адресно-довідкові та контактні відомості;

– інформація про структуру кожного з учасників освітнього веб-середовища, його підрозділи, адресно-довідкові та контактні відомості;

– нормативно-правові документи, що регламентують діяльність освітнього веб-середовища, а також його учасників – юридичних осіб у частині, що стосується безпосередньої діяльності кожного учасника в межах освітнього веб-середовища;

– кадрова інформація про професорсько-викладацький склад учасників освітнього веб-середовища – юридичних осіб, а також окремих авторів електронних підручників, дистанційних курсів та інших інформаційних ресурсів навчального призначення, що розміщені у освітньому веб-середовищі, які є готовими стати учасниками навчального процесу в електронній формі у межах веб-середовища;

– фінансові документи, що стосуються діяльності освітнього веб-середовища;

– господарсько-адміністративні документи й переписка учасників, яка становить внутрішній документообіг освітнього веб-середовища.

4. Програмні ресурси:

– програмне забезпечення (ПЗ) обчислювальних комплексів установ – учасників освітнього веб-середовища;

– програмне забезпечення підтримки технології єдиного доступу до освітніх ресурсів;

– програмне забезпечення підтримки навчального процесу в освітньому веб-середовищі;

5. Метадані та класифікатори:

– структуровані метадані на всі інформаційні ресурси, що є у складі освітнього веб-середовища, які уособлюють характеристики зазначених сутностей для цілей їх ідентифікації, пошуку, оцінювання та керування ними;

– класифікатори галузей знань;

– класифікатори інформаційних ресурсів;

– класифікатори навчальних дисциплін;

– класифікатори професій та спеціальностей;

– класифікатори тестів і тестових завдань.

6. Довідники різного призначення.

Формально цифрові ресурси з будь-якої категорії потрібно розрізняти також стосовно інформаційного потоку, до складу якого вони включаються. У створюваній системі керування ресурсами таких інтегральних потоків два: «освітньо-науковий» та «адміністративний» (або «управлінський»).

З точки зору автоматизації управління зазначеними потоками, систему керування ресурсами можна розглядати як корпоративну організацію, у якій визначені правила формування електронних форм освітньо-наукової та управлінської інформації, регламент її зберігання, обробки, використання усередині корпорації й подання у «зовнішній світ».

Ззовні автоматизація системи керування ресурсами нагадує рішення подібних завдань у промислових, фінансових або державних корпораціях. Однак специфіка освіти як об'єкта автоматизації полягає в тому, що значну частину корпоративного потоку даних становить освітньо-наукова інформація, що надходить разом з «чисто» управлінськими даними. Із цього погляду система керування ресурсами ближче до інформаційних систем фінансових або державних корпорацій, ніж промислових (основна продукція яких не може бути представлена в електронній формі).

В той же час, між цифровими ресурсами системи керування ресурсами та фінансових або державних установ існують принципові відмінності. Першою з них є те, що основна частина даних фінансових або державних установ є літерною або вербальною інформацією. Що стосується освітньо-наукова інформація, яка складає інформаційну основу навчальних ресурсів, є не тільки літерною та вербальною, а й невербальною (математичні й структурні хімічні формули, таблиці, схеми, креслення, малюнки, карти, аудіо й відео об'єкти тощо). Друга відмінність полягає у тому, що більша частина освітньо-наукової

інформації є, як правило, слабо структурованою або неструктурованою. Крім того, необхідно враховувати, що у галузі освіти межа розділу інформації на управлінську і освітньо-наукову досить умовна, оскільки дані цих потоків достатньо перекриваються і, отже, є корельованими.

Таким чином існує необхідність у розробці та впровадженні в освітніх установах і закладах єдиного підходу до інтеграції існуючих і новостворюваних ресурсів в єдине освітнє середовище. При цьому забезпечиться можливість використання всіма суб'єктами освіти уже наявних цифрових ресурсів, а новостворювані ресурси, що у подальшому включатимуться до складу єдиного середовища і поступово замінюватимуть старі, будуть забезпечувати такий ступінь інтеграції та керування ними, який даватиме змогу максимально підвищити ефективність їх масового використання у навчальному процесі. Це дозволить підняти навчальний процес на рівень, який відповідатиме новим реаліям інформаційного суспільства.

Уніфікація структури веб-ресурсів

Найбільш поширений спосіб створення системи дистанційного навчання довгий час полягав в тому, щоб перевести навчальні матеріали в HTML-формат і розмістити їх на сайтах навчальних закладів. Наразі всі розробники згодні з тим, що одного тільки доступу до навчального матеріалу через Інтернет не достатньо для створення повноцінної навчальної системи. Очевидно, що навчання передбачає не просто читання навчального матеріалу, але також активне його осмислення і використання отриманих знань на практиці. Важливою метою є можливість використання навчальних ресурсів різних розробників, різних навчальних закладів.

Однак на сьогоднішній день вже сотні освітніх навчальних закладів різного рівня акредитації (ВНЗ, ПТУ, ЗНЗ, окремі підрозділи університетів, інститутів, лабораторій тощо) з дуже широким географічним розподілом мають свої електронні засоби навчання. Розроблено ряд електронних підручників і курсів ДН, створення яких спиралося або на вимоги Міністерства Освіти, або на вимоги приватного замовника. Проте на сьогодні відсутній стандарт, який би описував механізм створення єдиного формалізованого формату електронних навчальних засобів. Вимоги міністерства до даного виду ресурсів припускають наявність певного функціонального набору та набору форматів даних, які можуть бути використані при створенні електронних ресурсів, але не описані технології взаємодії модулів, формати імпорту та експорту даних. Це, в свою чергу, призводить до того, що спостерігається певна надмірність курсів, які неможливо порівняти між собою як в цілому, так і частинами. Приблизно така ж ситуація спостерігається на ринку систем дистанційного навчання. Тому при створенні єдиного освітнього вебе-простору стратегічно вірним є напрямок об'єднання існуючих ресурсів.

Будь-який електронний навчальний засіб будь-якого розробника включає наступні елементи, які зображені на рис. 5.

Рис. 5. Механізм побудови електронного навчального засобу

У більшості систем зміст представлений структурою, яку відносно легко можна отримати. Що стосується контрольних запитань і завдань, то вони і відповіді на них зберігаються в окремому місці, отже, їх можна отримати та зробити доступ до цих ресурсів, і сформувати базу завдань, що дефрагментуються по дисциплінах.

Тестові завдання різних типів для поточного, тематичного та підсумкового контролю, що передбачають простий вибір, множинний вибір, введення тексту, впорядкування, вставку графічних об’єктів в запитаннях та варіантах відповіді та ін..можуть бути структуровані за типами. Текст здебільшого представлено у форматах *.htm, *.rtf. *.txt, *.doc. Із цих чотирьох форматів два безпроблемно відображаються у будь-якому браузері, а два інших конвертуються за допомогою стандартного програмного забезпечення. Таким чином, усі тексти, що знаходяться в системах ДН та електронних підручниках, за розумних трудовитрат можуть відображатися в Інтернет браузерах. Причому навчальний матеріал

має бути розподілений на розділи, параграфи, уроки з окремих тем навчальної програми.

Малюнки представляються у форматах *.jpg, *.png, *.bmp, *.gif, *.tif; анімація – у форматах *.swf, *.dcr; відео – *.mpg, *.avi, *.wmp, *.asf; звукоряди – *.wav, *.wma, *.asf, *.mp3, *.mid. Усі ці формати даних за розумних трудовитрат можуть відображатися в Інтернет браузерах.

Імпорт, експорт створеного уроку (уроків) або певного медіа об'єкту. В усіх засобах навчання є API для імпорту та експорту файлів. Тому після обробки дані можуть бути експортовані назад до своєї початкової середи розробки, що є важливим, оскільки ми отримуємо можливість одночасно покращувати зміст і не нести додаткових витрат на придбання, модернізацію ПЗ та навчання персоналу.

Для об'єднання та спільного використання навчальних ресурсів, що були створені на базі різних платформ ДН, потрібно дефрагментувати існуючий матеріал, спираючись на певні загальні принципи. Найбільш зручно ці принципи можна представити у вигляді XML-формату. Процес конвертації відбувається наступним чином (рис. 6). Спочатку система аналізує зміст дистанційного курсу, отримує посилання на безпосередньо матеріали. Наступним кроком є завантаження матеріалів до системи.

Потім аналізується можливість відображення даних у браузерах. Якщо дані не відображаються, то пропонується два шляхи. Перший – система автоматичної конвертації, побудована на базі модуля «Публікатор» системи дистанційного навчання «Віртуальний Університет» [6], який дозволяє в автоматичному режимі переводити дані з форматів Microsoft Word, Microsoft Excel в формат *.html з встроюванням зображень, гіперпосилань та інших об'єктів, що знаходяться у конвертованих файлах. Другий шлях – якщо файл неможливо автоматично конвертувати – видаються відповідні методичні рекомендації для конвертування вручну. Наприклад, формат *.ppt програми PowerPoint можна перевести до форматів представлення, що відображатимуться браузерами, кількома методами, зокрема, зберегти сторінки презентації як зображення, або за допомогою буферу обміну перенести інформацію із цих сторінок у Microsoft Word, а потім зберегти вже у форматі *.doc. Як показує практика, це найбільш універсальна рекомендація. Виключення становлять тільки медіа-файли та файли, що потребують наявності спеціального програмного забезпечення.

