автор тез вважає неправильною ситуацію, коли деякі науково-педагогічні працівники роблять із цього висновок, ніби при викладанні курсів з комбінаторних багатокрокових ігор досить приділяти увагу лише підходу з *SG*-оцінками позицій та ігнорувати підхід із самими лише виграшними та програшними позиціями (відомими також як *N*- та *P*-позиції). (Наприклад, у [2, розд. 4] безальтернативно описується сам лише підхід з *SG*-оцінками – навіть там, де поняття виграшних і програшних позицій було б зрозумілішим і легшим для знаходження, і це не зважаючи на те, що книга [2] в цілому позиціонується як опис багатьох різних способів аналізу багатьох різних ігор.) А ці тези, на думку автора, є достатнім аргументом, чому слід приділяти увагу обом підходам, підкреслюючи переваги й недоліки кожного з них.

**Література:**

1. Богданов Э., Луценко А. Комбинаторная теория игр. Теорема Шпрага–Гранди. *Зимняя школа по программированию* : збірник. Харків : ХНУРЕ, 2010. С. 123–165.
2. Арсак Ж. Программирование игр и головоломок. Москва : Наука, 1990. 224 с.
3. Ferguson T. G. Game theory : class notes. 2020. 33 с. URL : https://www.cs.cmu.edu/afs/cs/academic/class/15859-f01/www/notes/comb.pdf (дата звернення : 04.05.2023).
4. Неупереджена гра *Вікіпедія* : веб-сайт URL: https://uk.wikipedia.org/wiki/Неупереджена_гра (дата звернення : 04.05.2023).
5. Вступ до алгоритмів. Переклад з англійської третього видання / Кормен Т. Г., Лейзерсон Ч. Е., Рівест Р. Л., Стайн К. Київ : К.І.С., 2019, 1288 с.
6. Матеріали школи програмування переможців міжнародного конкурсу з інформатики та комп'ютернох вправності «Бебрас–2017»: навчально-методичний посібник / Жуковський С. С., Порубльов І. М., Скляр І. В., Шпакович Р. С. Кам'янець-Подільський : Аксіома, 2018. 96 с.
7. Функція Шпрага-Гранді *Вікіпедія* : веб-сайт URL: https://uk.wikipedia.org/wiki/Функція_Шпрага-Гранді (дата звернення : 04.05.2023).

**УДК 004.942:519.179.2**

## ВИЗНАЧЕННЯ ЛОКАЛІЗАЦІЇ КОНФЛІКТІВ У КОМБІНОВАНОМУ ПІДХОДІ ДО ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ ПРОГРАМНИХ КОМПОНЕНТІВ З ПАРАЛЕЛІЗМОМ

*Супруненко О. О., Онищенко Б. О.*
*Черкаський національний університет імені Богдана Хмельницького*

**Abstract.** Two aspects of the application of the method's of synthesis and analysis of the combined simulation model of software are considered. The first aspect is related to building a model for partially connected parallel processes, the

second - concerns the way of localization of a conflict situation when the elements of the model are fully covered by invariants elements.

У сучасних проектах висока складність програмного забезпечення, яке розробляється, потребує детального аналізу проектних рішень. Тому є необхідність розробки підходів, методів та інструментальних засобів моделювання ПЗ і його компонентів [1]. Для підвищення працездатності моделей програмних засобів з паралелізмом разом зі структурним аналізом потрібно проводити їх динамічний аналіз [2]. Ця задача є актуальною і для розробленого ПЗ, яке потребує аналізу реалізованих рішень для підвищення ефективності його супроводу. Підходи та інструментальні засоби дослідження мають бути пристосовані до моделювання реалізованого ПЗ, що передбачає візуальну перевірку моделі на відповідність реалізованому коду, а також дозволяє автоматизовано аналізувати її динамічні властивості.

Для побудови та проведення поглибленого аналізу моделей ПЗ у роботі [3] запропонований комбінований підхід до імітаційного моделювання систем з паралелізмом, який передбачає побудову моделі програмного засобу на основі мережі Петрі. Зазначена модель має однозначний математичний опис, що дозволяє аналізувати статичні та динамічні властивості аналітично. Також модель будується за принципом структурної подібності, що дозволяє проводити візуальний аналіз в процесі побудови моделі та при її імітаційному моделюванні. Це важливо [4] як при проектуванні програмних компонентів, так і при збірці їх у цілісний програмний продукт, а також при аналізі реалізованого ПЗ.

