Originality. *In this article, taking into account the latest trends in educational policy and the specifics of training in specialties 104 "Physics and Astronomy", 105 "Applied Physics and Nanomaterials", 113 "Applied Mathematics", 126 "Information Systems and Technologies", namely - the trend towards algorithmization when solving certain practical problems, it is advisable to use visualization in the study of mathematical analysis by the above students – and during the assimilation of theoretical material, and when solving problems. We have identified three stages of mastering mathematical formulas and proposed a number of exercises.*

Conclusion. *We see further research in the development of the same system of tasks for each topic of mathematical analysis, taking into account the specifics of training future professionals in the above specialties.*

Keywords: *mathematical analysis, semiotic approach, mathematical formulas, application of integrals in the exercises of geometry and mechanics, physics students, programming students*

Одержано редакцією 26.08.2021 р.
Прийнято до публікації 27.10.2021 р.

УДК 519.688

DOI 10.31651/2076-5886-2021-1-32-49

PACS 02.70.-с

ПОРУБЛЬОВ Ілля Миколайович,
викладач, Черкаський національний університет імені Богдана Хмельницького
e-mail: ilya@vu.cdu.edu.ua
ORCID: 0000-0001-7369-3862

ЩЕ ОДИН АЛГОРИТМ ПОШУКУ ЧИСЕЛ З МАКСИМАЛЬНОЮ КІЛЬКІСТЮ ДІЛЬНИКІВ (НА ОСНОВІ ІДЕЇ МНОЖИНИ НЕДОМІНОВАНИХ ПАР)

Ця стаття по суті є продовженням статті [6]. В ній розглянуто майже ту саму постановку задачі (пошук числа, що має максимальну кількість дільників серед всіх чисел проміжку, але цього разу лише від 1 до вказаного N, і треба шукати мінімальне з чисел, що мають цю максимальну кількість дільників). Попри схожість формулювань, запропоновано принципово інший спосіб розв'язання, що базується на введеному понятті недомінованих пар: пара (число, кількість його дільників) вважається недомінованою, якщо не існує менших чисел, що мали б більшу або рівну кількість дільників. Аналогічно [6], розглянуто узагальнення, коли максимальна кількість дільників шукається не серед усіх чисел проміжку, а лише серед не кратних деякому K (натуральному, більшому або рівному 2, не обов'язково простому).

Ключові слова: *кількість дільників, надскладені числа, недоміновані пари, впорядкована словникова структура даних, map, lower_bound, upper_bound.*

Вступ

Оскільки стаття є по суті продовженням [6], причини актуальності аналогічні — істотно різні кількості дільників у різних чисел, що впливає на тривалість роботи деяких алгоритмів. Використано спільну з [6] нумерацію: означення та твердження, що вже були там і виявилися важливими також і тут, повторені під тими ж номерами, а новим надані нові (не використані в [6]) номери.

Дотримано точки зору, що натуральними є цілі додатні числа (а 0 не є).

Означення 1. Будемо, згідно [2], позначати кількість дільників числа n як $\tau(n)$.

Означення 2. Будемо, згідно [1], називати число n ***надскладеним***, якщо кількість його дільників строго перевищує кількість дільників будь-якого меншого числа. Інакше кажучи, n надскладене, коли

$$\forall j_{1\le j<n}\left(\tau(j)<\tau(n)\right). \qquad (1)$$

Як і в [6], зручно вважати, що прості числа занумеровані як $p_0=2$, $p_1=3$, $p_2=5$, …, а коли з контексту ясно, про розкладення якого числа йдеться, то k_0, k_1, k_2, … позначають відповідні показники степенів ($p_0 = 2$ підноситься до степеню k_0, $p_1 = 3$ до степеню k_1, …, причому k_i можуть бути як натуральними, так і нулями).

Мета статті

Описати принципово інший, чим у [6], розроблений алгоритм для розв'язання задачі пошуку чисел з великою кількістю дільників, у таких варіантах постановки:

1. Дано натуральне число N. Для проміжку всіх натуральних чисел від 1 до N (включно), знайти мінімальне з чисел, що має максимальну кількість дільників (інакше кажучи, останнє, воно ж найбільше, надскладене число цього проміжку).

3. Дані натуральні числа N та K ($K\ge 2$). Для проміжку натуральних чисел від 1 до N (включно), але ігноруючи числа, кратні K, знайти мінімальне з чисел, що має максимальну кількість дільників,

а) якщо гарантовано, що число K просте;

б) якщо такої гарантії нема.

В усіх варіантах постановки, потрібно знайти і максимальну кількість дільників, і мінімальне з чисел, для яких досягається така кількість дільників. Таку пару будемо називати також *повною відповіддю*. Коли виникатиме необхідність розрізнити, яке одне з цих двох чисел мається на увазі, вживатимуться словосполучення *відповідь-кількість* (дільників) та *відповідь-значення* (на якому числі досягається така кількість дільників).

Основною ідеєю цієї статті, повністю новою відносно [6], є ідея недомінованих пар, введена в озн. 4.

Виклад основного матеріалу

Повторимо потрібні зараз означення, твердження (без доведень) та наслідки з [6].

Твердження 1. Нехай відоме канонічне розкладення числа N на прості множники

$$N=p_1^{k_1}\cdot p_2^{k_2}\cdot\ldots\cdot p_m^{k_m} \qquad (2)$$

(всі p_i прості та різні, всі k_i натуральні). Тоді кількість різних дільників N дорівнює

$$\tau\left(p_1^{k_1}\cdot p_2^{k_2}\cdot\ldots\cdot p_m^{k_m}\right)=(k_1+1)\cdot(k_2+1)\cdot\ldots\cdot(k_m+1). \qquad (3)$$

(доведення можна бачити, наприклад, у [4, § 9])

Наслідок твердження 1. Якщо числа a та b взаємно прості, то $\tau(a\cdot b) = \tau(a)\cdot\tau(b)$; інакше, $\tau(a\cdot b) < \tau(a)\cdot\tau(b)$.

Твердження 2. Для всіх надскладених чисел, $k_0 \ge k_1 \ge \ldots$

Означення 3. Для постановки № 3б, будемо позначати $z_0, z_1, \ldots$ показники, з якими прості $p_0=2$, $p_1=3$, $p_2=5$, … входять у розкладення K, кратність якому заборонена: $K=p_0^{z_0}\cdot p_1^{z_1}\cdot\ldots$ Значення z_0, z_1, … цілі невід'ємні (можуть дорівнювати 0).

На цьому основні повтори з [6] завершені, далі йде новий матеріал.

Будемо працювати з парами $(n_i, \tau(n_i))$, тобто числом та кількістю його дільників. Теоретично, другий елемент кожної такої пари визначається першим. Однак, будемо зберігати готові другі елементи, а не обчислювати їх через перші, бо так швидше.

Означення 4. Будемо казати, що пара (a, $\tau(a)$) ***домінує*** пару (b, $\tau(b)$), й позначати це знаком «$\succ$», як $(a,\tau(a))\succ(b,\tau(b))$, коли

$$(a<b)\wedge(\tau(a)\geq\tau(b))\,; \qquad (12)$$

будемо також називати пару (b, $\tau(b)$) ***домінованою*** парою (a, $\tau(a)$), відповідно пару (a, $\tau(a)$) — ***домінуючою*** пару (b, $\tau(b)$).

Наприклад: пара (9, 3) домінує пару (13, 2), але пара (9, 3) домінована будь-якою з пар (6, 4) чи (4, 3). Це озн. 4 введено самостійно, взявши за основу Парето-домінування, описане у [7] та пізніше популяризоване в багатьох інших джерелах, зокрема [3].

Твердження 8. Введене в озн. 4 домінування є нелінійним (неповним) відношенням строгого порядку.

Доведення. Пари (a, $\tau(a)$) та (b, $\tau(b)$) однотипні та їх дві, тому домінування можна розглядати як бінарне відношення. Враховуючи строгу нерівність $a<b$ у перших дужках (12), відношення домінування іррефлексивне (жодна пара не домінує саму себе). Та сама умова $a<b$ гарантує також антисиметричність (не може бути двох різних пар, кожна з яких домінує іншу). Тепер транзитивність: якщо $(a,\tau(a))\succ(b,\tau(b))$ та $(b,\tau(b))\succ(c,\tau(c))$, то $a<b$, $b<c$, $\tau(a)\geq\tau(b)$, $\tau(b)\geq\tau(c)$, тому, за транзитивністю звичайних числових “$<$” та “$\geq$”, маємо також $a<c$, $\tau(a)\geq\tau(c)$, тобто $(a,\tau(a))\succ(c,\tau(c))$. Це відношення не є повним, тобто можуть бути такі пари (a, $\tau(a)$) та (b, $\tau(b)$), що жодна з них не домінує іншу; прикладами таких пар є, скажімо, (40, 8) та (100, 9). Отже, всі ознаки нелінійного відношення строгого порядку перевірено. ■

Твердження 9. Нехай $(a,\tau(a))\succ(c,\tau(c))$. Тоді для будь-якого натурального m, взаємно простого з a, $(a\cdot m,\tau(a\cdot m))\succ(c\cdot m,\tau(c\cdot m))$.

Доведення. Маємо $(a,\tau(a))\succ(c,\tau(c))$, тобто $a<c$ та $\tau(a)\geq\tau(c)$. Помноживши першу нерівність на m, маємо $a\cdot m<c\cdot m$, що є складовою (12) для пар ($a\cdot m$, $\tau(a\cdot m)$) та ($c\cdot m$, $\tau(c)$). Помноживши другу нерівність на $\tau(m)$, маємо $\tau(a)\cdot\tau(m)\geq\tau(c)\cdot\tau(m)$. Оскільки m взаємно просте з a, за наслідком твердж. 1, $\tau(a\cdot m)=\tau(a)\cdot\tau(m)$. Оскільки невідомо, чи m взаємно просте з c, за тим самим наслідком маємо $\tau(c\cdot m)\leq\tau(c)\cdot\tau(m)$. Тобто, маємо ланцюжок $\tau(a\cdot m)=\tau(a)\cdot\tau(m)\geq\tau(c)\cdot\tau(m)\geq\tau(c\cdot m)$; лишивши тільки крайні його вирази $\tau(a\cdot m)\geq\tau(c\cdot m)$, отримаємо іншу складову (12) для пар ($a\cdot m$, $\tau(a\cdot m)$) та ($c\cdot m$, $\tau(c)$). ■

Примітка до твердження 9. Взаємна простота a і m важлива, без неї твердження неправильне. Наприклад, $(12, 6)\succ(35, 4)$, але якщо домножити 12 та 35 на 16 (не є взаємно простим з 12), вийдуть пари (384, 14) та (560, 20), жодна з них не домінує іншу.