Рис. 6. Технологія конвертації ресурсів до єдиного формату

Результатом обробки матеріалів буде таблиця відповідностей ресурсів (табл. 1).

Табл. 1. Форма представлення навчальних матеріалів

Назва файлу	**Тип**	**Сумісність з браузерами**	**Розташування у ієрархічній структурі**	**Фактичний опис**
Text1.txt	Текст	Так	1.1	Математичний аналіз/ інтеграли
Pic2.psd	Рисунок	Ні	1.1	Математичний аналіз/ матриці

Згідно з цією таблицею відповідностей можна визначити, які файли містить той чи інший дистанційний курс.

Після відповідних конвертацій даних система надає можливості для формування окремих занять і частин або повних дистанційних курсів.

Формування занять відбувається наступним чином. Використовуються існуючі логічні одиниці курсу (тема, розділ, підрозділ), файли, що стосуються цих одиниць, допоміжні коментарі та зв'язки між цими елементами. Тонке налаштування дозволяє використовувати матеріали на рівні власне даних (використовувати не структурну одиницю загалом, а її частину, поняття, визначення, медіа дані або зображення з цієї одиниці). Для того, щоб інформація відображалася коректно, використовується Конструктор занять.

Конструктор занять слугує для гнучкого налаштування представлення даних, дозволяє розташовувати навчальний матеріал у будь-якому порядку, з додаванням власних коментарів, висновків, додаткової інформації. Простота роботи з конструктором занять дозволяє ці дії виконувати спеціалістові своєї галузі, що не має навіть базових знать мови html. Розглянемо найбільш простий приклад (див. табл. 2).

Табл. 2. Приклад побудови заняття

<Data1 Категорія порівняння>	<Data2 Чисельні методи>	<Data3 Імітаційне моделювання>
<Data4 Загальне порівняння >	<lesson name="Переваги та недоліки чисельних методів" subdir="lesson1">	<lesson name="Переваги та недоліки імітаційного моделювання" subdir="lesson2">
<Data5 Точність результатів>	<lesson name=" Точність результатів чисельних методів" subdir="lesson3">	<lesson name="Точність результатів імітаційного моделювання" subdir="lesson4">

Тобто, при представленні даних у вигляді таблиці можна у будь-яку комірку вставити як опис, коментар або назву, так і власне навчальний матеріал. Також існує можливість модифікувати параметри тегу <Data> для зміни зовнішнього вигляду елементу заняття:

itmFont = "Arial" - шрифт
itmFontSize = "16" розмір
itmFontColor = "000000" колір
itmFontBold = "0" напівжирний
itmFontItalic = "0" курсив
itmFontUnderline = "0" підкреслений.

Також задаються інші параметри. Параметри кожного тегу обираються зі списку Тобто заповнення цієї таблиці також не потребує від користувача будь-яких знань з області програмування. Ці параметри та їх кількість можна змінювати у шаблоні засобами Microsoft Word, а також після завантаження створеного заняття або курсу в конкретну систему ДН. Оскільки шаблон передбачає багаторазове використання, як правило, створюється бібліотека шаблонів (порівняння, фільтри, вибірки та ін.).

При формуванні занять використовується форма представлення матеріалів. У користувача є можливість використання тільки частини даних. За результатами конвертації користувача отримує *.zip-архів, що містить необхідну частину дистанційного курсу, тестові запитання та відповіді на них. Існує окремий режим, коли дані курсу та запитання з відповідями конвертуються окремо і пов'язуються між собою тільки спеціальними посиланнями. Такий формат набагато простіше.

Дослідження показали, що привести будь-які матеріали до загального XML-формату для розробників не є складною задачею. А завдяки можливостям імпорту або експорту розробник, користуючись загальним XML-форматом, зможе конвертувати будь-які матеріали у свою систему підтримки ДН.

У результаті дослідження розроблено універсальний XML-формат для будь-яких типів навчальних матеріалів.

Висновки

Аналіз існуючих веб-порталів, та їх структур показав що усі вони мають спільні риси та правила функціонування. Спираючись на це запропонована концепція об'єднання усіх навчальних матеріалів у рамках єдиного формату. На прикладі навчальних матеріалів розглянута структура та методологія приведення навчальних матеріалів до такого формату. Подана концепція дозволила вирішити проблему інтеграції порталів на рівні електронних підручників та начальних матеріалів. Розроблена концепція реалізована на практиці в системі ДН «Віртуальний університет» і використовується в веб-порталах www.rozumniki.ua. та www.testportal.org.ua. Це дозволило поєднати в єдиний формат навчальні матеріали з різних систем дистанційного навчання різних розробників навчальних курсів і з'явилась можливість переносу із фо-

рмату в формат навчальних курсів провідних виробників програмного забезпечення та начальних матеріалів.

Список літератури

1. Томашевський В.М. Огляд сучасного стану систем дистанційного навчання / В.М. Томашевський, Ю.Л. Новіков, П. А. Камінська / Наукові праці: Науково-методичний журнал. Т. 135. Вип. 122. Комп'ютерні технології. Миколаїв: Вид. ЧДУ ім. Петра Могили. – 2011
2. The Sharable Content Object Reference Model (SCORM) [Електронний ресурс] // Режим доступу: http://www.adlnet.gov/
3. IMS Global Learning Consortium [Електронний ресурс] // Режим доступу: http://www.imsglobal.org
4. IEEE Standard for Learning Technology-Learning Technology Systems Architecture (LTSA) [Електронний ресурс] // Режим доступу: http://www.ieeeltsc.org
5. Офіційний веб-сайт Shibboleth [Електронний ресурс] // Режим доступу: http://shibboleth.internet2.edu/
6. Про систему дистанційного навчання "Віртуальний Університет" [Електронний ресурс] // Режим доступу: http://vu.net.ua/

УДК 681.3.053

СЕЛИВАНОВ В.Л.,
ВОРОБЙОВ В.В.

ОПТИМАЛЬНЫЕ СТРУКТУРЫ ДЕКОДИРУЮЩИХ СЕТОК ПРЕОБРАЗОВАТЕЛЕЙ КОД-НАПРЯЖЕНИЕ ДВОИЧНО-ДЕСЯТИЧНЫХ СИСТЕМ СЧИСЛЕНИЯ

Показываются основные принципы построения декодирующих сеток (ДС) для преобразователей код-напряжение (ПКН) двоично-десятичных систем счисления (ДДСС). Предложены оптимальные структуры декодирующих сеток для ПКН ДДСС.

The subject of the article is about basic principles of creating scale transform circuits in voltage D/A conversion (DAC) schemes for the binary-coded decimal numbers (BCD). Optimal transform circuits for BCD DAC was also given.

1. Введение

Преобразователи код-напряжение (ПКН) широко используются в цифровой измерительной аппаратуре (в аналого-цифровых преобразователях АЦП последовательного счёта и поразрядного уравновешивания), для формирования управляющих сигналов в системах автоматического управления и регулирования, а также в цифро-аналоговых вычислительных комплексах (гибридных вычислительных системах).

Декодирующие сетки (ДС) – это аналоговые сумматоры напряжений (параллельного типа или комбинированные), у которых коэффициенты передач $k_i\ (i = \overline{1,n})$ пропорциональны весам разрядов декодируемого числа $g_i\ (i = \overline{1,n})$.

Рассматривая ДС для ДДСС необходимо рассмотреть особенности таких систем счисления.

Предположим, что m – количество десятичных разрядов (количество тетрад), соответственно $n = 4m$ – количество двоичных разрядов.

В настоящий момент в основном используются всего две структуры ДС для ДДСС.

2. Существующие варианты ДС ДДСС

В цифровой части вычислительной системы при использовании двоично-десятичных кодов наиболее часто используют коды ДДСС 8421.

Рассмотрим пример построения схемы ДС для ПКН ДДСС 8421 при $n = 8, m = 2$.

Для такой системы счисления веса соответствующих разрядов ДДСС 8421 равны $g_1 = 0{,}8$, $g_2 = 0{,}4$, $g_3 = 0{,}2$ и $g_4 = 0{,}1$, $g_5 = 0{,}08$, $g_6 = 0{,}04$, $g_7 = 0{,}02$, $g_8 = 0{,}01$.

Для данных значений весов разрядов получаются следующие значения резисторов параллельной ДС: $R_1 = R$; $R_2 = 2R$; $R_3 = 4R$; $R_4 = 8R$; $R_5 = 10R$; $R_6 = 20R$; $R_7 = 40R$; $R_8 = 80R$.

Паралельная структура такой ДС [1,2,3,4,5] приведена на рис. 1а.

Значение R_x, при котором не будут изменяться эквивалентные сопротивления R_{eqs+1} при любом количестве тетрад определяется с учётом того, что R_{eq5} должно быть равно R_x, $R_{eq9} = 10\,R_x$, $R_{eq13} = 100\,R_x$ и т.д, где R_{eqs+1} - эквивалентное сопротивление части схемы, начиная с R_{s+1} и заканчивая $R_{n,n+1}$ (R_{89} для $n = 8$).

