ПРИСКОРЕННЯ АЛГОРИТМУ ВИЗНАЧЕННЯ ДОСКОНАЛИХ ЧИСЕЛ ЗА РАХУНОК ЗМЕНШЕННЯ АСИМПТОТИЧНОЇ ОБЧИСЛЮВАЛЬНОЇ СКЛАДНОСТІ

Шпортько Олександр
кандидат технічних наук, доцент,
доцент кафедри інформаційних систем та обчислювальних методів
Приватного вищого навчального закладу
«Міжнародний економіко-гуманітарний університет
імені академіка Степана Дем'янчука»

Коротчук Богдан
здобувач першого (бакалаврського) рівня вищої освіти
другого року навчання Приватного вищого навчального закладу
«Міжнародний економіко-гуманітарний університет
імені академіка Степана Дем'янчука»

Іскра Андрій
здобувач третього (освітньо-наукового) рівня вищої освіти
Приватного вищого навчального закладу
«Міжнародний економіко-гуманітарний університет
імені академіка Степана Дем'янчука»

Панчук Олександр
здобувач третього (освітньо-наукового) рівня вищої освіти
Приватного вищого навчального закладу
«Міжнародний економіко-гуманітарний університет
імені академіка Степана Дем'янчука»
м. Рівне, Україна

На цей час поняття *обчислювальної складності алгоритму* [1] розглядається в освітніх закладах лише на інтуїтивному рівні [2; 3]. І даремно, адже саме показники обчислювальної складності «дають уявлення про час виконання алгоритмів» [4, с. 57], дозволяють зрозуміти, чому один алгоритм буде працювати значно швидше від іншого, привчають майбутніх програмістів звертати увагу не лише на правильність розв'язку поставленої задачі, а й на його ефективність, не лише на час виконання програми, а й на обсяги використовуваної пам'яті. На нашу думку, поняття *асимптотичної складності алгоритму* [5] доцільно вводити не лише для порівняння ефективності різних алгоритмів сортування у ВНЗ чи у 9 класі навчальних закладів з поглибленим вивченням інформатики [6, с. 165-172], а й навіть при вивченні вкладених циклів у 7 класі загальноосвітніх шкіл [2].

Покажемо, наприклад, як можна використати асимптотичну складність для аналізу ефективності алгоритмів визначення досконалих чисел на заданому інтервалі.

Як відомо, натуральне число називається *досконалим*, якщо воно рівне сумі своїх дільників за винятком самого себе. Наприклад, число 6 – досконале, оскільки 6=1+2+3, а число 8 – не досконале, адже 8≠1+2+4 [7, с. 136]. Два перших досконалих числа (6 і 28) були відомі задовго до Евкліда, але саме він не лише знайшов два наступні досконалі числа (496 і 8128), а й довів досконаліcть чисел, які можна подати у вигляді $N = 2^{p-1}(2^p-1)$, де 2^p-1 – просте число [8]. На сьогодні відомо понад 50 досконалих чисел і їх подальші пошуки тривають за допомогою проекту розподілених обчислень GIMPS [9].

Зрозуміло, що для визначення досконалості кожного натурального числа *num* (скорочення від *number*) з введеного інтервалу [*m*; *n*] можна скористатися очевидним алгоритмом [10], який для чергового *num* послідовно перебирає натуральні числа з інтервалу [1; *num*-1], визначає серед них дільники числа *num* та обчислює їх суму. Якщо знайдена сума дорівнює числу *num*, то це число досконале. Обчислювальна складність такого алгоритму – величина порядку $O(N^2)$ і тому програми, які його реалізують, можуть не вкладатися у відведений час виконання.

Для зменшення обчислювальної складності зауважимо, що *частка* від ділення числа N на його дільник не може бути меншим 2 (адже частку 1 забезпечує сам дільник N, який недопустимий за умовою задачі), тому дільники числа, менші за N, доцільно шукати не в інтервалі [1; N-1], а в [1; $N/2$] [11, с. 47]. Кількість операцій визначення дільника становить тепер $N^2/4$, але *порядок* асимптотичної обчислювальної складності складає все ще $O(N^2)$, тому час виконання відповідної програми кардинально не зменшиться.