Означення 5. Будемо називати пару (v, $\tau(v)$) ***абсолютно недомінованою***, коли не існує жодної такої пари (a, $\tau(a)$), яка домінувала б пару (v, $\tau(v)$).

Примітка до означення 5. Якщо, враховуючи твердж. 8, вважати, що «$\succ$» з озн. 4 означає «нелінійно більше», то множина абсолютно недомінованих пар з цього озн. 5 є множиною всіх максимумів з множини $\{(n, \tau(n)) \mid n\in\mathbf{N}\}$.

Твердження 10. Абсолютно недомінованими є ті й лише ті пари (v, $\tau(v)$), які утворені з надскладеного (згідно озн. 2) числа v та кількості його дільників.

Доведення. Враховуючи нерівність $a<b$ у (12), для $(a, \tau(a))\succ(v, \tau(v))$ необхідно (але не достатньо) $a<v$. Якщо v надскладене, жодна пара (a, $\tau(a)$) при $a<v$ не домінує пару (v, $\tau(v)$), бо (1) вимагає $\tau(a)<\tau(v)$, а (12) вимагає $\tau(a)\geq\tau(v)$, що несумісно. Якщо ж

v не надскладене, то $\exists j_{1\le j<v}(\tau(j)\ge\tau(v))$ (бо це заперечення (1)), і можна взяти те $j = j^*$, яке задовольняє квантор існування, і для нього виконуються одночасно $j^* < v$ та $\tau(j^*) \ge \tau(v)$, що за (12) означає $(j^*, \tau(j^*)) \succ (v, \tau(v))$, тож $(v, \tau(v))$ не є абсолютно недомінованою. ■

Означення 6. Будемо називати пару $(v, \tau(v))$ ***недомінованою за умови …*** (замість «…» вказується додаткова умова), коли сама пара $(v, \tau(v))$ задовольняє вказану додаткову умову і не існує жодної такої пари $(a, \tau(a))$, для якої теж виконувалася б указана додаткова умова, і яка домінувала б пару $(v, \tau(v))$.

Примітка 1 до означення 6. Дозволяються як ситуація, що існує одна чи кілька пар $(a, \tau(a))$, які домінують $(v, \tau(v))$, але для цих $(a, \tau(a))$ не виконується додаткова умова, так і ситуація, що таких пар не існує. Тому, кожна абсолютно недомінована пара також і недомінована за будь-якої умови, якій задовольняє, а пара, недомінована за деякої умови, може бути абсолютно недомінованою, а може й не бути.

Примітка 2 до означення 6. Додаткова умова не змінює смисл домінування (озн. 4 чи формулу (12)). Змінюється лише множина пар, до яких його застосовують.

Означення 7. Позначимо за $S(N, j)$ (де N — натуральне, j — ціле невід'ємне) множину всіх пар $(v, \tau(v))$, недомінованих за умови «$1 \le v \le N$ та в розкладенні v ненульові степені можуть мати лише прості $p_0 = 2$, $p_1 = 3$, …, p_j». (Будемо також називати такі умови «умовами для $S(N, j)$».)

Примітка 1 до означення 7. Степені p_0, p_1, …, p_j можуть бути як ненульовими, так і нульовими, як вийде; головне, щоб для p_{j+1}, p_{j+2}, … степені були нульовими.

Примітка 2 до означення 7. З прим. 1 випливає, що коли пара задовольняє умовам для $S(N, j)$, то вона задовольняє також і умовам для $S(N, j+1)$, $S(N, j+2)$, …

Примітка 3 до означення 7. З прим. 2 не випливає, ніби $S(N, 0) \subseteq S(N, 1) \subseteq \ldots$, бо деякі пари, дозволені в $S(N, j+1)$ і заборонені в $S(N, j)$, можуть домінувати деякі пари з $S(N, j)$. Наприклад: $(8, 4) \in S(10, 0)$, але $(8, 4) \notin S(10, 1)$, бо $(6, 4) \in S(10, 1)$ і $(6, 4) \succ (8, 4)$.

Твердження 11. Для будь-якого $j \ge 0$ та будь-якого $N \ge 1$, якщо $(v, \tau(v)) \in S(N, j+1)$ і при цьому в розкладенні v показник $k_{j+1} = 0$, то $(v, \tau(v)) \in S(N, j)$.

Доведення. $(v, \tau(v)) \in S(N, j+1)$ означає, що $(v, \tau(v))$ задовольняє умовам для $S(N, j+1)$; раз при цьому $k_{j+1} = 0$, то й умовам для $S(N, j)$. Тому, $(v, \tau(v)) \notin S(N, j)$ могло би бути, лише якби існувала деяка пара $(a, \tau(a)) \succ (v, \tau(v))$, що задовольняла б умови для $S(N, j)$. Але, за прим. 2 до озн. 7, ця $(a, \tau(a))$ задовольняла б також і умови для $S(N, j+1)$. Тоді, враховуючи $(a, \tau(a)) \succ (v, \tau(v))$, було б $(v, \tau(v)) \notin S(N, j+1)$, що протирічить умові. ■

Твердження 12. Якщо множина S містить лише пари $(a_i, \tau(a_i))$, жодна з яких не домінує жодну іншу пару з S, і впорядкувати (відсортувати) всі пари $(a_0, \tau(a_0))$, $(a_1, \tau(a_1))$, …, $(a_T, \tau(a_T))$ з S, так, щоб виконалося $a_0 \le a_1 \le \ldots \le a_T$, то автоматично виконаються також $a_0 < a_1 < \ldots < a_T$ та $\tau(a_0) < \tau(a_1) < \ldots < \tau(a_T)$.

Доведення. $a_0 < a_1 < \ldots < a_T$ відрізняється від $a_0 \le a_1 \le \ldots \le a_T$ в точності забороною на повтори однакових значень, а правильність цієї заборони випливає з того, що розглядаємо класичні множини (не мультимножини), а пари $(a_i, \tau(a_i))$ не можуть відрізнятися лише другими елементами, маючи однакові перші, бо a_i визначає $\tau(a_i)$. Ситуація, коли для деяких a_i та a_k (таких, що $(a_i, \tau(a_i)) \in S$ та $(a_k, \tau(a_k)) \in S$) мають місце водночас $a_i < a_k$ та $\tau(a_i) \ge \tau(a_k)$, неможлива, бо це означало б $(a_i, \tau(a_i)) \succ (a_k, \tau(a_k))$, а це заборонено частиною «якщо» цього твердження. Тоді, раз при $a_i < a_k$ можливо лише

$\tau(a_i) < \tau(a_k)$, впорядкування $a_0 < a_1 < \ldots < a_T$ призводитиме до $\tau(a_0) < \tau(a_1) < \ldots < \tau(a_T)$. ■

Примітка до твердження 12. Зокрема, це твердження правильне для всіх $S(N, j)$ при будь-якому $j \geq 0$ та будь-якому $N \geq 1$, бо за озн. 7 кожна $S(N, j)$ якраз і є множиною пар, недомінованих за певних умов. Однак, важливо, що це твердж. 12 виконується не лише для $S(N, j)$, а також і для інших множин пар $(a_i, \tau(a_i))$, в яких жодна пара не домінує жодну іншу пару, навіть якщо нема чітко сформульованої додаткової умови.

Твердження 13. $S(N, 0) = \{(1, 1), (2, 2), (4, 3), \ldots, (2^{\lfloor \log_2 N \rfloor}, \lfloor \log_2 N \rfloor + 1)\}$.

Доведення. В межах умови «розглядаються лише степені двійки» збільшення числа означає збільшення кількості дільників, і для них умова (12) (більша чи рівна кількість дільників у меншого числа) неможлива. А що це вичерпний перелік пар, недомінованих за потрібної для $S(N, 0)$ умови, забезпечується тим, що у множину включили всі пари, де першими елементами є степені двійки вказаного діапазону. ■

Твердж. 13 малокорисне саме по собі, бо мало толку в задачі, обмеженій на самі лише степені двійки. Але воно стає дуже корисним, якщо розглянути його як базу математичної індукції та побудувати й довести відповідний крок індукції.

Твердження 14. Нехай вже відома множина $S(N, j)$, і над нею виконують такі дії:

1. $S_{\text{next}} := S(N, j)$, тобто спочатку нова множина S_{next} складається з усіх пар, що є елементами $S(N, j)$, і лише з них.
2. Для кожної з пар $(a, \tau(a))$, що належать $S(N, j)$, виконати такі дії:
   2.1. перебрати пари $\left(a \cdot p_{j+1}, \tau(a) \cdot 2\right)$, $\left(a \cdot p_{j+1}^2, \tau(a) \cdot 3\right)$, …, $\left(a \cdot p_{j+1}^s, \tau(a) \cdot (s+1)\right)$ (де s можна виразити як $\lfloor \log_{p_{j+1}} (N / a) \rfloor$ чи підібрати циклом `while`), тобто всі пари, де перший елемент не перевищує N і є добутком a на деякий степінь простого p_{j+1}; позначивши ці пари вигляду $\left(a \cdot p_{j+1}^i, \tau(a) \cdot (i+1)\right)$ як $(b, \tau(b))$, для кожної з них виконати такі дії:
      2.1.1. `if` ($(b, \tau(b))$ не домінована жодною з пар, вже наявних у S_{next}) `{`
         2.1.1.1. включити $(b, \tau(b))$ у S_{next};
         2.1.1.2. `if` ($(b, \tau(b))$ домінує хоча б одну з пар, вже наявних у S_{next})
            2.1.1.2.1. вилучити з S_{next} усі пари, доміновані парою $(b, \tau(b))$;
         `}`

Тоді утворена цими діями множина S_{next} дорівнює $S(N, j+1)$.