Для части схемы рис. 1а, которая обведена пунктиром, эквивалентное сопротивление R_{eq5}, которое должно быть равно R_x, определяется из следующего соотношения:

$$\frac{1}{R_{eq5}} = \frac{1}{R_x} = \frac{1}{10R} + \frac{1}{20R} + \frac{1}{40R} + \frac{1}{80R} + \frac{1}{10R_x}. \qquad (1)$$

Таким образом $R_{eq5} = R_x = 4{,}8R$, а $R_9 = 48R$.

Отметим, что в такой ДС большое количество разных номиналов резисторов

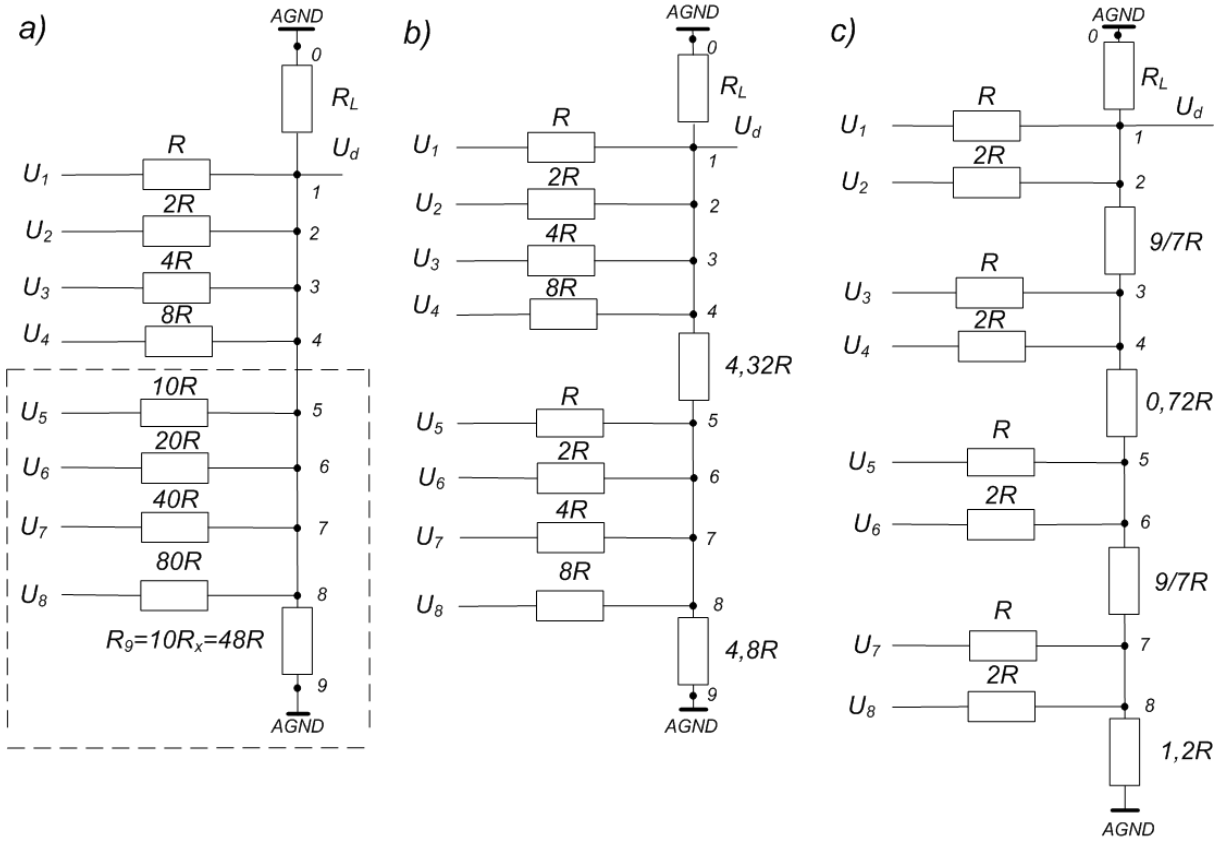


Рис. 1. Декодирующие сетки ПКН ДДСС 8421

($n+1$ без учета R_L), большой разброс номиналов резисторов ($R_{max} / R_{min} = 8 * 10^{m-1}$) и общее количество резисторов $n+2$.

Для уменьшения количества номиналов резисторов и уменьшения их разброса используют эквивалентные преобразования структуры, т.е. переходят к комбинированным структурам сумматора напряжений. Для этого определяют значения $R_{eq,s+1}(s=\overline{1,n-1})$, выбирают значения $C_{s+1}(s=\overline{1,n-1})$, где C_{s+1} – это некоторый постоянный коэффициент, показывающий во сколько раз необходимо уменьшить сопротивления всех резисторов ниже s-й точки, и рассчитывают значения сопротивлений резисторов связи $R_{s,s+1}(s=\overline{1,n-1})$, включенных между s-й и $s+1$-й точками, по формуле:

$$R_{s,s+1} = \frac{C_{s+1}-1}{C_{s+1}} R_{eq,s+1}(s=\overline{1,n-1}). \qquad (2)$$

При использовании такой комбинированной структуры резисторы связи включаются между тетрадами (R_{45}, R_{89} и т.д.). Так как $R_{eq5} = R_x = 4,8R$ и при $C_5 = 10$, из (2) получаем $R_{45} = 4,32R$. Схема такой комбинированной ДС ПКН ДДСС 8421 приведена на рис. 1b.

Такая ДС имеет 6 разных номиналов резисторов, разброс номиналов резисторов $R_{max} / R_{min} = 8$ (без учета R_L) и общее количество резисторов $1,25n$. Такие ДС на данный момент считаются наиболее оптимальными и наиболее часто используются [2, 6].

3. Оптимальная ДС для ПКН ДДСС 8421

Если внутри тетрад дополнительно включить резисторы связи между парами резисторов внутри тетрад (R_{23}, R_{67} и т.д.), то можно получить наиболее оптимальную схему, выбрав C_3, C_7 и т.д. равными 4, и C_5, C_9 и т.д. равными 2,5. Используя формулу (2) можно рассчитать сопротивления резисторов связи такой оптимальной комбинированной структуры. Соответствующая схема оптимальной ДС ПКН ДДСС 8421 приведена на рис. 1c.

Такая ДС содержит всего 5 разных номиналов резисторов (без учета R_L) и разброс номиналов резисторов $R_{max} / R_{min} \approx 2,78$ при общем количестве резисторов $1,5n$. Следовательно, предложенная структура ДС является наиболее оптимальной.

4. Оптимальная ДС для ПКН ДДСС 2421 (код Айкена)

Для повышения точности в ПКН ДДСС часто используют промежуточное преобразование кода ДДСС 8421 в код ДДСС 2421 (код Айкена) или в код ДДСС 3321. Рассмотрим оптимальные структуры ДС ДДСС 2421.

Схема стандартной ДС (при $n = 8, m = 2$) для декодирования кода Айкена изображена на рис. 2а. Для такой ДДСС $g_1 = 0,2$, $g_2 = 0,4$, $g_3 = 0,2$, $g_4 = 0,1$, $g_5 = 0,02$, $g_6 = 0,04$, $g_7 = 0,02$, $g_8 = 0,01$, и значения сопротивления резисторов параллельной ДС следующие: $R_1 = R$, $R_2 = 0,5R$, $R_3 = R$, $R_4 = 2R$, $R_5 = 10R$, $R_6 = 5R$, $R_7 = 10R$, $R_8 = 20R$.

Для части схемы рис. 2а, которая обведена пунктиром, эквивалентное сопротивление R_{eq5}, которое должно быть равно R_x, определяется из следующего соотношения:

$$\frac{1}{R_{eq5}} = \frac{1}{R_x} = \frac{1}{10R} + \frac{1}{5R} + \frac{1}{10R} + \frac{1}{20R} + \frac{1}{10R_x}. \qquad (3)$$

Таким образом $R_{eq5} = R_x = 2R$, а $R_9 = 20R$.

В такой ДС большое количество разных номиналов резисторов ($n+1$ без учета R_L), большой разброс номиналов резисторов ($R_{max} / R_{min} = 2 * 10^{m-1}$) и общее количество резисторов $n+2$.

При использовании известной комбинированной структуры резисторы связи включаются между тетрадами (R_{45}, R_{89} и т.д.). Так как $R_{eq5} = R_x = 2R$ и $C_5 = 10$ получаем $R_{45} = 1,8R$. Схема такой комбинированной ДС ПКН ДДСС 2421 приведена на рис. 2b.

В такой ДС 4 разных номинала резисторов, разброс номиналов резисторов $R_{max} / R_{min} = 4$ и общее количество резисторов $-1,25n$.

Если воспользоваться формулой (2) и применить эквивалентное преобразование структуры включением резисторов связи ($R_{34}, R_{45}, R_{78}, R_{89}$ и т.д.) между тетрадами и внутри тетрады (перед последним резистором каждой тетрады) и выбрать C_5, C_9 и т.д. равным 4, и C_4, C_8 и т.д. равным 2,5, то можно получить оптимальную ДС ДДСС 2421.

При этом $R_{34} = 0,75R_{eq4} = 0,75R$; $R_{45} = 0,6R_{eq_5} = 1,2R$, а с учетом $C_4 = 4$ новое значение R_{45} равно $0,3R$.

Оптимальная структура такой комбинированной ДС ПКН ДДСС 2421 изображена на рис. 2с.

В такой ДС всего три номинала (без учета R_L), при минимальном разбросе номиналов ($R_{max} / R_{min} = 2$), и количестве резисторов – $1,5n$.