Для кардинального зменшення обчислювальної складності алгоритму зауважимо, що в суму для визначення досконалості чергового числа $N > 1$ завжди ввійде тривіальний дільник 1. Для всіх інших нетривіальних дільників a числа N ($N \% a = 0$) завжди можна визначити також дільник $b = N / a$ цього ж числа N і тому нетривіальні дільники числа N варто шукати навіть не в інтервалі [2; $N/2$], а в [2; $\sqrt{N}$] [11, с. 47-48], але тоді до суми потрібно додавати не лише кожен знайдений дільник a, а й його частку від ділення N на цей дільник, тобто дільник b. Причому додавання дільника b потрібно виконувати лише при $a \neq b$. Порядок асимптотичної обчислювальної складності з цими модифікаціями знизився до $O(N\sqrt{N})$, що суттєво зменшило час виконання відповідних програм.

За результатами дослідження нами зроблено такі **висновки**:

1. Кардинальне зменшення обчислювальної складності алгоритму можливо саме за рахунок зменшення асимптотичної обчислювальної складності.

2. Поняття асимптотичної обчислювальної складності в контексті часу виконання [12] варто формувати в учнів на уроках інформатики всіх загальноосвітніх шкіл, починаючи з 7 класу, в процесі вивчення вкладених циклів. Це формуватиме навики написання не лише правильних, а й ефективних (в контексті часу виконання та використання обчислювальних ресурсів) програм.

3. Пошук досконалих чисел на вказаному інтервалі доцільно виконувати лише серед парних натуральних чисел, аналізуючи їх нетривіальні дільники до квадратного кореня з цих чисел, зменшених на одиницю. При цьому до суми дільників потрібно додавати не лише знайдені дільники, а й отримані частки. Такі вдосконалення відносно варіанту послідовного перебору натуральних чисел інтервалу та їх дільників дають змогу зменшити асимптотичну обчислювальну складність з $O(N^2)$ до $O(N\sqrt{N})$.

ЛІТЕРАТУРА

1. Обчислювальна складність. URL: https://znaimo.com.ua/Обчислювальна_складність#link0 (дата звернення: 15.09.2024).
2. Програма курсу «Інформатика». 5 – 9 класи загальноосвітніх навчальних закладів. URL: https://mon.gov.ua/storage/app/media/zagalna%20serednya/programy-5-9-klas/onovlennya-12-2017/programa-informatika-5-9-traven-2015.pdf (дата звернення: 15.09.2024).
3. Програма курсу «Інформатика». 8 – 9 класи загальноосвітніх навчальних закладів з поглибленим вивченням інформатики. URL: https://mon.gov.ua/storage/app/media/zagalna%20serednya/programy-5-9-klas/informatika.pdf (дата звернення: 15.09.2024).
4. Шинкаренко В. І. Особливості практичного застосування показників обчислювальної складності алгоритмів. *Проблеми програмування*. 2008. № 2-3. С. 57-63.
5. Оцінка складності алгоритмів, або Що таке O(log n). URL: https://echo.lviv.ua/dev/53 (дата звернення: 15.09.2024).
6. Руденко В. Д., Речич Н. В., Потієнко В. О. Інформатика для загальноосвітніх навчальних закладів з поглибленим вивченням інформатики : підруч. для 9 кл. загальноосвіт. навч. закл. Харків : Вид-во «Ранок». 2017. 240 с.
7. Вивчаємо математику: Досконалі числа. URL: http://matematikav6.blogspot.com/2014/09/blog-post_66.html (дата звернення: 15.09.2024).
8. Perfect number. URL: https://en.wikipedia.org/wiki/Perfect_number (дата звернення: 15.09.2024).
9. Great Internet Mersenne Prime Search. URL: https://www.mersenne.org/ (дата звернення: 15.09.2024).

10. Теорія чисел – математичні основи розв'язування олімпіадних задач. URL: https://www.e-olymp.com/uk/blogs/posts/53 (дата звернення: 15.09.2024).
11. Креневич А. П., Обвінцев О. В. С у задачах і прикладах : навчальний посібник із дисципліни "Інформатика та програмування". Київ: Видавничо-поліграфічний центр "Київський університет", 2011. 208 с.
12. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein Introduction to Algorithms, Third Edition. Cambridge: MIT Press, 2009. 1320 p.