Доведення. Спочатку переконаємося, що ніяка пара, яка не задовольняє умову $S(N, j+1)$, не може потрапити в S_{next}, навіть тимчасово. Всі пари, які потрапляють в S_{next} ініціалізацією в п. 1, задовольняють умову для $S(N, j+1)$, враховуючи прим. 1 до озн. 7. Пари, які потрапляють в S_{next} у п. 2.1.1.1, задовольняють умову для $S(N, j+1)$, бо побудовані з тих, де в розкладенні є лише прості $p_0, p_1, \ldots, p_j$, домноженням на степінь простого p_{j+1}, а умова, що домножений перший елемент $\leq N$, теж забезпечена в п. 2.1.

Переконаємося, що неможлива ситуація, коли деяка пара $(f, \tau(f))$ недомінована за умов для $S(N, j+1)$, але не включена в S_{next}. Випадок, коли таке стається з f, для розкладення якого $k_{j+1} = 0$, неможливий (за твердж. 11, така пара вже є в $S(N, j)$, тож вона копіюється в S_{next} у п. 1). Інакше (для розкладення f маємо $k_{j+1} > 0$), слід дослідити, що саме додається у п. 2.1.1.1. Позначимо степінь з основою p_{j+1}, який входить у f, за r, а частку f / r за g (наприклад, при $f = 1800 = 2^3 \cdot 3^2 \cdot 5^2$, $p_{j+1} = 5$ буде $r = 5^2 = 25$, $g = 2^3 \cdot 3^2 = 72$). Якщо $(g, \tau(g)) \in S(N, j)$, то $(f, \tau(f))$ буде однією з пар $(b, \tau(b))$, які перебиратимуться вкладеними циклами п. 2 та п. 2.1, і в разі успішної перевірки п. 2.1.1 пара $(f, \tau(f))$ буде включена до S_{next} у п. 2.1.1.1 (а в разі неуспішної, її і не слід туди додавати, бо її домінує якась пара (одна чи кілька, це несуттєво), котра, згідно з попереднім абзацем, задовольняє умову для $S(N, j+1)$). Протилежна ситуація $(g, \tau(g)) \notin S(N, j)$ означає, що $(g, \tau(g))$ домінована деякою (щонайменше однією; якщо їх кілька, візьмемо будь-яку одну) парою $(a^*, \tau(a^*))$, належною $S(N, j)$. В такому разі, враховуючи взаємну простоту a^* з r (розкладення a^* не містить ненульових степенів p_{j+1}, тоді як r тільки їх і містить) та твердж. 9, все одно $(r{\cdot}a^*, \tau(r{\cdot}a^*)) \succ (f, \tau(f))$, причому пара $(r{\cdot}a^*, \tau(r{\cdot}a^*))$ буде розглянута у п. 2.1, бо $(a^*, \tau(a^*)) \in S(N, j)$, і, раз $(a^*, \tau(a^*)) \succ (g, \tau(g))$, то $a^* < g$, а отже й $r{\cdot}a^* < r{\cdot}g = f \le N$. Таким чином, ніщо потрібне не буде пропущене й правильність не втратиться, а з точки зору ефективності добре не розглядати безперспективну пару $(f, \tau(f))$. Далі (п. 2.1.1.2.1) вилучаються лише пари, доміновані деякою парою, що задовольняє умову для $S(N, j+1)$. Тому, пари, недоміновані за умов для $S(N, j+1)$, не будуть вилучені.

Таким чином, S_{next} не може не містити пар, які повинні бути в $S(N, j+1)$; лишилося показати, що S_{next} не може містити непотрібних (яких повинно не бути в $S(N, j+1)$). Що там не може бути пар, які навіть не задовольняють умову для $S(N, j+1)$, вже показано в першому абзаці цього доведення. Лишилося довести, що в S_{next} не може бути водночас і пари $(u, \tau(u))$, і пари $(w, \tau(w))$, які обидві задовольняють умову для $S(N, j+1)$, але $(w, \tau(w)) \succ (u, \tau(u))$. За припущенням індукції, в момент ініціалізації $S_{\text{next}} := S(N, j)$ у п. 1 такого не було, і лишається показати, що таке не могло з'явитися при виконанні цих дій. Єдиним (крім ініціалізації у п. 1) місцем, де пари включаються у S_{next}, є п. 2.1.1.1; ситуація, що домінована $(u, \tau(u))$ включається у S_{next}, коли там вже є домінуюча $(w, \tau(w))$, неможлива, бо цього не допустить умова п. 2.1.1; ситуація, що домінуюча $(w, \tau(w))$ включається у S_{next}, коли там вже є домінована $(u, \tau(u))$, тимчасово можлива, але, якщо таке і трапиться у п. 2.1.1.1, то п. 2.1.1.2.1 тут же вилучить $(u, \tau(u))$ з S_{next}, і після завершення п. 2.1.1.2.1 наявність у S_{next} відразу обох пар знов неможлива. ■

Примітка 1 до твердження 14. Для такої побудови $S(N, j+1)$ треба **лише** $S(N, j)$, а $S(N, j–1)$, $S(N, j–2)$, …, $S(N, 0)$ не потрібні (звісно, при $j \ge 1$). Це дозволяє економити пам'ять, починаючи побудову $S(N, j+1)$ при $j \ge 1$ з очищення вже не потрібної $S(N, j–1)$.

Примітка 2 до твердження 14. Важливо створювати в п. 1 копію множини, а не використовувати одну множину і для циклу п. 2, і для змін у п. 2.1.1.1 та п. 2.1.1.2.1. Хоча б тому, що пари, описані в п. 2.1, правильні, лише якщо a не кратне p_{j+1}.

Примітка 3 до твердження 14. З останнього абзацу доведення випливає, що наприкінці **кожної** ітерації циклу п. 2.1 множина S_{next} містить лише пари, жодна з яких не домінує жодну іншу. □

Теоретично, якщо застосувати твердж. 14 нескінченну кількість разів, вийде недомінованість за умови «задіяні прості p_0, p_1, … (до нескінченності)», тобто абсолютна недомінованість, яка за твердж. 10 є характеристичною властивістю надскладених. Практично, апеляція до нескінченності неконструктивна, але є верхня межа проміжка N, і можна сформулювати та довести твердж. 15 та 16.

Твердження 15. Для будь-якого $j \ge 0$ та будь-якого $N \ge 1$, для кожної пари $(a, \tau(a)) \in S(N, j)$, якщо розкласти перший елемент цієї пари як $a = p_0^{k_0} \cdot p_1^{k_1} \cdot \ldots \cdot p_j^{k_j} \cdot \ldots$, виявиться $k_0 \ge k_1 \ge \ldots \ge k_j \ge k_{j+1} = k_{j+2} = \ldots = 0$.

Доведення. $k_{j+1} = k_{j+2} = \ldots = 0$ слідує з озн. 7. Лишається розібратися зі складовою $k_0 \ge k_1 \ge \ldots \ge k_j \ge 0$ (нетривіальною лише при при $j \ge 1$). Припустимо, ніби є хоч одна пара $(a^*, \tau(a^*)) \in S(N, j)$, така, що в $a^* = p_0^{k_0} \cdot p_1^{k_1} \cdot \ldots \cdot p_j^{k_j}$ має місце $p_u < p_w$ і $k_u < k_w$; тоді можна замінити $p_u^{k_u} \cdot p_w^{k_w}$ на $p_u^{k_w} \cdot p_w^{k_u}$ (лишивши решту степенів незмінними), й отримати таке число a', що $a' < a^*$ (всі множники лишаються додатними, і при цьому $k_w - k_u$ разів замість більшого p_w взято менше p_u), і кількості дільників однакові ($\tau(a^*) = \tau(a')$, бо при обчисленні за (3), обмін місцями множників (k_w+1) та (k_u+1) не впливає на результат, а решта множників, якщо є, незмінні). Тобто, $(a', \tau(a')) \succ (a^*, \tau(a^*))$. При цьому $a' < N$ (бо в ланцюжку $a' < a^* \le N$ перша нерівність з’ясована кілька рядків тому, а друга випливає з $(a^*, \tau(a^*)) \in S(N, j)$. Також, $(a^*, \tau(a^*)) \in S(N, j)$ значить, що розкладення a^* не містить ненульових степенів простих, більших p_j, а від заміни, яка перетворює a^* в a', такі степені не з’являться. Тому, $(a', \tau(a'))$ теж задовольняє умови для $S(N, j)$. В поєднанні з $(a', \tau(a')) \succ (a^*, \tau(a^*))$, маємо протиріччя з $(a^*, \tau(a^*)) \in S(N, j)$, і пояснити це протиріччя можна лише хибністю припущення про існування такого a^*. ■

Примітка до твердження 15. Ці твердження й доведення дуже схожі на твердж. 2 і його доведення, але у твердж. 2 йшлося про надскладені, а тут про елементи $S(N, j)$.

Твердження 16. Для кожного скінченного N, існує деяке (залежне від N) скінченне $m_{\max}$, таке, що $S(N, m_{\max}) = S(N, m_{\max}+1) = \ldots = S(N, \infty)$.

Доведення. За алгоритмом побудови $S(N, j+1)$ з твердж. 14, ні за яких N та j $S(N, j+1)$ не може бути власною підмножиною $S(N, j)$: пару можуть вилучити з S_{next} лише в п. 2.1.1.2.1, лише після включення в п. 2.1.1.1 нової домінуючої пари. Однак, з урахуванням твердж. 15, для нової пари має виконуватися $k_0 \ge k_1 \ge \ldots \ge k_j \ge k_{j+1} \ge 1$, а також $p_0^{k_0} \cdot p_1^{k_1} \cdot \ldots \cdot p_j^{k_j} \cdot p_{j+1}^{k_{j+1}} \le N$ (за означенням $S(N, j+1)$). Якщо (аналогічно доведенню насл. 1 твердж. 2 з [6]) перебирати добутки $2 \cdot 3$, потім $2 \cdot 3 \cdot 5$, потім $2 \cdot 3 \cdot 5 \cdot 7$, …, доки добуток не перевищить N, то те значення $j = j^*$, при якому вперше $\prod_{i=0}^{j+1} p_i > N$, можна взяти за $m_{\max}$. Тому що при такому j^* (а також всіх більших) не може бути водночас $k_0 \ge k_1 \ge \ldots \ge k_{j^*} \ge k_{j^*+1} \ge 1$ та $p_0^{k_0} \cdot p_1^{k_1} \cdot \ldots \cdot p_{j^*}^{k_{j^*}} \cdot p_{j^*+1}^{k_{j^*+1}} \le N$ (навіть $p_0^1 \cdot p_1^1 \cdot \ldots \cdot p_{j^*}^1 \cdot p_{j^*+1}^1 > N$, а від збільшення показників добуток може лише збільшитися). ■

Таке $m_{\max}$ є лише оцінкою згори; як правило, відмінності між $S(N, j+1)$ та $S(N, j)$ зникають раніше, бо при досить великих N та j множині $S(N, j)$ належать переважно пари, де кілька початкових $k_0, k_1, \ldots$ більші 1. Теоретично виразити це точніше не здається можливим. Тому, практично доречніше спиратися на наступне твердж. 17; але с*а*ме завдяки твердж. 16 відомо, що цей момент настане, причому досить швидко.