5. Оптимальная ДС для ПКН ДДСС 3321

Схема стандартной ДС (при $n = 8, m = 2$) для ПКН ДДСС 3321 изображена на рис. 3а. Для такой ДДСС $g_1 = g_2 = 0,3$, $g_3 = 0,2$, $g_4 = 0,1$, $g_5 = g_6 = 0,03$, $g_6 = 0,02$, $g_7 = 0,02$, и $g_8 = 0,01$, и значения сопротивления резисторов параллельной ДС следующие $R_1 = R_2 = R$; $R_3 = 1,5R$; $R_4 = 3R$; $R_5 = R_6 = 10R$; $R_7 = 15R$; $R_8 = 30R$.

Для части схемы рис. 3а, которая обведена пунктиром, эквивалентное сопротивление R_{eq5}, определяется из следующего соотношения:

$$\frac{1}{R_{eq5}} = \frac{1}{R_x} = \frac{1}{10R} + \frac{1}{10R} + \frac{1}{15R} + \frac{1}{30R} + \frac{1}{10R_x}, \qquad (4)$$

Таким образом $R_9 = 30R$, а $R_{eq5} = R_x = 3R$.

В такой ДС большое количество разных номиналов резисторов ($n+1$ без учета R_L), большой разброс номиналов резисторов ($R_{max} / R_{min} = 4 * 10^{m-1}$) и общее количество резисторов $n+2$. При использовании комбинированной структуры резисторы связи включаются между тетрадами (R_{45}, R_{89} и т.д.). Так как $R_{eq5} = R_x = 3R$ и $C_5 = 10$ получаем $R_{45} = 2,7R$. Схема такой комбинированной ДС ПКН ДДСС 3321 приведена на рис. 3b.

В такой ДС 4 разных номинала резисторов, разброс номиналов резисторов $R_{max} / R_{min} = 4$ и общее количество резисторов $-1,25n$.

Для получения оптимальной структуры комбинированной ДС следует включить

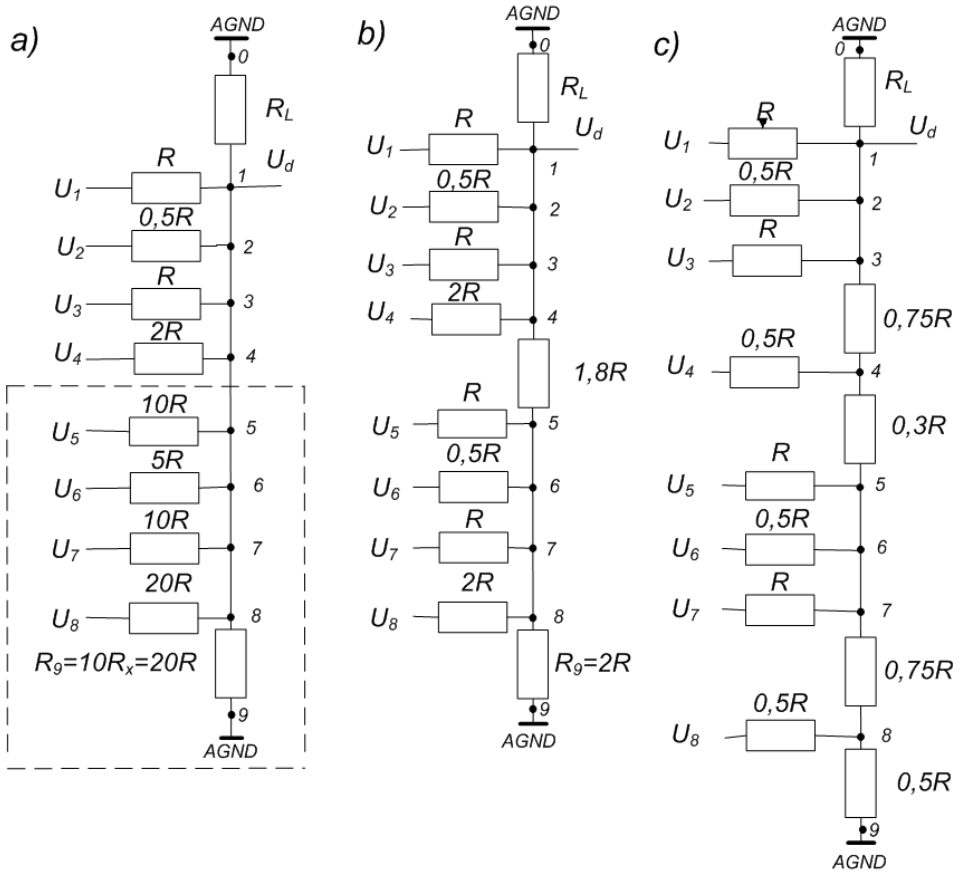


Рис. 2. Декодирующие сетки ПКН ДДСС 2421

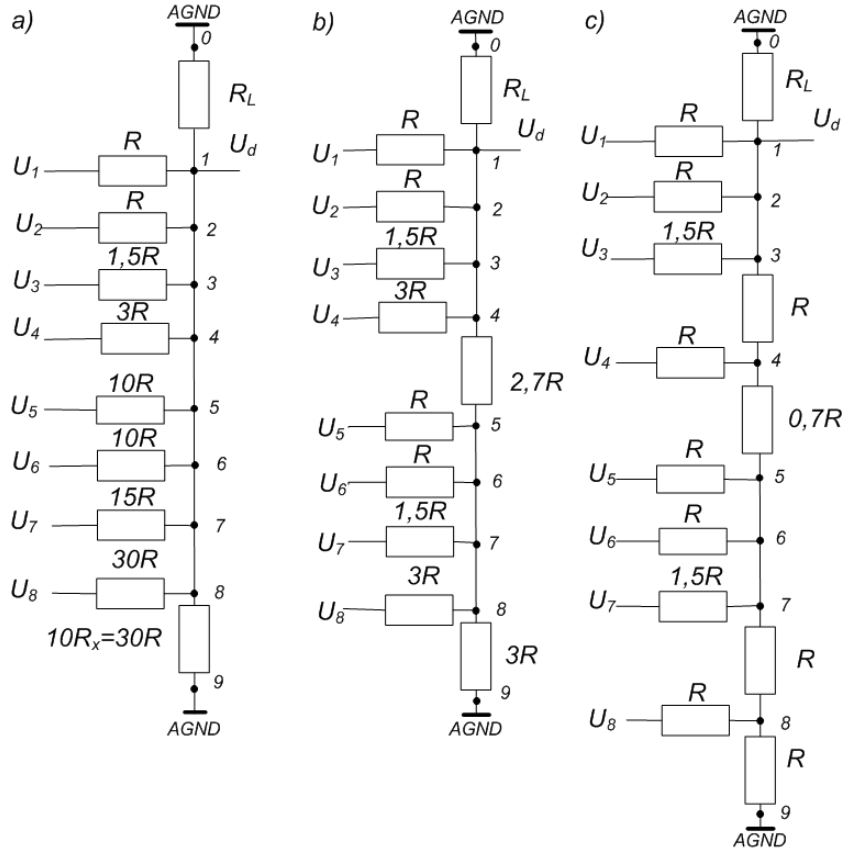


Рис. 3. Декодирующие сетки ПКН ДДСС 3321

резисторы связи между тетрадами и внутри тетрады (R_{34}, R_{45}, R_{78}, R_{89} и т.д.) и выбрать C_5, C_9 и т.д. равным $10/3$, и C_4, C_8 и т.д. равным

3. Полученная оптимальная структура ДС изображена на рис. 3с.

Эта оптимальная ДС состоит из трех номиналов ($0,7R$ - R - $1,5R$ без учета R_L) при разб-

росе номиналов в 2,14 раза и количестве резисторов – $1{,}5n$.

Выводы

Предложенные структуры ДС являются оптимальными исходя из оценочной характеристики по трём параметрам: количество разных номиналов резисторов, разброс номиналов резисторов и количество резисторов используемых в структуре.

Такие ДС также могут быть использованы в умножающих ПКТ в качестве формирователя эталонных токов – так называемая инверсная резистивная матрица ИРМ (инверсное включение сумматора напряжений) для формирования взвешенных эталонных токов.

Список литературы

1. Волович Г.И. Схемотехника аналоговых и цифро-аналоговых электронных устройств. Москва, 2005 г.
2. Гнатек Ю.Р. Справочник по цифроаналоговым и аналогоцифровым преобразователям. Москва, 1982 г.
3. Лебедев О.Н., Марцинкявичюс А.-Й.К. и др. Микросхемы памяти. ЦАП и АЦП. "КУбК-а" Москва, 1996 г.
4. Сентурия С.Д., Уэдлок Б.Д. Электронные схемы и их применения, Издательство "МИР", Москва, 1977.
5. Raj Kamal. Digital Systems Principles and Design. Pearson Education India, 2007.
6. Корн Г., Корн Т., Электронные аналоговые и аналого-цифровые вычислительные машины, пер. с англ., М., 1967.

УДК 519.85

ШОВГУН Н.В.

АНАЛІЗ КРЕДИТОСПРОМОЖНОСТІ ПОЗИЧАЛЬНИКА ЗА ДОПОМОГОЮ МЕТОДІВ З НЕЧІТКОЮ ЛОГІКОЮ

В статті зроблено аналіз найбільш широко застосованих методів оцінки кредитоспроможності позичальника. Розглянуті методи оцінки кредитоспроможності на основі нечіткої логіки. Показані результати оцінки кредитоспроможності позичальника за допомогою методів з нечіткою логікою.