На основі комбінованого підходу до імітаційного моделювання систем з паралелізмом [3-4] побудована методика синтезу та аналізу комбінованої імітаційної моделі ПЗ, яка передбачає проведення імітаційного моделювання для виявлення «поверхневих» помилок та побудову матричного опису моделі. На наступному етапі матричний опис дозволяє провести поглиблений аналіз моделі на основі перевірки властивостей: живості, безпечності (обмеженості), повторюваності, збережуваності, безконфліктності та керованості [2].

Архітектура комбінованого підходу до імітаційного моделювання систем з паралелізмом [4] передбачає використання інструментарію на основі формальних мов мереж Петрі L-типу з елементами мови G-типу. Це дозволяє при аналізі моделі працювати з проміжними розмітками. Також використання елементів мов G-типу надає можливість аналізувати локальні компоненти моделі та коректність їх злиття у часткові підмоделі, що при подальшій збірці складають цілісну модель програмного засобу.

Важливим при використанні методики є аналіз імітаційної моделі на єдиному представленні, яке однозначно описується математично, що спрощує аналіз компонентів та збірної моделі ПЗ у порівнянні з методиками, які для перевірки певних груп властивостей використовують кілька моделей з різними засобами опису [5].

Натомість, при аналізі основних конфліктних ситуацій на моделі треба звертати увагу не лише на загальну логіку роботи модельованої системи, а і на особливості розв'язуваних даною системою завдань, оскільки такі особливості не вдіється автоматично виявити вищенаведеним комбінованим підходом. Наприклад, на рис. 1 (*а*) представлений варіант моделі, що відображає виконання трьох задач, який не залежить від порядку їх виконання. Варіант моделі (рис. 1, *а*) не має критичних властивостей.

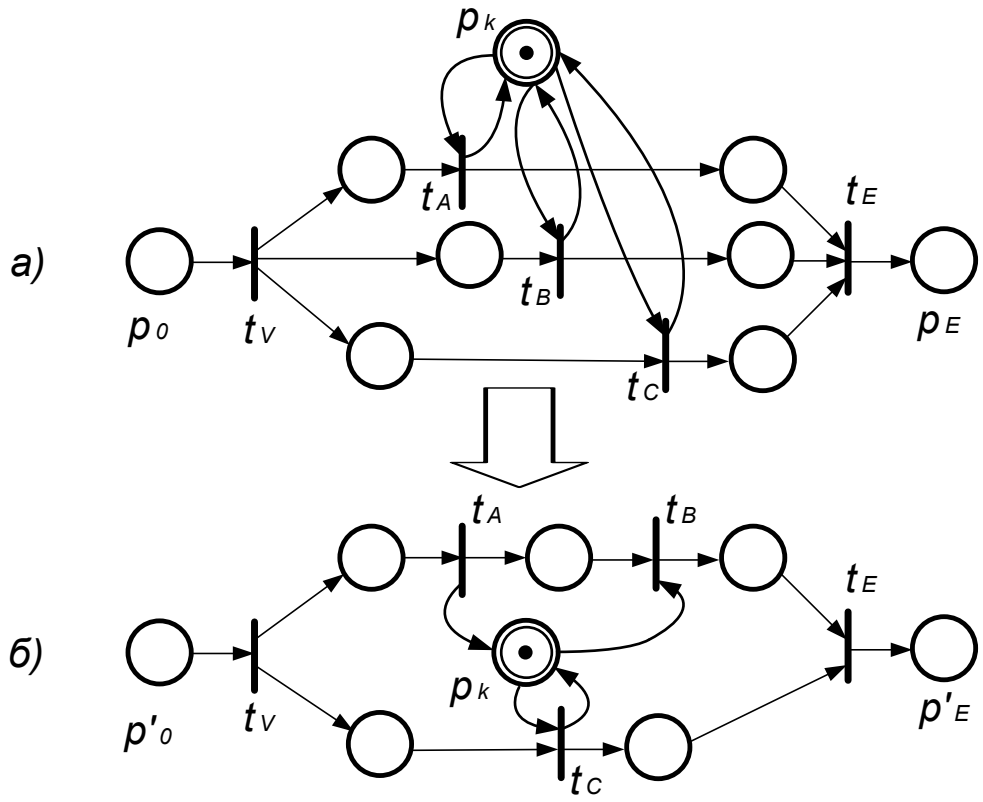


Рис. 1. Перетворення недетермінованого порядку обробки задач на частково впорядкований.