Твердження 17. Коли у процесі знаходження $S(N, 0)$, $S(N, 1)$, … вперше настане ситуація $S(N, j+1) = S(N, j)$, надалі буде $S(N, j) = S(N, j+1) = S(N, j+2) = \ldots = S(N, \infty)$, тож далі цю послідовність можна не будувати.

Доведення. В усіх парах, що належать $S(N, j)$, маємо $k_{j+1} = 0$ (в $S(N, j)$ дійшли до p_j, але не до p_{j+1}); $S(N, j+1) = S(N, j)$ означає, що k_{j+1} ні в якої пари не змінилося, тобто теж дорівнює 0. Отже, $S(N, j+2)$, $S(N, j+3)$, … будуватимуться з пар, де $k_{j+1} = 0$; враховуючи монотонність (твердж. 15), це означає $k_{j+1} = k_{j+2} = \ldots = 0$, тобто нових пар не буде. ■

Примітка 1 до твердження 17. Важливо, щоб однаковими були самі множ*и*ни пар $S(N, j+1)$ та $S(N, j)$, не лише кількості пар у них. Однаковість кількості не виключає ситуації, коли значення пар інші, й це впливає на остаточну відповідь.

Примітка 2 до твердження 17. Теоретично зручно говорити про «перевірку, чи $S(N, j+1) = S(N, j)$», але, раз побудова $S(N, j+1)$ починається з $S_{\text{next}} := S(N, j)$, технічно зручніше мати окрему змінну-прапорець, де відслідковувати, чи була множина змінена.

Твердження 18. Якщо задачу (у постановці № 1) треба розв’язати для кількох (як правило, різних) N_1, N_2, …, N_q, достатньо знайти $N_{\max} = \max(N_1, N_2, \ldots, N_q)$, побудувати один раз множ*и*ни пар $S(N_{\max}, 0)$, $S(N_{\max}, 1)$, …, $S(N_{\max}, j^*)$, $S(N_{\max}, j^*+1)$ (де j^* — те j, при якому вперше досягається $S(N_{\max}, j) = S(N_{\max}, j+1)$), після чого для кожного N_1, N_2, …, N_q знайти в $S(N_{\max}, j^*)$ ту пару, перший елемент якої є найближчим знизу до відповідного N_1, N_2, …, N_q, й це будуть шукані відповіді.

Доведення. $S(N_{\max}, j^*)$ містить усю інформацію, потрібну для знаходження суперскладених з діапазону від 1 до $N_{\max}$, а жодне з чисел $N_1, N_2, \ldots, N_q$ не перевищує $N_{\max}$. ■

Основна ідея технічного подання множин недомінованих пар. Розглянемо впорядковану словникову структуру даних, що підтримує за час $O(\log(size))$ вставку пари (ключ, значення), вилучення пари та пошук пари за ключем (як за точним збігом ключа, так і за запитами «найближчий згори/знизу»). Такими є, зокрема, `map` з C++ STL та `TreeMap` з Java collections. Оскільки пристосування до цієї задачі `map` чи `TreeMap` водночас і нетривіальне, і значно простіше за цілком власну реалізацію спеціалізованої структури даних, а технічні деталі дещо відрізняються навіть для `map` та `TreeMap`, далі наведено опис пристосування до задачі C++ STL `map`. Однак, це лише одна з можливих реалізацій. Більш того, ці ідеї частково спираються на роботу Ф. Препарати [8], написану до появи C++ STL чи Java. Крім того, автор статті дякує М. Рибаку за опис схожої ідеї (в застосуванні до іншої задачі) під час аналізу задач II етапу AUCPC-2007.

Будемо тримати множини недомінованих пар (як-то $S(N, j)$, S_{next} з алгоритму з твердж. 14, інші аналогічні) кожну в своєму примірнику `map`, де ключами є числа a, супутніми значеннями кількості їхніх дільників $\tau(a)$.

Твердження 19. Нехай пара $(b, \tau(b))$ подана окремими змінними `long long b` та `int tauB`, а `map`-ом, у якому подана S_{next}, є `map<long long, int> Snext`. Тоді п. 2.1.1 алгоритму з твердж. 14 «`if` пара $(b, \tau(b))$ не домінована жодною з пар, вже наявних у S_{next}», можна реалізувати такими діями (13):

```
auto it = Snext.lower_bound(b);
it--;                                                    (13)
if(it->second < tauB) {... (вкладені підпункти) ...}
```

Асимптотична складність цього фрагменту (без вкладених підпунктів) становить $O(\log T)$, де T = `Snext.size()` (тут і в подальших твердженнях).

Доведення. «`Snext.lower_bound(b)`» знаходить ітератор пари, першої (з найменшим ключем) серед тих, де `it->first` >= `b`; потім, «`it--`» переходить до попередньої пари, ключ якої максимальний серед строго менших b. (Метод `lower_bound` не може вертати значення `Snext.begin()`, бо всі `b` більші 1, а `Snext` містить пару (1, 1), яка кладеться у $S(N, 0)$, копіюється в подальші $S(N, 1)$, $S(N, 2)$, …, й ніколи не вилучається (бо ніщо не домінує її); тому, після `it--` ітератор все ще валідний. Випадок, що `b` перевищить найбільший з наявних у `Snext` ключів і `Snext.lower_bound(b)` поверне значення `Snext.end()`, можливий; але `it--` працює з цим коректно, окремого розгалуження не треба.) Якщо при цьому умова «`it->second < tauB`» хибна, тобто `it->second >= tauB`, то `(it->first, it->second)` $\succ$ `(b, tauB)`, тобто нова пара `(b, tauB)` домінована і її слід пропустити (що буде зроблено, бо `else`-гілки нема).

Ще треба показати, що при істинності «`it->second < tauB`» пару `(b, tauB)` не домінує не лише конкретно пара `(it->first, it->second)`, а й ніяка інша з наявних у `Snext`. Тут важливо, що `Snext` відсортована за зростанням як `first`-ів (бо така властивість C++ STL `map`), так і `second`-ів (за твердж. 12 та прим. 3 до твердж. 14), тож якщо `it->second < tauB`, то пари, ключі яких ще менші, мають ще менші `second`-и, й тому не домінують `(b, tauB)`. Пари, ключі яких більші b, теж не домінують `(b, tauB)`.

Як відомо, кожна з описаних у (13) дій (не враховуючи внутрішність фігурних дужок) працює за $O(\log T)$ дій, де T = `Snext.size()`. ■

Твердження 20. П. 2.1.1.1 алгоритму з твердж. 14 «включити $(b, \tau(b))$ у S_{next}» можна реалізувати очевидним кодом (14):

```
Snext[b] = tauB;                                   (14)
```

Це, очевидно, має асимптотичну складність $O(\log T)$, де $T =$ `Snext.size()`.

Твердження 21. П. 2.1.1.2 алгоритму з твердж. 14 «`if((`$b, \tau(b)$`)` домінує хоча б одну з пар, вже наявних у S_{next}`)`» можна реалізувати таким кодом (15):

```
auto it = Snext.upper_bound(b);
if(it != Snext.end()  && it->second <= tauB) {… … …}      (15)
```

При цьому, асимптотична складність (без вкладених підпунктів) становитиме $O(\log T)$.

Доведення. `Snext.upper_bound(b)` шукає пару з мінімальним серед більших b ключем. Істинність умови `if`-а виражає, що така пара існує, причому кіль-ть дільників $\leq$ `tauB`. Тобто, (`b`, `tauB`)$\succ$(`it->first`, `it->second`).

Лишилося перевірити, що за хибності умови `if`-а пара $(b, \tau(b))$ не домінує жодну з пар, вже наявних у S_{next}. Випадок, коли `upper_bound` повертає значення `Snext.end()`, можливий (якщо щойно додана пара (`b`, `tauB`) має найбільший у `Snext` ключ), і в цьому випадку така пара не може домінувати ніяку з пар, вже наявних у S_{next} до (14). Отже, «`it != Snext.end()`» і уникає звернення до невалідного ітератора, і правильно за смислом. Якщо ж причина хибності інша (`it != Snext.end()  && it->second > tauB`), то, враховуючи монотонність `second`-ів, раз у найближчої згори пари більше дільників, у подальших їх ще більше, і (`b`, `tauB`) не може домінувати жодну з них. Також, (`b`, `tauB`) не може домінувати й пари, ключі яких менші b.

Виклик `upper_bound` працює за $O(\log T)$, а решта використаних дій за $\Theta(1)$. ■

Твердження 22. П. 2.1.1.2.1 алгоритму з твердж. 14 «вилучити з S_{next} усі пари, доміновані парою $(b, \tau(b))$» можна реалізувати таким кодом (16):

```
while(it != Snext.end()  &&  it->second <= tauB) {
    it = Snext.erase(it); /* вилучає пару ітератора it */
    /* і переставляє it на пару, яка до вилучення була наступною */      (16)
}
```

де початковим значенням `it` є знайдене у (15). Асимптотична оцінка часу виконання цього коду становить гарантовані $O(X \cdot \log T)$, де X — кількість домінованих пар, які будуть вилучені цим циклом, $T =$ `Snext.size()`. Амортизована ж оцінка (смисл поняття див. [5, розд. 17]) при цьому становить $O(\log T)$, тобто не залежить від X.