This paper is a survey on the most popular methods for assessing the creditworthiness of the borrower. Also, it presented methods for evaluating creditworthiness based on fuzzy logic and obtained results of assessing the creditworthiness of the borrower using methods of fuzzy logic.

1. Вступ

Основним напрямом діяльності комерційного банку є видача кредитів. По деяким оцінкам кредитування дає майже половину прибутку банку.

Разом з тим, кредитування пов'язано з ризиком, зумовленим можливим невиконання своїх обов'язків позичальником. Відомо, що цей ризик є один з найзначніших ризиків банку.

Споживче кредитування фізичних осіб є одним з основних банківських продуктів. При цьому для мінімізації втрат банку потрібен ретельний відбір позичальників та ефективна оцінка їх кредитоспроможності. Для цього в банках існують моделі і методи прийняття рішень по кредитним заявкам.

Насамперед, для прийняття рішення має значення детальна анкета позичальника, у якій містяться багато даних – від матеріального положення позичальника до його особистих якостей. З врахуванням усіх цих даних прийняте рішення повинно мінімізувати ризик та одночасно не повинно мати наслідком необґрунтовану втрату позичальника.

Умови, у яких має бути прийняте рішення, часто характеризується неповнотою даних, їх різнорідністю та недостатністю.

Метою даної статті є дослідження існуючих підходів до визначення кредитоспроможності позичальника, аналіз їх особливостей та вибір методик, які найбільш ефективно дозволяють прийняти рішення у конкретних умовах діяльності комерційного банку. Запропоновано використовувати для аналізу кредитоспроможності фізичних осіб нечіткий контролер Мамдані та нечітку нейрону мережу TSK з механізмом нечіткого логічного висновку Сугено.

2. Аналіз відомих методів оцінки кредитоспроможності позичальника

Найчастіше для оцінки кредито-спроможності позичальника використовують:
- статистичні методи;
- дерева рішень;
- генетичні алгоритми;
- нейронні мережі.

Статистичні методи на основі дискримінантного аналізу використовуються при вирішенні задач класифікації. Так, відома модель Альтмана, що використовується для аналізу кредитоспроможності юридичних осіб, побудована на основі множинного дискримінантного аналізу. Для аналізу кредитоспроможності фізичних осіб використовують лінійну або логістичну регресії [1]. Наприклад, при використанні лінійної регресії, функція, що визначає кредитний рейтинг, апроксимується лінійною функцією щодо компонентів вектора характеристик позичальника, тобто

$$p = a_0 + a_1 \cdot x_1 + a_2 \cdot x_2 + \ldots a_N \cdot x_N,$$

де a_0 – вільний член;

a_i при $i = 1, \ldots, N$ – вагові коефіцієнти характеристик позичальника;

x_i – характеристики позичальника.

Всі регресійні методи чутливі до кореляції між характеристиками, тому в моделі не повинно бути сильно корельованих характеристик позичальника.

Статистичні методи спираються на усереднені характеристики вибірки, але при дослідженні реальних складних життєвих феноменів ці характеристики можуть не відповідати дійсності. Оцінка кредитоспроможності позичальника за допомогою цих методів потребує великої кількості історичних данних щодо кредитів,

що не завжди можливо. Також характерною є проблема недостатньої кількості прикладів позичальників, що виявилися не спроможними погасити свою заборгованість. Одне з вирішень цієї проблеми запропоновано в [2]. Суттєвим недоліком статистичних методів вважають вимоги до спеціальної підготовки користувача [3].

Метод дерев рішень відрізняється високою швидкістю обробки даних і навчання при збереженні властивостей систем нечіткого логічного висновку.

При використанні методу дерев рішень для класифікації кредитних заявок застосовується набір правил, що формується при побудові дерева на основі навчальної вибірки. Дерево включає взаємопов'язані початковий (кореневий), проміжні та кінцеві вузли Кожному з вузлів відповідає умова (правило) класифікації об'єктів. Для побудови дерева на кожному внутрішньому вузлі необхідно знайти таку умову, яка б розбивало множину, асоційовану з цим вузлом на підмножину. В якості такої перевірки повинен бути вибрана одна з характеристик. Обрана характеристика повинна розбити множину так, щоб одержані в підсумку підмножини складалися з об'єктів, що належать до одного класу, або були максимально наближеними до цього, тобто кількість об'єктів з інших класів в кожній з цих множин було якомога менше.

Але у метода дерев рішень існують суттєві недоліки. Він не підходять для задач з великим числом можливих розв’язків і умова (правило) може формулюватися тільки в термінах «більше/менше», що заважає застосуванню до задач, де класс визначається більш складним поєднанням змінних [4].

Генетичні алгоритми грунтуються на стохастичному пошуку глобального оптимума цільової функції. Ідея генетичних алгоритмів запозичена в живої природи і полягає в організації еволюційного процесу (за допомогою операцій схрещування, мутації та селекції), кінцевою ціллю якого є оптимальне рішення задачі.

Генетичні алгоритми мають ряд недоліків. Критерій відбору хромосом і сама процедура відбору евристичні і не гарантують знаходження «кращого» рішення. Також необхідно мати в наявності досить великий обсяг вхідних данних для завершення процедури селекції [5].

Під нейронними мережами розуміють обчислювальні структури, які моделюють прості біологічні процеси, що зазвичай асоціюють з процессами в людському мозку. Вони являють собою совокупність елементів (штучних нейронів), пов'язаних між собою синаптичними зв'язками [6].

Недоліком нейронних мереж називають те, що вони являють собою «чорний ящик» та відсутність твердих правил, щодо вибору швидкості навчання мережі для вирішення конкретних задач [6].

Існує багато архітектур нейронних мереж. Так наприклад, в роботі [7] для класифікації клієнтів німецького та австралійського банків використовувалися такі мережі: мережа Кохонена, мережа BackPropagation, радіально-базисна мережа, каскадна мережа.

Загальними недоліками вищеописаних методів є вимоги до об’єму вхідних даних та жорсткі вимоги до характеристик і критеріїв відбору позичальника. В реальному житті середовище позичальника постійно змінюється, як результат – змінюються його сімейний та фінансовий стан, що вносить невизначеність в інформацію щодо клієнта. Неправильна оцінка позичальника в цих умовах може призвести до збільшення ризику банку чи втрати потенційно надійних клієнтів.

Врахувати таку невизначеність можна за допомогою методів нечіткої логіки. Даний підхід дає можливість працювати як з кількісними, так і з якісними характеристиками.

3. Методи оцінки кредитоспроможності позичальника за допомогою нечіткої логіки

Задачу оцінки кредитоспроможності можна сформулювати таким чином. Кожна кредитна заявка задається вектором $\{X_1, X_2, ..., X_i, ..., X_N\}$, де X_i – певним чином формалізовані дані з анкети позичальника та параметри кредиту. Далі по заданому вектору треба прийняти рішення про надання кредиту, тобто класифікувати позичальника як «надійного», чи класифікувати як «поганого».

В статті [8] використовувався нечіткий контролер з алгоритмом нечіткого логічного висновку Мамдані для аналізу кредитоспроможності юридичних осіб. В загальному вигляді нечіткий логічний висновок має наступні етапи:

1. Визначення множини вхідних змінних:
   $X = \{X_1, X_2, ..., X_i, ..., X_M\}$;
2. Визначення множини вихідних змінних:
   $D = \{D_1, D_2, ..., D_i, ..., D_M\}$;

3. Формування базової терм-множини з відповідними функціями належності кожного терма: $A = \{a_1, a_2, ..., a_i\}$;

4. Формування кінцевої множини нечітких правил, узгоджених щодо використовуваних в них змінних;

5. Знаходження чіткого значення для кожної з вихідних лінгвістичних змінних.

Даний підхід можна використати і для розв'язання задачі оцінки кредитоспроможності фізичних осіб.

Розглянемо алгоритм нечіткого висновку Мамдані.

Нехай базу знань складають два нечітких правила:

$П_1$: якщо x є A_1 і y є B_1, то z є C_1,

$П_2$: якщо x є A_2 і y є B_2, то z є C_2,

де x і y вхідні змінні, z – вихідна змінна, A_1, A_2, B_1, B_2, C_1, C_2 – деякі задані функції належності, при цьому чітке значення z треба визначити на основі даної інформації і чітких значень x, y. Етапами алгоритму є:

1. Введення нечіткості. Знаходимо степінь істиності для передумов кожного правила: $A_1(x_0)$, $A_2(x_0)$, $B_1(x_0)$, $B_2(x_0)$.

2. Логічне виведення. Знаходимо рівні «відсікання» для передумов кожного з правил (з використанням операції мінімума):

$\alpha_1 = A_1(x_0) \cap B_1(y_0)$;

$\alpha_2 = A_2(x_0) \cap B_2(y_0)$.

Далі знаходимо «відсікання» функції належності:

$C_1' = (\alpha_1 \cap C_1(z))$;

$C_2' = (\alpha_2 \cap C_2(z))$.