Але, коли за особливостями функціонування модельованої системи з'являються обмеження, наприклад, потрібно, щоб деяка задача *B* виконувалась лише після виконання задачі *A*, аналізовану модель потрібно корегувати. Один з варіантів такої корекції представлений на рис. 1 (б).

Таким чином, аналіз моделі ПЗ потрібно проводити не лише з використанням імітаційного моделювання. Для з'ясування особливостей задачі потрібно аналізувати відображені у моделі бізнес-процеси та їх обмеження. Для ефективного вирішення подібних задач потрібний глибший аналіз особливостей функціонування моделі в сенсі реалізації функціонування кожного бізнес-процесу, який передбачені умовами задачі.

Іншим важливим аспектом застосування запропонованої методики є реалізація підходу, в якому передбачене використання інваріант разом з рангом матриці інцидентності для визначення повної чи неповної керованості моделі,

що на практиці означає передбачуване чи неповністю передбачуване функціонування моделі. При визначенні неповної керованості моделі за рангом матриці інцидентності, у випадках повного покриття елементами інваріант всіх складових моделі, виникає необхідність визначення локалізації логічних помилок в елементах моделі. Пропонований підхід дозволяє визначити елементи моделі, які провокують неповну керованість, керуючись розмітками, які вказані у S-інваріантах.

У тестовій моделі 1 [2] з початковою розміткою $\mu_0 = [1 \quad 0 \quad 0 \quad 0]^T$ дотримані властивості живості, обмеженості та повторюваності (за Т-інваріантом), властивості збережуваності та безконфліктності (за S-інваріантом). Але ранг матриці інцидентності $\text{rang}(W) < 4$ вказує на можливість непередбачуваного функціонування моделі. Яким чином визначити вершини, що викликають непередбачувану поведінку моделі? При встановленні у моделі 1 [2] початкової розмітки, на яку вказує S-інваріанта: $S = [5 \quad 2 \quad 1 \quad 1]^T$ і проведенні імітаційного експерименту за послідовностями, відмінними від Т-інваріани, приходимо до розмітки $\mu_{53} = [0 \quad 0 \quad 0 \quad 31]^T$, що дозволяє виявити прихований тупик. За цією розміткою визначаємо локалізацію тупика, що, очевидно, пов'язана з вершиною місця $p_4$. У цій вершині відбувається накопичення міток, що приводить до виникнення тупика. Таким чином, у пропонованій методиці застосоване поєднання імітаційного моделювання з етапами аналітичного дослідження моделі, що дозволяє не тільки вказати на наявність потенційної конфліктної ситуації, а й визначити елементи моделі, з яких потрібно починати аналіз при її корегуванні.

**Література:**

1. Яковина В.С. Федасюк Д.В., Мамроха Н.М. (2010). Аналіз використання аспектно-орієнтованого програмування як засобу підвищення надійності програмного забезпечення. *Інженерія програмного забезпечення*, 2, 24-29.

2. Супруненко О.О., Онищенко Б.О., Гребенович Ю.Є. (2022). Аналіз прихованих помилок у моделях програмних систем на основі мереж Петрі. *Електронне моделювання*, 2(44), 38-50. doi: https://doi.org/10.15407/emodel.44.02.038

3. Супруненко О.О. (2019). Комбінований підхід до імітаційного моделювання динаміки програмних систем на основі інтерпретацій мереж Петрі. *KPI Science News*, 5-6, 43-53. doi: 10.20535/kpi-sn.2019.5-6.174596

4. Suprunenko, O. (2021). Combined approach architecture development to simulation modeling of systems with parallelism. Eastern-European Journal of Enterprise Technologies, 4(4(112)), 74-82. doi: https://doi.org/10.15587/1729-4061.2021.239212.

5. Esparza J. (1994). Model Checking using net unfoldings. *Science of Computer Programming*, 23, 151-195.