Доведення. Поки умова у `while` виконується, щоразу має місце аргументація з аналізу фрагмента (15) про те, чому (`b`, `tauB`)$\succ$(`it->first`, `it->second`). Ось цей код і вилучає таку доміновану пару. Коли умова перестає виконуватися, той же аналіз фрагмента (15) пояснює, чому `Snext` вже не містить пар, домінованих парою (`b`, `tauB`).

Кожен виклик `Snext.erase(it)` вкладається в $O(\log T)$, тому, X вилучень гарантовано вкладаються у $O(X \cdot \log T)$.

Конкретну пару неможливо вилучити більше одного разу (після вилучення її вже нема в S_{next}), тож сумарна за весь алгоритм кількість вилучень у п. 2.1.1.2.1 менша суми кількостей пар, скопійованих у п. 1, та пар, вставлених у п. 2.1.1.1. У свою чергу, пари, скопійовані в п. 1 при поточній побудові $S(N, j{+}1)$, не вилучені з S_{next} при попередній побудові $S(N, j)$. Тому, сумарна кількість усіх вилучень відрізняється від сумарної кількості всіх вставок лише на $|S(N, j^{*}{+}1)| - |S(N, 0)|$, чим асимптотичний аналіз нехтує у порівнянні, скажімо, з кількістю ітерацій п. 2. Тому, в середньому за весь алгоритм

$X \approx 1$ (як частка кількості всіх вилучень і кількості всіх вставок, які майже однакові). ■

Деякі дрібні оптимізації (14)-(16). Оскільки (15) викликається зразу після (14), можна замінити у (14) `operator[]` на `insert` і оптимізувати (15), замінивши `upper_bound` (складності $O(\log T)$) на ++ (складності $\Theta(1)$), ось так:

```
auto it = (Snext.insert(make_pair(b, tauB)).second)++;
```

(Цей рядок замінює собою і (14), і перший з рядків (15).) Але асимптотика алгоритму в цілому від того не змінюється, а зрозумілість коду погіршується, тому це недоцільно.

Ще, `if`, яким завершується (15), та `while`, яким починається (16), мають однакові умови, тому можна прибрати `if`, бо `while` робить цю перевірку зокрема й на початку циклу. Ця оптимізація більш-менш доречна (але неістотна). □

На жаль, не здається можливим оцінити аналітично кількість $T =$ `Snext.size()`, від якої залежить не лише асимптотична оцінка (13)–(16), а й кількість ітерацій п. 2.

Що ж, раз нема аналітичного вираження цих кількостей, спробуємо дослідити їх експериментально. Реалізацію цього алгоритму можна побачити за посиланням ejudge.ckipo.edu.ua/maxDivsNonDominated1.cpp.

Таблиця 4 — кількості пар у множинах $S(N, 0)$, $S(N, 1)$, … (при значеннях N у межах стандартних цілочисельних типів).

N \ j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	…	19
p_j	2	3	5	7	11	13	17	19	23	29	31	37	41	43	…	71
10^3	10	15	14	15	15											
10^9	30	62	71	66	66	65	65	66	66							
10^{18}	60	156	198	198	194	184	175	171	163	159	156	156	156			
10^{36}	120	389	553	624	627	595	581	537	527	497	489	493	467	441	…	368

Таблиця 5 — експериментально визначені кількісні характеристики множин недомінованих пар (при великих N, з «довгою» арифметикою).

N	10^{20}	10^{50}	10^{100}	10^{200}	10^{500}	10^{1000}
j^* (min, при якому $S(N, j^*) = S(N, j^*+1)$)	12	26	47	84	184	334
p_{j^*}	41	103	223	439	1 103	2 251
$\max_{0 \le j \le j^*+1} \lvert S(N, j) \rvert$	242	1 093	3 511	11 494	56 344	191 555
$\sum_{j=0}^{j^*+1} \lvert S(N, j) \rvert$	2 688	20 703	99 269	476 839	3 733 353	17 553 112
Час (секунд)	0,015	0,156	1,544	15,834	418,316	5 483,411
Час (с) алгоритму 1 з [6]	0,046	32,323	31 647,563	Більший доби, далі не досліжувався		

Перша спроба такого експерименту описана в табл. 4. Рядки цієї таблиці виражають, для якого N будують множ*и*ни недомінованих пар. Стовпчики — номери j простих чисел p_j та відповідних їм $S(N, j)$. Елементи основної частини таблиці означають кількості елементів-пар у відповідній $S(N, j)$. Порожні клітинки виражають, що, згідно твердж. 17, при поточних N та j нема сенсу будувати $S(N, j)$.

Розміри всіх $S(N, j)$ з табл. 4 (навіть для $N = 10^{36}$) досить малі, а виконання майже миттєве. Тому, для вимірювань при ще більших N, алгоритм переписано з використанням довгої арифметики (реалізованої самостійно). Результати див. у табл. 5.

Стовпчики табл. 5 позначають значення N, для яких проведено побудову $S(N, 0)$, $S(N, 1)$, …. Для кожного з цих N окремо, перелік множин $S(N, 0)$, $S(N, 1)$, … будувався «до кінця», тобто доки не наставала ситуація $S(N, j^*) = S(N, j^*{+}1)$. Рядки табл. 5 виражають, при якому (за номером j^* та значенням p_{j^*}) простому це наставало, якими були максимальна та сумарна кількість елементів-пар у побудованих множинах $S(N, 0)$, $S(N, 1)$, …, $S(N, j^*{+}1)$, та який час займало обчислення (тактова частота комп'ютера 3,2 ГГц, компілятор g++ (Windows), рівень оптимізацій «–O2», точність вимірювання 1 тік ≈ 0,015 с, усереднене з двох спроб, тобто ті самі, що у [6]). Для повноти порівняння, переписаний (з цією ж реалізацією довгої арифметики) також і алгоритм 1 з [6]. Час виконання нового алгоритму значно менший (при тому, що він знаходить усі надскладені проміжку, а алгоритм з [6] знаходив лише максимальне).

Модифікація алгоритму для постановки № 3а. Аналогічно [6], тут теж досить викреслити K з послідовності простих, і теж час роботи при цьому може або зменшитися, або лишитися незмінним.

Модифікація алгоритму для постановки № 3б. На відміну від [6] (де розгляд складених K потребував великої кількості дивних модифікацій основного алгоритму), тепер істотних модифікацій основного алгоритму менше й вони природніші.

Спочатку варто сформулювати кілька випадків, коли відповідь постановки № 3б можна отримати, розв'язавши задачу в постановці № 1. Ці випадки покривають лише малу частину можливих вхідних даних, але корисні не лише тим, що дозволяють отримати відповідь для цих вхідних даних, а й тим, що дозволяють проводити теоретичні міркування. Твердж. 23–25 очевидні й наводяться без доведень.

Твердження 23. Якщо для деяких N, K задачу розв'язали алгоритмом для постановки № 1 (з тим самим N, проігнорувавши K), і відповідь-значення виявилася не кратною K, то повна відповідь є правильною також і для постановки № 3б.

Твердження 24. Якщо $K > N$, то повна відповідь для постановки № 3б дорівнює відповіді для постановки № 1 (з тим самим N, проігнорувавши K).

Твердження 25. Якщо $K = N$, то повна відповідь для постановки № 3б дорівнює відповіді для постановки № 1 (взявши $N-1$ замість N, проігнорувавши K).

Твердження 26. Нехай $i' = \max\{\, i \mid z_i > 0 \}$ (індекс останнього простого, наявного в розкладенні K у ненульовій степені), j^* є числом, для якого $S(N, j^*) = S(N, j^*{+}1) = S(N, j^*{+}2) = \ldots = S(N, \infty)$, і при цьому $j^* < i'$. Тоді відповідь для постановки № 3б дорівнює відповіді для постановки № 1 (з тим самим N, ігноруючи K).

Доведення. З частини «нехай» випливає, що $S(N, \infty) = S(N, j^*)$ при $j^* < i'$; тобто, в усіх парах з $S(N, \infty)$ перший елемент не кратний $p_{i'}$, а отже й не кратний K. Тому, відповідь-значення для постановки № 1 не кратна K, отже застосовне твердж. 23. ■

Означення 8. Позначимо за $R(N, j, K)$ (де N — натуральне число, j — ціле невід'ємне число, K — натуральне число, причому $K \geq 2$) множину всіх пар $(v, \tau(v))$, недомінованих за умови «$1 \leq v \leq N$, в розкладенні v є лише прості $p_0 = 2$, $p_1 = 3$, …, p_j, та забезпечено, що ні саме v, ні результати його домножень на які б не було степені подальших простих p_{j+1}, p_{j+2}, … не можуть бути кратними K».

Примітка до означення 8. Тут стверджується, що для $(v, \tau(v)) \in R(N, j, K)$ треба $v \leq N$, але для рішення «$(v, \tau(v)) \notin R(N, j, K)$, бо результат домноження v на деякі степені

p_{j+1}, p_{j+2}, … кратний K» неважливо, чи добуток, кратний K, буде меншим, рівним чи більшим N. Наприклад, (4, 3) та (48, 10) однаково належать $S(200, 1)$, але не належать $R(200, 1, 100)$, бо $4 \times 25 = 100 = 1 \times 100$, а $48 \times 25 = 1200 = 12 \times 100$. І тут неважливо, що $100 < 200$, а $1200 > 200$; важливо лише, що і 100, і 1200 кратні 100. □

Озн. 8 може здатися неконструктивним (як перевіряти кратність для відразу всіх результатів домножень v на які б не було степені подальших простих?). Однак, твердж. 27 (очевидне, тому без доведення) та твердж. 28 вирішують цю проблему.

Твердження 27. При натуральних a, b вигляду $a = p_0^{a_0} \cdot p_1^{a_1} \cdot \ldots$ та $b = p_0^{b_0} \cdot p_1^{b_1} \cdot \ldots$, кратність $a \vdots b$ має місце тоді й тільки тоді, коли $\forall i\, (a_i \geq b_i)$.