3. Композиція. Знаходимо об'єднання знайдених відсічених функцій належності з використанням операції максимум, отримуємо підсумкову нечітку підмножину для змінної виходу з функцією належності:

$$\mu_\Sigma = C(z) = C_1'(z) \cup C_2'(z) = (\alpha_1 \cap C_1(z)) \cup (\alpha_2 \cap C_2(z)).$$

4. Зведення до чіткості з використанням, наприклад, центроїдного методу [8].

В алгоритмі нечіткого висновку Сугено, використовується такий набір правил [9]:

$П_1$: якщо x є A_1 і y є B_1, то $z=a_1x+ b_1y$,

$П_2$: якщо x є A_2 і y є B_2, то $z=a_2x+ b_2y$,

де x і y вхідні змінні, z – вихідна змінна, A_1, A_2, B_1, B_2, C_1, C_2 – деякі задані функції належності, a_1, a_2, b_1, b_2 – деякі числа.

Алгоритм має вигляд:

1. Введення нечіткості як в алгоритмі Мамдані.

2. Нечітке виведення. Знаходимо $\alpha_1 = A_1(x_0) \cap B_1(y_0)$, $\alpha_2 = A_2(x_0) \cap B_2(y_0)$ та індивідуальні виходи правил:

$\dot{z}_1 = a_1 x_0 + b_1 y_0$;

$\dot{z}_2 = a_2 x_0 + b_2 y_0$.

3. Визначення чіткого значення змінної виведення:

$$z_0 = \frac{\alpha_1 \dot{z}_1 + \alpha_2 \dot{z}_2}{\alpha_1 + \alpha_1}$$

Розглянемо нечітку нейронну мережу TSK (Takagi, Sugeno, Kang'a) з механізмом нечіткого логічного висновку Сугено. Правила мережі можна представити у такому вигляді [9]:

$$R_1 : якщо \quad x_1 \in A_1^{(1)}; x_2 \in A_2^{(1)}, ..., x_n \in A_n^{(1)},$$

$$то \ y_1 = p_{10} + \sum_{j=1}^{N} p_{1j} x_j;$$

$$R_M : якщо \quad x_1 \in A_1^{(M)}; x_2 \in A_2^{(M)}, ..., x_n \in A_n^{(M)},$$

$$то \ y_M = p_{M0} + \sum_{j=1}^{N} p_{Mj} x_j,$$

де $A_i^{(k)}$ – значення лінгвістичної змінної x_i для правила R_k з функцією належності

$$\mu_A^{(k)}(x_i) = \frac{1}{1 + \left(\frac{x_i - c_i^{(k)}}{\sigma_i^{(k)}} \right)^{2b_i^{(k)}}}.$$

Композиція результатів має вигляд:

$$y(x) = \frac{\sum_{k=1}^{M} w_k y_k(x)}{\sum_{k=1}^{M} w_k},$$

де $w_k = \mu_A^{(k)}(x)$ – степінь виконання умов правила, $\mu_A^{(k)}(x) = \prod_{j=1}^{N} \left[\frac{1}{1 + \left(\frac{x_j - c_j^{(k)}}{\sigma_j^{(k)}} \right)^{2b_j^{(k)}}} \right]$.

5. Результати досліджень

Для аналізу кредитоспроможності позичальника за допомогою ННМ TSK та контролера Мамдані використовувалися дані одного з вітчизняних банків по 100 кредитним заявкам за 2010 р.

У якості вхідних змінних використовувався вектор X= {Сума кредиту, Дохід, Термін кредиту, Вік, Термін проживання в квартирі, Освіта}.

Співвідношення навчальної та перевірочної вибірки було вибрано як 70 до 30. Результати досліджень зведені в таблиці 1-2. Результати оцінки кредитоспроможності позичальника за допомогою лінійної та логістичної регресії, кластерного аналізу методом нечітких к-середніх [9] та вище наведених методів представлені в таблиці 3.

Табл. 1. Результати оцінки кредитоспроможності позичальника за допомогою ННМ TSK

Кількість правил	СКО навчальної вибірки	% невірних класифікацій на навчальної вибірці	СКО перевірочної вибірки	% невірних класифікацій на перевірочній вибірці
3	0.0316	2.857	0.0178	6.66
5	0.0149	0	0.0127	3.33
7	0.0194	0	0.027	3.33

Табл. 2. Результати оцінки кредитоспроможності позичальника за допомогою контролера Мамдані

Кількість правил	СКО навчальної вибірки	% невірних класифікацій на навчальної вибірці	СКО перевірочної вибірки	% невірних класифікацій на перевірочній вибірці
3	0.08349	6.66	0.1092	10
5	0.0193	2.857	0.0305	6.66
7	0.0131	1.4286	0.0382	6.66

Табл. 3. Порівняльні результати оцінки кредитоспроможності позичальника різними методами

Метод	СКО навчальної вибірки	% невірних класифікацій на навчальної вибірці	СКО перевірочної вибірки	% невірних класифікацій на перевірочній вибірці
Лінійна регрессія	0.0987	8.57	0.04334	13.33
Кластерний аналіз		8.57		16.66
Логістична регерсія	0.0171	2.857	0.0344	10
ННМ TSK	0.0149	0	0.0127	3.33
НК Мамдані	0.0193	2.857	0.0305	6.66

Як можна бачити з аналізу таблиць, ННМ TSK та контролер Мамдані дають майже однаково задовільні рішення при кількості правил 3-7. Процент невірних класифікацій при застосуванні статистичних методів аналізу тієї самої вибірки надто високий.

6. Висновки

1.Основною проблемою в аналізі кредитоспроможності фізичної особи виступає неточність даних, недостатня база знань про минуле клієнтів та необхідність працювати з інгвістичними характеристиками, які важко піддаються математичній обробці.

2. В специфічних умовах функціонування вітчизняних банків методи з використанням нечіткої логіки, закрема методи на базі нечіткої нейронної мережі TSK та контролера Мамдані дають непогані результати, які дозволяють звести відсоток незадовільних рішень до 3…10%.

Перелік посилань

1. Руководство по кредитному скорингу / под ред. Элизабет Мэйз;– Минск: Гревцов Паблишер, 2008г. – 458с.
2. Паклин Н.Б., Уланов С.В., Царьков С.В. Построение классификаторов на несбалансированных выборках на примере кредитного скоринга // Искусственный интеллект. – 2010г. - №3 – с. 528-534

3. А.А. Ежов, С.А. Шумкий Нейрокомпьютинг и его применения в экономике и бизнесе. – М.: МИФИ, 1998.- 224 с.
4. Тегеран Т. Программируем колективный разум. – СПб: Символ-Плюс, 2008. – 368 с.
5. Рутковская Д., Пилинський М., Рутковский Л. Нейронніе сети, генетические алгоритмі и нечеткие системы.– М.: Горячая линия -Телеком, 2006. – 452 с.
6. В.В. Круглов, М.И. Дли, Р.Ю. Голунов Нечеткая логика и искусственные нейронные сети. – М.: Физ-матлит, 2000. — 224 с.
7. D. Michie, D.J. Spiegelhalter, C.C. Taylor Machine Learning, Neural and Statistical Classification. 1994
8. Зайченко Ю.П. Оценка кредитних банковских рисков с использованием нечеткой логики// Intelligent Information and Engineering Systems. – 2008. - №13 – с. 190-200
9. Зайченко Ю.П. Нечеткие модели и методы в интеллектуальных системах. – К.: Издательский дом «Слово», 2008. – 334с.

УДК 681.327.12.001.362

ДОРОГОЙ Я.Ю.

АРХИТЕКТУРА ОБОБЩЕННЫХ СВЕРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ

В статье рассмотрена архитектура обобщенных сверточных нейронных сетей, позволяющих использовать преимущества классических сверточных нейронных сетей с дополнительными возможностями нового класса в задачах распознавания человека по фотопортрету.

The structure of the generalized convolutional neural networks which allow advantages of classical convolutional neural networks to be used with capabilities of new network class for problem of human face recognition was described in this article.

Введение в проблему

Для распознавания лиц людей широко используется архитектура нейронной сети, которая получила название сверточной нейронной сети (СНС). Данная нейронная сеть была впервые описана в [1,2] и сейчас успешно применяется для решения широкого класса задач связанных с распознаванием паттернов, таких как распознавание трехмерных объектов, предсказание погоды, автоматическое управление и др.

Изначально архитектура сверточной нейронной сети разрабатывалась с учетом особенностей строения некоторых участков мозга человека, ответственных за зрение.

В данной статье предлагается ряд улучшений в архитектуре обычной сверточной нейронной сети, которые позволят повысить эффективность работы системы распознавания, дадут возможность применять сверточные нейронные сети к более широкому классу задач.

Статья является продолжением исследований сверточных нейронных сетей, впервые описанных в [3].

Анализ существующих решений

В 1981 году нейробиологи Торстен Визел и Девид Хабел исследовали зрительную кору головного мозга кошки и выявили, что существуют так называемые простые клетки, которые особенно сильно реагируют на прямые линии под разными углами и сложные клетки, что реагируют на движение линий в одном направлении.

Позднее Ян Лекун предложил использовать так называемые сверточные нейронные сети, как аналог зрительной коры головного мозга для распознавания изображений [1,2].

Данный тип сетей хорошо зарекомендовал себя для решения проблемы распознавания человека по фотопортрету [3]. В ходе подробного исследования был выявлен ряд недостатков СНС:

1. Высокая сложность архитектуры.
2. Полносвязность.
3. Фиксированная площадь окна слоя свертки.