Твердження 28. Ситуація з озн. 8 «ні саме $v = p_0^{k_0} \cdot p_1^{k_1} \cdot \ldots \cdot p_j^{k_j}$, ні результати його домножень на які б не було степені подальших простих p_{j+1}, p_{j+2}, … не можуть бути кратними $K = p_0^{z_0} \cdot p_1^{z_1} \cdot \ldots$» настає тоді й тільки тоді, коли

$$\exists i^*{}_{0 \leq i^* \leq j}\, (k_{i^*} < z_{i^*}). \qquad (17)$$

Доведення. Якщо (17) виконується, то, на які б степені p_{j+1}, p_{j+2}, … не домножали v (тобто, як би не збільшували k_{j+1}, k_{j+2}, …), це не збільшить те k_{i^*} (при $0 \leq i^* \leq j$), при якому $k_{i^*} < z_{*i}$; отже, домножене v не стане кратним K. Якщо ж (17) не виконується, тобто $\forall i_{\,0 \leq i \leq j}\, (k_i \geq z_i)$, то в межах $0 \leq i \leq j$ виконується рівно те, що треба згідно твердж. 27, і для кратності $v \vdots K$ лишається забезпечити, щоб $k_i \geq z_i$ виконалося також і для $i = j+1$, $j+2$, …, а це забезпечити дуже легко (наприклад, взяти $k_{j+1} = z_{j+1}$, $k_{j+2} = z_{j+2}$, …). ■

Тепер варто сформулювати й довести твердження 29 та 30, які будуть аналогами тверджень 13 та 14, тільки для $R(N, j, K)$ замість $S(N, j)$.

Твердження 29. $R(N, 0, K)$ складається з пар $\{(1, 1), (2, 2), (4, 3), \ldots, (2^x, x+1)\}$, де $x = \min(\lfloor \log_2 N \rfloor, z_0 - 1)$, а z_0, згідно озн. 3, є показником $p_0 = 2$ в розкладенні K. Зокрема, при $z_0 = 0$ (будь-якому непарному K) та будь-якому N, маємо $R(N, 0, K) = \varnothing$.

Доведення. З одного боку, потрібно задовольнити умову, що v є степенем двійки (містить лише просте $p_0 = 2$), тому $R(N, 0, K)$ не може містити інших пар, крім (згаданих у твердженні 13) пар (1, 1), (2, 2), (4, 3), …, $(2^{\lfloor \log_2 N \rfloor}, \lfloor \log_2 N \rfloor + 1)$. З іншого боку, коли дозволене лиш одне просте p_0, формула (17) з твердж. 28 спрощується до $k_0 < z_0$. Щоб задовольнити обидві вимоги одночасно, слід вибрати мінімум.

Також, при $z_0 = 0$ не існує жодного значення показника k_0, яке було б водночас і цілим невід'ємним (це потрібно для будь-якого розкладення), і меншим z_0. ■

Наслідок з твердження 29. Якщо $K < N$, останньою парою $R(N, 0, K)$ є $(2^{z_0 - 1}, z_0)$.

Доведення. Оскільки $z_0 - 1$ є одним з аргументів мінімума, досить довести, що з $K < N$ випливає $z_0 - 1 \leq \lfloor \log_2 N \rfloor$. Припустимо протилежне, тобто $\lfloor \log_2 N \rfloor < z_0 - 1$. Розглянемо очевидну нерівність $a - 1 < \lfloor a \rfloor$ і підставимо $\log_2 N$ в якості a; вийде $(\log_2 N) - 1 < \lfloor \log_2 N \rfloor$. Тоді з припущення випливає $(\log_2 N) - 1 < z_0 - 1$, тобто $\log_2 N < z_0$, тобто $N < 2^{z_0}$. Оскільки K дорівнює добутку $2^{z_0} = p_0^{z_0}$ на $p_1^{z_1} \cdot p_2^{z_2} \cdot \ldots$, це означає $N < K$, що протирічить $K < N$. Отже, припущення неправильне, й $x = z_0 - 1$. ■

Твердження 30, скорочене формулювання. Щоб побудувати $R(N, j+1, K)$, слід, по-перше, застосувати такі ж дії, як у твердж. 14 до $S(N, j)$, але до $R(N, j, K)$; якщо $z_{j+1} > 0$, то, по-друге, слід застосувати також аналогічні дії до $S(N, j)$, але з поправками: при домноженнях пар починати не з множника p_{j+1}, а з $p_{j+1}^0 = 1$, та закінчувати, враховуючи як вимогу «домножене число не перевищує N», так і вимогу «$k_{j+1} < z_{j+1}$».

Твердження 30, детальне формулювання. Нехай вже відомі множини пар $R(N, j, K)$ та $S(N, j)$, і над ними виконують такі дії:

1. $R_{\text{next}} := R(N, j, K)$, тобто спочатку нова множина R_{next} складається з усіх пар, що є елементами $R(N, j, K)$, і лише з них.
2. Для кожної з пар $(a, \tau(a))$, що належать $R(N, j, K)$, виконати такі дії:
   2.1. перебрати пари $(a \cdot p_{j+1}, \tau(a) \cdot 2)$, $(a \cdot p_{j+1}^2, \tau(a) \cdot 3)$, …, $(a \cdot p_{j+1}^s, \tau(a) \cdot (s+1))$ (де s можна виразити як $\lfloor \log_{p_{j+1}}(N/a) \rfloor$ чи підібрати циклом `while`), тобто всі пари, де перший елемент не перевищує N і є добутком a на деякий степінь простого p_{j+1}; позначивши ці пари вигляду $(a \cdot p_{j+1}^i, \tau(a) \cdot (i+1))$ як $(b, \tau(b))$, для кожної з них виконати такі дії:
      2.1.1. `if` ($(b, \tau(b))$ не домінована жодною з пар, вже наявних у R_{next}) `{`
         2.1.1.1. включити $(b, \tau(b))$ у R_{next};
         2.1.1.2. `if` ($(b, \tau(b))$ домінує хоча б одну з пар, вже наявних у R_{next})
            2.1.1.2.1. вилучити з R_{next} усі пари, доміновані парою $(b, \tau(b))$;
         `}`
3. Якщо $z_{j+1} > 0$, то:
   3.1. Для кожної з пар $(a, \tau(a))$, що належать $S(N, j)$, виконати такі дії:
      3.1.1. перебрати пари $(a, \tau(a))$, $(a \cdot p_{j+1}, \tau(a) \cdot 2)$, $(a \cdot p_{j+1}^2, \tau(a) \cdot 3)$, …, $(a \cdot p_{j+1}^s, \tau(a) \cdot (s+1))$ (де s можна підібрати циклом `while` чи виразити як $\min(z_{j+1} - 1, \lfloor \log_{p_{j+1}}(N/a) \rfloor)$), тобто всі пари, де перший елемент є добутком a на деякий (можливо, 0-й) степінь простого p_{j+1}, і при цьому добуток не перевищує N, а показник не перевищує $z_{j+1} - 1$; позначивши ці пари вигляду $(a \cdot p_{j+1}^i, \tau(a) \cdot (i+1))$ як $(b, \tau(b))$, для кожної з них виконати такі дії:
         3.1.1.1. `if` ($(b, \tau(b))$ не домінована жодною з пар, вже наявних у R_{next}) `{`
            3.1.1.1.1. включити $(b, \tau(b))$ у R_{next};
            3.1.1.1.2. `if` ($(b, \tau(b))$ домінує хоча б одну з пар, вже наявних у R_{next})
               3.1.1.1.2.1. вилучити з R_{next} усі пари, доміновані парою $(b, \tau(b))$;
            `}`

Тоді утворена цими діями множина R_{next} дорівнює $R(N, j+1, K)$.

Ідея доведення. Доведення в цілому аналогічне доведенню твердж. 14 (алгоритм за деяких обставин додає та/або вилучає інші пари, але аргументація, чому додані пари справді слід додати, а вилучені справді слід вилучити, аналогічна.) Головне, що потребує нових міркувань — використання при побудові $R(N, j+1, K)$ як $R(N, j, K)$, так і $S(N, j)$, та потреба забезпечити умову «результати домножень першого елемента на які б не було степені подальших простих p_{j+1}, p_{j+2}, … не будуть кратними K». Для всіх пар з $R(N, j, K)$ ця некратність вже забезпечена, тож домноження цих пар на який би не було степінь простого p_{j+1} не можуть її порушити, й можна діяти так само, як у алгоритмі з твердж. 14. При використанні ж пар з $S(N, j)$ некратність забезпечується тим, що додаткові множники обмежені, щоб забезпечити $k_{j+1} < z_{j+1}$. □

Примітки. Для твердж. 30 дійсні аналоги всіх приміток до твердж. 14. □

При реалізації алгоритму з твердження 30 можна взяти за основу фрагменти коду (13)–(16), але вони потребують деяких правок. Наприклад, в аналізі (13) сказано, що `Snext.upper_bound(b)` не може вертати значення `Snext.begin()`, але тепер, в аналогічному коді для п. 3.1.1.1, при $R(N, j, K) = \varnothing$ `Rnext.upper_bound(b)` може вертати значення `Rnext.begin()`. Також, тепер треба переконатися в правильності обробки ситуації, коли одна й та сама пара (`b`, `tauB`) перевіряється як у п. 2.1.1, так і в п. 3.1.1.1 (в алгоритмі з твердж. 14 основна теорема арифметики про єдність розкладення робила таку ситуацію неможливою). Для скорочення обсягу, такі деталі пропущені. Охочі можуть прочитати код за посиланням ejudge.ckipo.edu.ua/…

Ще слід з'ясувати, до яких пір треба повторювати перетворення $R(N, j, K)$ та $S(N, j)$ у $R(N, j+1, K)$. Використовувати тут незмінне твердж. 17 неправильно, бо деякі з використаних у доведеннях твердж. 15–17 властивості для постановки № 3б не виконуються. Однак, можна сформулювати й довести твердж. 31–33, котрі є правильними для постановки № 3б аналогами тверджень 15–17.

Твердження 31. Для будь-якого $j \geq 0$, будь-якого $N \geq 1$ та будь-якого $K \geq 2$, для кожної пари $(a, \tau(a)) \in R(N, j, K)$, якщо розкласти перший елемент цієї пари як $a = p_0^{k_0} \cdot p_1^{k_1} \cdot \ldots \cdot p_j^{k_j} \cdot \ldots$, а потім викреслити з цього розкладення степені тих простих, для яких $z_i > 0$, то послідовність тих k_i, що залишаться, буде монотонно незростаючою.