Цель работы

Целью работы является разработка новой архитектуры сверточной нейронной сети, которая позволит устранить недостатки, описанные выше, а также, по возможности, улучшить точность распознавания и скорость обучения сети.

Архитектура классической сверточной нейронной сети

Рассмострим архитектуру обычной сверточной нейронной сети (рис. 1.), описанной в [3].

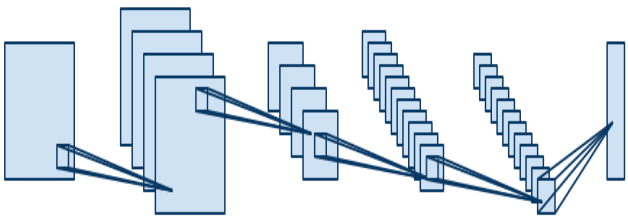


Рис. 1. Структура СНС

Нейронная сеть состоит из пар слоев - слоев подвыборки и слоев свертки, каждый из которых в свою очередь состоит из карт признаков.

Нетрудно убедиться в том, что каждая карта признаков в идеале фильтрует изображение, находя какой-то один определенный, специфичный для данной карты, признак. Например, первая карта признаков научена искать кружочки, вторая – квадратики и т.д.

Исходное изображение подается на входной слой. В первом слое подвыборки каждая карта признаков осуществляет поиск определенного, закрепленного только за данной картой, признака. Достигается это за счет использования общей для всей карты признаков матрицы весов и особой организацией локального рецептивного поля для каждого нейрона такой карты.

Каждый нейрон карты признаков получает входные данные от прямоугольной области размера $n \times m$ входного изображения. Такая область достаточно мала и множество таких областей на входном изображении пересекаются и накладываются по принципу черепицы.

Смежные нейроны карты признаков получают в качестве входного воздействия смежные прямоугольные области, причем весовые коэффициенты для всех нейронов карты признаков будут одинаковыми.

Для простоты изложения будем называть область, формирующую локальное рецептивное поле нейрона слоя подвыборки, окном. Соответственно, площадь окна – количеством нейронов в такой области.

Таким образом, можно говорить о том, что карта признаков в целом осуществляет операцию поиска признака во входных данных. Другие карты признаков имеют другой набор весовых коэффициентов и, соответственно, осуществляют поиск других признаков во входных данных.

Конкретные признаки, извлекаемые той или иной картой признаков, определяются в процессе обучения нейронной сети с учителем.

n и m – достаточно малые числа, которые определяют разрешающую способность нейронной сети – минимальный размер признака, который данная сеть может регистрировать.

Для введения инвариантности нейронной сети к смещениям и небольшим деформациям, используется слой свертки. Для каждой карты признаков существует соответствующая ей карта свертки, которая уменьшает размерность карты признаков с n×m до n/2×m/2 путем усреднения значений по квадрату 2×2 нейронов.

После выполнения свертки сеть теряет часть информации о точном положении найденного признака, но сохраняет информацию относительно взаимного расположения различных признаков.

Следующий слой подвыборки осуществляет аналогичную первому слою сегментацию входных данных на прямоугольные области $n \times m$, только входными данными второго слоя служит выход первого слоя. Т. о., каждая карта признаков второго слоя осуществляет поиск признаков второго порядка одновременно во всех картах признаков первого слоя.

Очевидно, что с ростом количества слоев уменьшается размерность каждой карты признаков, хотя, в целом, количество нейронов в слое сильно растет за счет использования большего количества карт признаков в верхних слоях сети.

Сверточной нейронной сети с тремя парами слоев подвыборки-свертки вполне достаточно для точного распознавания лиц людей [1].

Такая нейронная сеть хорошо себя зарекомендовала в задачах распознавания, но ее использование в некоторых случаях достаточно проблематично.

Одной из проблем классической сверточной нейронной сети является подбор оптимального значения размера локального рецептивного поля (окна) нейрона в слое подвыборки. Малые значеня n и m позволяют повысить разрешающую способность сети и дают возможность находить довольно малые признаки, но в тоже время аналогичный признак большего масштаба будет пропущен и принят за совокупность других признаков. Таким образом, классическая сверточная нейронная сеть плохо работает с изображениями, на которых могут присутствовать одинаковые признаки разного масштаба (например, на ненормализованных по масштабу изображениях)

Архитектура обобщенной сверточной нейронной сети

Допустим, на картинке есть как маленькие кружочки, так и большие, и необходимо найти максимум кружочков в первом же слое подвыборки.

Меняя площадь локального рецептивного поля каждого нейрона в каждой карте признаков можно добиться нахождения кружочков разного размера за один раз, но тогда кусочки других размеров останутся незамеченными.

В первом слое подвыборки сделаем несколько карт признаков с одним размером рецептивного поля, несколько – с чуть большим размером поля, несколько – с еще большим.

Для решения этой проблемы предлагается использовать в одном слое карты признаков хо-

тя и одного размера, но с разными размерами окна для разных карт признаков.

Например, вместо 10 карт признаков с размером окна каждого нейрона *3×3* предлагается использовать 5 карт признаков с размером поля *3×3*, 3 карты с полем *5×5* и 2 карты признаков с размером окна *7×7*.

Такая конфигурация позволит находить признаки разного размера одновременно сразу в первом слое подвыборки, что должно повысить общее качество распознавания объектов, ускорить обучение нейронной сети и способствовать уменьшению количества карт признаков в высших слоях нейронной сети, что приведет к уменьшению количества связей нейронной сети, и как следствие – уменьшение потребления памяти и ускорение обучения.

При такой организации первого слоя подвыборки закономерно возникает проблема с пересечением окон смежных нейронов одной карты признаков – для получения одинакового размера карт признаков с разными размерами окон необходимо разместить одинаковое количество таких полей на входном слое, что ведет к увеличению площади пересечения смежных полей с ростом размера поля.

К сожалению, бороться с таким резким увеличением количества связей крайне проблематично, но следующее улучшение позволит уменьшить общее количество связей в сверточных нейронных сетях, что особенно актуально для описанной в данной статье архитектуры сверточной нейронной сети.

Рассмотрим подробно один нейрон из карты признаков первого слоя подвыборки. Такой нейрон получает информацию из прямоугольной области входного изображения, формирующей его локальное рецептивное поле (окно). Общее количество входящих связей для нейрона равно $n \times m$, где $(n; m)$ – размер окна.

С увеличением n и m пропорционально растет количество связей, но в тоже время вклад, вносимый одной связью в суммарный вход нейрона уменьшается.

Это означает, что при большом количестве связей можно без значительного ущерба для качества работы сети переходить к разреженной форме соединений, удаляя некоторую малую их часть случайным образом.

Например, для окна *5×5* нейронов можно удалить 5 соединений, незначительно уменьшив точность, но при этом на 20% сократив общее количество соединений.

В отличие от соединений, общее количество весов за счет использования методики разделяемых весов, растет незначительно и нет смысла их сокращать.

Для данной карты признаков, применяя разрежение матрицы связей входного слоя с картой признаков, можно получить значительное сокращение связей, уменьшив, тем самым, расход памяти и ускорив процесс обучения сети.

Следует отметить, что в случае разрежения матрицы связей каждый нейрон карты признаков получает индивидуальную, отличную от соседних нейронов в данной карте признаков, матрицу связей.

Применение вышеизложенных рекомендаций по улучшению архитектуры сверточной нейронной сети ставит задачу оптимального распределения карт признаков с разным размером поля, а также задачу выбора количества отсекаемых связей в случае разрежения матриц связей.

К сожалению, количественные параметры сверточной нейронной сети должны подбираться строго под поставленную задачу после проведения ряда экспериментов для доказательства допустимости той или иной конфигурации.

Следует также отметить, что приведенные выше выкладки, несмотря на свою ориентированность на задачу распознавания образов, могут быть вполне использованы для построения сверточных нейронных сетей, решающих иные сходные задачи, к числу которых относятся прогнозирование и управление.

Эксперименты и результаты

Для проверки данных гипотез был проведен ряд экспериментов со множеством сверточных нейронных сетей. В нейронные сети вносились предложенные улучшения, осуществлялся анализ степени влияния той или иной модификации на качество распознавания, скорость обучения и пр. Несмотря на то, что в конечном итоге предложенные улучшения оправдали лишь некоторые из возложенных на них надежд, тем не менее, они позволяют значительно повысить качество распознавания объектов, в т. ч. и лиц людей. Все эксперименты проводились с использованием набора программного обеспечения PANN [4,5].

Для исследования нейронные сети тренировались на фотографиях 5-и человек из базы лиц ORL Faces по 10 фотографий на каждого. Из

имеющихся 50 фотографий 60% были использованы для непосредственного обучения, в то время как остальные – для промежуточного тестирования нейронных сетей на ранее не виденных образах.