Доведення. Самі твердж. 15 і його доведення непридатні, бо від заміни $p_u^{k_u} \cdot p_w^{k_w}$ на $p_u^{k_w} \cdot p_w^{k_u}$ число може стати кратним K, не бувши кратним до заміни. Однак, невикресленими лишаються тільки такі u та w, для яких $z_u = z_w = 0$, а для них ця заміна не впливає на кратність K, тож для них можна повторити міркування доведення твердж. 15. ■

Твердження 32. Для кожного скінченного N та кожного скінченого K, існує деяке (залежне від N та K) скінченне $m_{\max}$, таке, що, починаючи з нього, множини недомінованих пар рівні: $R(N, m_{\max}, K) = R(N, m_{\max}+1, K) = \ldots = R(N, \infty, K)$.

Доведення. Проведемо міркування, аналогічні доведенню твердж. 16, але лише для тих простих, де $z_i = 0$ (завдяки твердж. 31 відомо, що серед них має місце монотонність, тому для них міркувати аналогічно можна). ■

Твердження 33. Нехай $i' = \max\{\, i \mid z_i > 0 \,\}$ (індекс останнього простого, наявного в розкладенні K у ненульовій степені). Якщо побудувати послідовність $R(N, 0, K)$, $R(N, 1, K)$, … щонайменше до $R(N, i', K)$ включно, а вже серед $j \geq i'$ почати перевірки, чи $R(N, j+1, K) = R(N, j, K)$, то після першої ж такої рівності буде $R(N, j, K) = R(N, j+1, K) = R(N, j+2, K) = \ldots = R(N, \infty, K)$, тож далі цю послідовність можна не продовжувати.

Доведення. З обмеження $j \geq i'$ випливають такі наслідки: (1) для них $R(N, j+1, K)$ будуються на основі лише $R(N, j, K)$, без використання $S(N, j)$; (2) встановлена у твердж. 31 монотонність тих з k_j, де $z_j = 0$, при $j \geq i'$ стосується всіх підряд таких j. Тому, в цих умовах можна повторити міркування, аналогічні доведенню твердж. 17. ■

Примітки 1–2 до твердження 33. Аналогічні приміткам до твердж. 17.

Примітка 3 до твердження 33. Не можна відкидати вимогу $j \geq i'$ і стверджувати, ніби «як тільки вперше настало $R(N, j+1, K) = R(N, j, K)$, так далі всі множини будуть рівними». Хоча б тому, що при $z_{j+1} > 0$ множина $R(N, j+1, K)$ залежить не лише від

$R(N, j, K)$, а й від $S(N, j)$. Це можна підтвердити й прикладом: $R(100, 0, 35) = R(100, 1, 35) = \varnothing$, але $R(100, 2, 35) = S(100, 1) \neq \varnothing$.

Примітка 4 до твердження 33. Практично доцільно мати **дві** умови завершення алгоритму: одна — на основі цього твердж. 33; інша — на основі поєднання тверджень 17 та 26. Тобто: якщо $S(N, j^*+1) = S(N, j^*)$ (твердж. 17) настає раніше, чим i' (останній з показників $z_i > 0$), то отримати відповідь з $S(N, j^*)$; якщо, навпаки, i' настає раніше, то скористатися поточним твердж. 33.

Примітка 5 до твердження 33. Якщо $j \geq i'$ (тобто, побудова $S(N, 0)$, $R(N, 0, K)$, $S(N, 1)$, $R(N, 1, K)$, … вже перейшла через i'), а умова $S(N, j^*+1) = S(N, j^*)$ (з твердження 17) все ще не настала, практично доцільно припиняти побудову $S(N, j^*)$, бо всі подальші $R(N, j+1, K)$ все одно вже не залежатимуть від $S(N, j)$. □

Таблиця 6 — експериментально визначені кількісні характеристики множин недомінованих пар (при великих N, складених K).

	N	10^{20}	10^{50}	10^{100}	10^{200}	10^{500}	10^{1000}
$K = 40 = 2^3 \times 5$	Max $\|S\|, \|R\|$	179	652	2 217	7 724	40 144	139 988
	Sum $\|S\|, \|R\|$	1 845	12 861	62 555	305 569	2 473 648	12 130 863
	Час (с)	0,101	0,226	1,005	9,492	259,265	3 601,578
$K = 829635693357621875 = 5^5 \times 17^4 \times 29^3 \times 47^2 \times 59$	Max $\|S\|, \|R\|$	242	1 093	3 511	11 494	56 344	191 555
	Sum $\|S\|, \|R\|$	4 993	32 210	136 503	599 747	4 313 155	19 441 352
	Час (с)	0,108	0,350	2,363	21,902	533,569	6 598,116
$K = 267727733530800000 = 2^7 \times 3^6 \times 5^5 \times 7^4 \times 11^3 \times 13^2 \times 17$	Max $\|S\|, \|R\|$	242	1 093	3 511	11 494	55 033	180 375
	Sum $\|S\|, \|R\|$	3 654	23 985	106 048	473 878	3 582 036	16 601 348
	Час (с)	0,031	0,288	1,793	16,348	405,571	5 259,354
$K = 235920057993400000 = 2^6 \times 5^5 \times 11^4 \times 17^3 \times 23^2 \times 31$	Max $\|S\|, \|R\|$	242	1 093	3 511	11 494	56 344	191 555
	Sum $\|S\|, \|R\|$	4 377	27 733	120 255	539 377	3 981 625	18 053 569
	Час (с)	0,101	0,343	2,121	19,406	477,877	5 979,104
$K = 281732026857788708 9 = 37^4 \times 83^3 \times 127^2 \times 163$	Max $\|S\|, \|R\|$	242	1 093	3 511	11 494	56 344	191 555
	Sum $\|S\|, \|R\|$	3 219	31 879	150 667	654 420	4 716 919	21 068 885
	Час (с)	0,093	0,318	2,363	21,551	539,193	6 907,099
$K = 97033579375 = 5^4 \times 11^3 \times 31^2 \times 997$	Max $\|S\|, \|R\|$	242	1 093	3 511	11 494	56 344	191 555
	Sum $\|S\|, \|R\|$	5 002	39 792	192 637	931 964	7 146 338	29 476 297
	Час (с)	0,101	0,350	3,221	30,980	798,833	9 565,622
$K = 321389516527343410 = 2 \times 5 \times 1979 \times 1993 \times 1999 \times 2011 \times 2027$	Max $\|S\|, \|R\|$	242	1 093	3 511	11 494	56 344	191 555
	Sum $\|S\|, \|R\|$	4 866	38 159	183 760	882 385	6 957 321	32 256 608
	Час (с)	0,015	0,312	2,854	29,452	790,792	10 178,284
$K = 2017 = 2017^1$	Max $\|S\|, \|R\|$	242	1 093	3 511	11 494	56 344	191 555
	Sum $\|S\|, \|R\|$	2 688	20 703	99 269	476 839	3 733 353	17 579 983
	Час (с)	0,008	0,164	1,575	16,208	422,504	5 526,303

Для аналізу того, як усе це вплинуло на час роботи, проведено експерименти, аналогічні тому, підсумки якого наведені в табл. 5, але при деяких K. Вміст табл. 6 аналогічний табл. 5, лише кількісні характеристики множин $S(N, j)$ замінено на характеристики фактично використаних $S(N, j)$ та $R(N, j, K)$ при деяких K. Параметри компіляції та комп'ютер використані ті самі, що для табл. 5. При побудові $R(N, j, K)$ та $S(N, j)$ використані обидві умови завершення циклу (і тверж. 17 та 26, і 33). Прим. 5 до твердж. 33 теж використана. Під «max $|S|$, $|R|$» мається на увазі максимальна серед кількостей пар деякої реально побудованої $S(N, j)$ чи $R(N, j, K)$, а під «sum $|S|$, $|R|$» — сумарна кількість пар усіх реально побудованих $S(N, j)$ та $R(N, j, K)$.

Бачимо, що максимальні та сумарні розміри $R(N, j, K)$ та $S(N, j)$, а також час роботи, не сильно відрізняються від відповідних (при тому ж N) розмірів $S(N, j)$, в межах інтуїтивно очікуваного «кілька разів». Випадки, коли вони виявляються навіть меншими, чим у табл. 5, можна пояснити тим, що завдяки прим. 5 до твердження 33 іноді можна не будувати значну частину $S(N, j)$; якщо при цьому $R(N, j, K)$ містять менше, чим $S(N, j)$, пар, то в таких випадках обсяг перебору зменшується.) Природно й те, що більшість величин виявилися мінімальними при $K = 40 = 2^3{\times}5$ (коли і виникли обмеження на степені $p_0 = 2$ та $p_2 = 5$ в $R(N, j, K)$, і дуже швидко припинили обчислювати $S(N, j)$), а максимальними при $K = 3213895165273434 10 = 2{\times}5{\times}1979{\times}1993{\times}1999{\times}{\times}2011{\times}2027$ (коли довелося дуже довго обчислювати і $S(N, j)$, і $R(N, j, K)$).

Перспективи продовження досліджень ідей цієї статті. Алгоритм дозволяє додаткові оптимізації, вплив яких ще не досліджений. Зокрема, такі.

1) Для постановки № 1 мало місце твердження 18, за яким можна було один раз побудувати серію множин $S(N_{\max}, 0)$, $S(N_{\max}, 1)$, …, і знаходити відповіді для всіх N_i. Очевидно, для постановки № 3б це так само, коли є різні N_i при однакових K, але коли є різні K_i, ситуація інакша, бо послідовність множин $R(N, 0, K)$, $R(N, 1, K)$, … будується для конкретного K. Однак, варто дослідити умови, за яких такі послідовності множин для різних K_i можна будувати, повторно використовуючи однакові проміжні частини. Наприклад, перша відмінність між $K_1 = 2^3{\times}5^2{\times}17 = 3400$ та $K_2 = 2^3{\times}5^2{\times}19 = 3800$ лише в z_6, відповідному $p_6 = 17$, а $z_0, z_1, \ldots, z_5$ однакові, тому й $R(N, 0, K)$, $R(N, 1, K)$, …, $R(N, 5, K)$ для $K_1 = 3400$ та $K_2 = 3800$ теж однакові (а з $R(N, 6, K)$ в разі великого N починається відмінність). Таким однаковим частинам сприяють також твердження 6 з [6] та його наслідок, які в деяких ситуаціях дозволяють перетворити різні K_i в однакові зменшені K_i'. Як краще організувати такі повторні використання, наскільки помітне вони можуть дати загальне прискорення і наскільки сильно заради них слід жертвувати економією пам’яті з прим. 1 до твердж. 30 — варто дослідити.

2) Твердж. 23–26 та прим. 4 до твердж. 33 згадують випадки, коли відповідь постановки № 3б можна отримати, застосувавши простіший алгоритм для постановки № 1; однак, цей перелік не є ні вичерпним, ні конструктивним. Можна спробувати дослідити глибше, коли варто пробувати розв’язати алгоритмом для постановки № 1, і лише якщо відповідь кратна K, то переходити до основного алгоритму для постановки № 3б, а коли відразу діяти за алгоритмом для постановки № 3б.

3) Було розглянуто реалізацію, де множини недомінованих пар подаються як C++ STL `map`, у якому ключами є ч*и*сла-значення, а `second`-ами — кількості дільників цих чисел. Однак, імовірно, що потрібні алгоритму властивості можна переформулювати й заново довести, організувавши `map` «навпаки», щоб ключами були кількості дільників, а відповідними супутніми значеннями мінімальне з чисел, яке має таку кількість дільників, потім переробити відповідно сам алгоритм. При цьому міркування стануть заплутанішими, та й програма, мабуть, ускладниться. Зате це може дати пришвидшення за рахунок того, що кількості дільників значно менші, чим ч*и*сла, які мають відповідну кількість дільників, а алгоритм робить багато викликів `upper_bound` / `lower_bound` (отже, багато порівнянь ключів). Для проміжків, де ч*и*сла-значення вже потребують довгої арифметики, а кількості дільників ще поміщаються у стандартні типи, слід сподіватися особливо помітного пришвидшення. Там, де кількості дільників теж потреб*у*ють довгої арифметики, слід очікувати меншого, але теж пришвидшення, бо час порівняння у довгій арифметиці залежить від довжин чисел.

Висновки

У цій статті запропоновано принципово інший, оснований на ідеї недомінованих пар, алгоритм вирішення тієї самої задачі, що в [6] — знаходження надскладених чисел, тобто тих, які мають рекордні кількості дільників, а також узагальнення цієї задачі, коли максимум шукається не серед усіх чисел проміжку, а лише серед не кратних деякому K. На відміну від алгоритму з [6], запропонований тут алгоритм не здається можливим узагальнити на випадок, коли проміжок починається не з 1, а з довільного $M \le N$. Однак, якщо розглядати лише проміжки від 1 до N, запропонований тут алгоритм ефективніший за розібрані в [6], а узагальнення для складених K природніше.

Список використаної літератури:

1. Стаття «Надскладене число» в українській вікіпедії. Режим доступу https://uk.wikipedia.org/wiki/Надскладене_число.
2. Бухштаб, А. А. Теория чисел. / А. А. Бухштаб. — М.: Просвещение, 1966 — 392 с.
3. Волошин, О. Ф. Моделі та методи прийняття рішень. / О. Ф. Волошин, С. О. Мащенко — К.: Видавничо-поліграфічний центр «Київський університет», 2010. — 336 с.
4. Дирихле, П. Г. Л. Лекции по теории чисел. В обработке и с добавлениями Р. Дедекинда. / П. Г. Л Дирихле. — М.: Объединённое научно-техническое издательство НКТП СССР, 1936 — 404 с.
5. Кормен, Т. Вступ до алгоритмів. / Томас Г. Кормен, Чарлз Е. Лейзерсон, Роналд Л. Рівест, Кліфорд Стайн. — К., К.І.С., 2019 — 1288 стор.
6. Порубльов, І. М. Деякі алгоритми пошуку чисел з максимальною кількістю дільників / Вісник Черкаського національного університету, серія «Прикладна математика. Інформатика». Випуск № 1.2020, с. 44–60.
7. Debreu, Gerard. Valuation Equilibrium and Pareto Optimum. / Proceedings of the National Academy of Sciences of the United States of America, Vol. 40, No. 7 (Jul. 15, 1954), pp. 588–592.
8. Preparata F. P., An optimal real-time algorithm for planar convex hulls / Communications of the ACM, Volume 22, Issue 7, 1979.

References:

1. The Article "Supercomposite number". Access mode: https://uk.wikipedia.org/wiki/Надскладене_число [in Ukrainian].
2. Bukhshtab, A.A. (1966) Theory of numbers. – M.: Education. – 392 p. [in Russian]
3. Voloshin, O.F. (2010) Models and methods of Decision-Making. – K.: Kyiv university publishing center. – 336 p. [in Ukrainian]
4. Dirichlet, P.G.L. (1936) Lectures on number theory. In processing and with additions by R. Dedekind. – M.: United Scientific and Technical Publishing House of the NKTP of the USSR. – 404 p. [in Russian]
5. Cormen T., Leiserson C., Rivest R., Stein C. (2019) Introduction to Algorithms. – K.: K. I. C. – 1288 p. [in Ukrainian].
6. Porublyov, I. Some algorithms for searching numbers with maximal quantity of divisors // Bulletin of the Cherkasy Bohdan Khmelnytsky National University. Series "Applied Mathematics. Informatics". Issue #1.2020, pp. 44–60.
7. Debreu, Gerard. Valuation Equilibrium and Pareto Optimum. / Proceedings of the National Academy of Sciences of the United States of America, Vol. 40, No. 7 (Jul. 15, 1954), pp. 588–592.
8. Preparata F. P., An optimal real-time algorithm for planar convex hulls / Communications of the ACM, Volume 22, Issue 7, 1979.

PORUBLYOV Ilya,
Lecturer, Bohdan Khmelnytsky National University of Cherkasy, Ukraine

YET ANOTHER ALGORITHM FOR SEARCH NUMBERS WITH MAXIMAL QUANTITY OF DIVISORS (BASED ON IDEA OF NON-DOMINATED PAIRS SET)

Summary. Introduction. *This article is essentially a continuation of the article [6]. It considers almost the same formulation of the problem (search for number with maximal quantity of divisors among all numbers in given interval, but this time only from 1 to given N, and it's necessary to find minimal of the numbers which have the maximal quantity of divisors). Despite similar formulation, a fundamentally different way of solving is proposed. It's based on the introduced concept of non-dominated pairs, namely: a pair (number, quantity of its divisors) is considered non-dominated iff there are no smaller numbers that have greater or equal quantity of divisors. Similarly to [6], this article considers generalization when maximum quantity of divisors is sought not among all numbers in the interval, but only among non-multiples of some K (natural, greater than or equal to 2, not necessarily prime).*

Purpose. *The purpose of the article is to describe the designed algorithm for solving problem of searching numbers with maximal quantity of divisors, based on sets of non-dominated pairs, and its variations. Similarly to [6], several problem formulation versions are considered:*

1. Given a positive integer N, for all integers in range between 1 and N (both inclusive), find number having maximal (among numbers in the range) quantity of divisors.

3. Given positive integers N and K ($K \geq 2$), for integers in range between 1 and N (both inclusive), but excluding multiples of K, find number having maximal (among the said numbers) quantity of divisors.

Results and conclusion. *After introducing definitions "pair (v, τ(v)) is non-dominated under constraints … (where … is replaced with some constraints) iff it satisfies the constraints and there is no satisfying the constraints pair (w, τ(w)) where w < v and τ(w) ≥ τ(v)" and "S(N, j) is set of all pairs (v, τ(v)), non-dominated under constraints v ≤ N and factorization of v contains non-zero degrees for $p_0 = 2$, $p_1 = 3$, …, p_j", an inductive algorithm building S(N, 0), S(N, 1), … is proposed and proved; theoretical and practical estimate for such j* that S(N, j*) = S(N, j*+1) = … = S(N, ∞) are proposed and proved. Algorithm finding high composite numbers using S(N, 0), S(N, 1), … is implemented. Comparing this algorithm with same-goal algorithm from [6], new one found to be much faster to build all highly-composite numbers in range 1..N, than algorithm from [6] to find maximal only.*

Modification for task formulation 3 is based on introducing new series of sets R(N, j, K) which are sets of all pairs (v, τ(v)), non-dominated under constraints v ≤ N and factorization of v contains non-zero degrees for $p_0 = 2$, $p_1 = 3$, …, p_j, and (what differs R(N, j, K) from S(N, j)) v can't become a multiple of K after multiplying with any product of any powers of p_{j+1}, p_{j+2}, …. Inductive algorithm building S(N, 0), R(N, 0, K), S(N, 1), R(N, 0, K), … is proposed and proved; theoretical and practical estimate for finishing the process are proposed and proved. This algorithm solves problem in formulation 3, consuming amounts of time and memory, comparable to algorithm for formulation 1.

Keywords: *quantity of divisors, highly composite numbers, non-dominated pairs, ordered dictionary data structure, map, lower_bound, upper_bound.*

Одержано редакцією 18.10.2021 р.
Прийнято до публікації 08.12.2021 р.

УДК 519.248

DOI 10.31651/2076-5886-2021-1-49-57

PACS 07.05.Tp, 62.20.M-, 85.40.Qx, 61.43.Bn, 64.60.De

ПАСІЧНИЙ Микола Олександрович,
кандидат фізико-математичних наук,
доцент, завідувач кафедри фізики,
Черкаський національний університет
імені Богдана Хмельницького
e-mail: pasichnyy@ukr.net
ORCID: 0000-0002-8434-1544

ТАТАРЧУК Євгеній Вікторович,
кандидат фізико-математичних наук,
доцент, доцент кафедри фізики,
Черкаський національний університет
імені Богдана Хмельницького
e-mail: etatar@ukr.net
ORCID: 0000-0001-5885-2079

МОДЕЛЮВАННЯ РОЗМІРНОГО ЕФЕКТУ У ПРОЦЕСІ ВІДМОВ МІКРОЕЛЕКТРОННИХ ПРИСТРОЇВ НА ОСНОВІ ПІДХОДУ ФОРМУВАННЯ ПЕРКОЛЯЦІЙНОГО КЛАСТЕРА

Створено комп'ютерну модель та досліджено процес відмов на основі підходу перколяційного кластера. Встановлено показникові залежності середнього часу до відмови, стандартного відхилення та коефіцієнта асиметрії розподілів часів до відмов від розміру системи.