Ниже приведены некоторые полученные зависимости для конфигурации классических сверточных нейронных сетей (рис. 2-8):

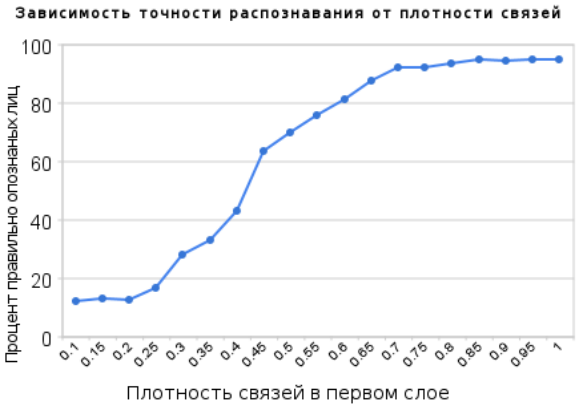


Рис. 2. Зависимость точности распознавания от плотности связей

Рис. 3. Зависимость точности распознавания от размеров окна в первом слое

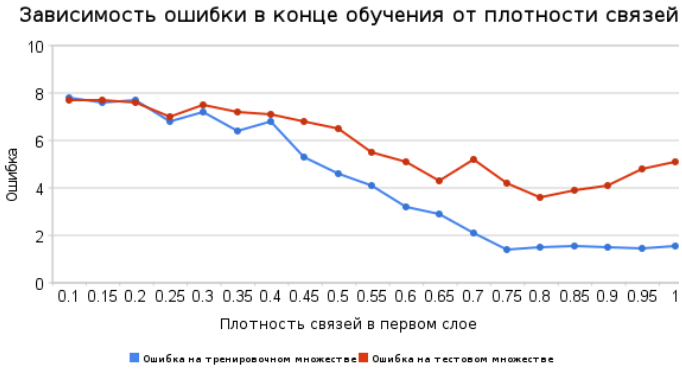


Рис. 4. Зависимость ошибки распознавания от плотности связей

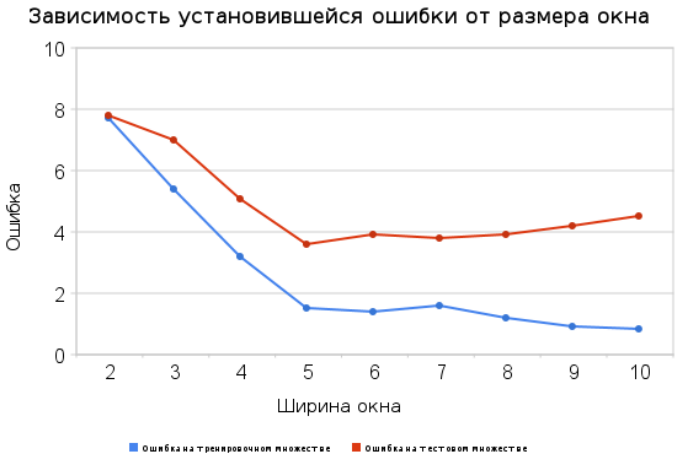


Рис. 5. Зависимость установившейся ошибки от размера окна

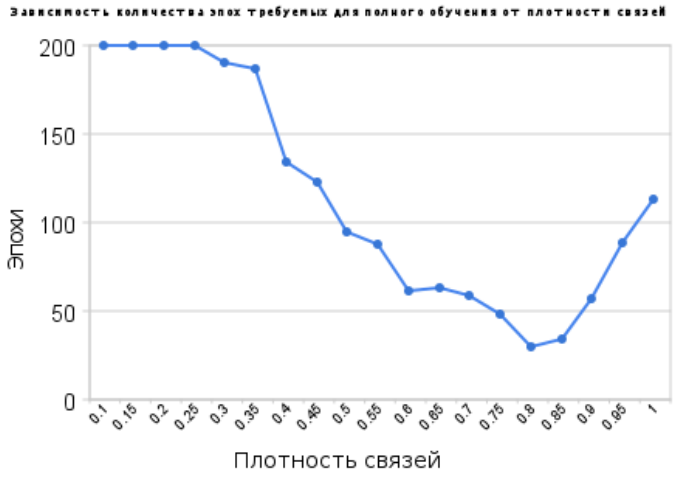


Рис. 6. Зависимость скорости обучения от плотности связей

Рис. 7. Зависимость скорости обучения от размера окна

Рис. 8. Зависимость скорости обучения от параметра скорости обучения

Очевидно, что чем больше плотность сети, тем лучше она учится. Но если плотность слишком большая (>0.8), то обучение замедляется.

Сравнение классической и обобщенной сверточных сетей

Далее была проведена серия экспериментов над топологиями сверточных нейронных сетей с внесенными изменениями, предложенными выше. Ниже приведены графики процессов обучения и тестирования двух различных нейронных сетей с различной конфигурацией.

Сеть №1 – это классическая сверточная нейронная сеть с размером окна *5×5* и 7-ю картами признаков в первом слое подвыборки и плотностью связей 80%, что дает в сумме 5600 связей в первом слое нейронной сети.

Сеть №2 – обобщенная сверточная нейронная сеть с двумя полносвязными картами признаков размером *3×3*, двумя картами признаков *5×5* и плотностью связей 80%, а также тремя картами признаков размером *7×7* и плотностью связей 50%. В результате количество связей в первом слое сократилось до 5320, в то время как в первом слое были использованы карты признаков разного размера – от *3×3* до *7×7*, что позволяет одновременно обрабатывать первичные признаки различного размера.

Результаты экспериментов представлены на рис. 9-10.

Для экспериментов были взяты 50 фотографий 5 людей – 30 шт. для обучения, 20 – для тестов. В результате экспериментов получены следующие данные:

- Сеть 1 – из 20 фотографий 2 неправильно опознаны;

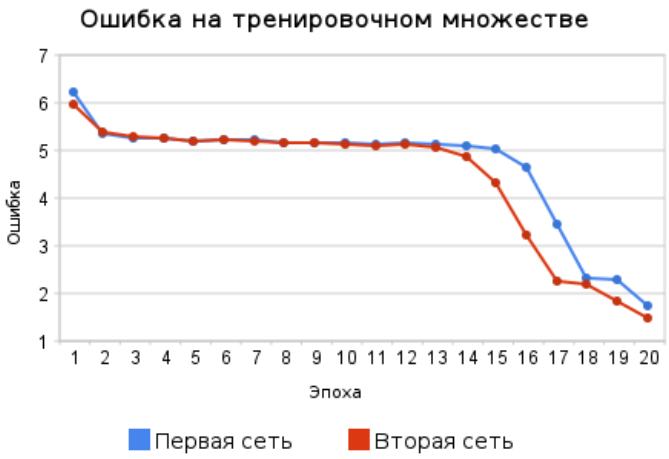


Рис. 9. Ошибка на тренировочном множестве

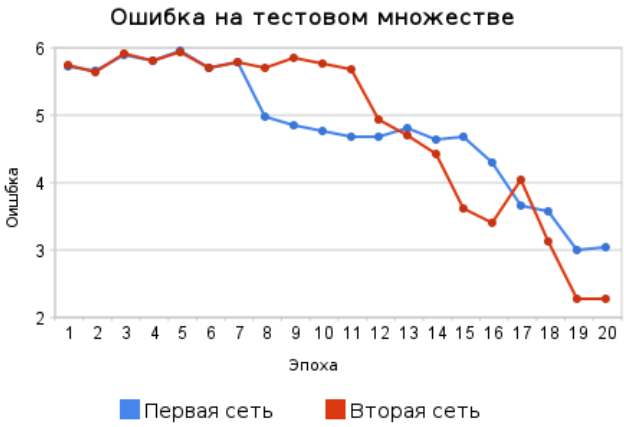


Рис. 10. Ошибка на тестовом множестве

- Сеть 2 – из 20 фото 0 неправильно опознанных.

После серии тестов оказалось, что для классической сверточной сети точность распознавания составляет порядка 97-98%, а для обобщенной сверточной нейронной сети – 99-99,5%.

Выводы

Эти и другие эксперименты над сверточными нейронными сетями подтвердили превосходство обобщенной топологии над классической. Использование нескольких карт признаков разного размера одновременно позволяет получить лучшее обобщение для обученной нейронной сети, т.е. снизить процент ложных срабатываний для образов, не входивших во множество обучения. Благодаря снижению плотности связей для больших карт признаков удается вносить эти изменения в архитектуру сети без роста количества связей, что благотворно сказывается на потреблении памяти, скорости обучения.

К сожалению, предложенные улучшения не дают видимого выигрыша в скорости обучения

сверточной нейронной сети, что дает почву для новых исследований в этой области.

Список литературы

1. LeCun Y. A theoretical framework for backpropagation // Proc. of IEEE. – 1998. – P.21-28.
2. LeCun Y., Bottou L., Bengio Y., Haffne P. Gradient-Based Learning Applied to Document Recognition // Proc. IEEE. – 1998. – P.59-67.
3. Дорогой Я.Ю. Применение компактных ячеистых сверточных нейронных сетей для биометрической идентификации человека по лицу // Вісник НТУУ “КПІ”, “Інформатика, управління та обчислювальна техніка”. – 2007. – №46. – С.135-149.
4. Дорогой Я.Ю., Яшин В.Е. Программный комплекс для симуляции многопоточных нейронных сетей // Вісник НТУУ “КПІ”, “Інформатика, управління та обчислювальна техніка”. – 2008. – №49. – С.123-127.
5. Дорогий Я.Ю., Яшин В. Є., Яцук С. В. Застосування багатопотокового симулятора нейронних мереж до задачі розпізнавання облич // Тези 5-ї Міжнародної науково-технічної конференції «Інформаційно-комп’ютерні технології 2010». - Житомир: ЖДТУ, 20-22 травня 2010 року. – С.53-55.