ПРО НЕОБХІДНІСТЬ ВИВЧЕННЯ ПОНЯТТЯ «ОБЧИСЛЮВАЛЬНОЇ СКЛАДНОСТІ АЛГОРИТМУ» В СЕРЕДНІХ ШКОЛАХ ТА ФАХОВИХ КОЛЕДЖАХ УКРАЇНИ

Шпортько О. В.
кандидат технічних наук, доцент,
доцент кафедри інформаційних систем та обчислювальних методів
Міжнародного економіко-гуманітарного університету
імені академіка Степана Дем'янчука
м. Рівне, Україна

Шпортько Л. В.
викладач циклової комісії інформатики і комп'ютерної техніки
Рівненського фахового коледжу економіки та бізнесу
м. Рівне, Україна

На сьогодні поняття *обчислювальної складності алгоритму* [1] розглядається в курсі інформатики середньої школи і у фахових коледжах 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 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].

Розглянемо, наприклад, варіанти алгоритмізації розв'язування класичної задачі пошуку досконалих чисел на проміжку від *m* до *n* [7]. 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.

Результати тестування програми мовою C#, розробленої за цим алгоритмом, на сайті https://www.e-olymp.com (задача № 848) свідчать, що незважаючи на правильність реалізованого алгоритму, ця програма не змогла пройти 25% тестів, оскільки вичерпала ліміт часу. Виконуючи 25% інших тестів програма вичерпала майже весь відведений час (понад 4 с). При цьому обсяг використання оперативної пам'яті практично не змінюється і коливається в межах 25% від максимально допустимого.

Після отримання таких результатів варто наголосити учням чи студентам фахових коледжів, що дієздатний алгоритм і відповідна програма – це ще не все. Потрібно, щоб програма виконувалася за відведений час і економно використовувала оперативну пам'ять. Скільки операцій визначення дільника виконує дана програма, якщо $m=1$, а n – як завгодно велике (позначимо його через N)? Використовуючи формулу суми n перших членів арифметичної прогресії, ця кількість рівна $N\times(N\text{-}1)/2$. І ось тут варто ввести поняття асимптотичної обчислювальної складності і наголосити, що це величина порядку $O(N^2)$.

Як зменшити обчислювальну складність цього алгоритму? Потрібно зауважити, 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 $O(N^2)$. Для програми з таким вдосконаленням час виконання тестів на згаданому сайті для невеликих N майже не змінився, для середніх – зменшився приблизно на 45%, а для великих – все ще не вкладається у 5 с. Тобто зменшення вдвічі кількості операцій визначення дільника пропорційно вплинуло на час виконання програми.

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. Покладемо $2 \le a \le b$. Оскільки $a \times b = N$, то $a^2 \le N$, тобто $2 \le a \le N$. Отже, нетривіальні дільники числа N варто шукати навіть не в інтервалі [2; N/2], а в [2; 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 при $a \neq b$, оскільки, за умовою задачі, у сумі дільники мають бути унікальні (це некоректно зроблено в [11, с. 48]). Цим самим вдасться уникнути подвійного додавання гранично можливого дільника.

Порядок асимптотичної обчислювальної складності з цими модифікаціями знизився до $O(N^{N})$ і тому середній час виконання тестів на сайті https://www.e-olymp.com для задачі № 848 зменшився в десятки разів, що дало змогу розв'язати поставлену задачу.

Зі сказаного вище приходимо до таких **висновків**:

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 вводити для студентів фахових коледжів та 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 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 $O(N^n)$.

Література:

1. Обчислювальна складність. URL: https://znaimo.com.ua/Обчислювальна_складність#link0 (дата звернення: 21.12.2020).
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: 21.12.2020).
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: 21.12.2020).
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: 21.12.2020).
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. Абрамов С. А., Гнездилова Г. Г., Капустина Е. Н., Селюн М. И. Задачи по программированию. Вологда, 2000. 596 с.
8. Perfect number. URL: https://en.wikipedia.org/wiki/Perfect_number (дата звернення: 21.12.2020).
9. Great Internet Mersenne Prime Search. URL: https://www.mersenne.org/ (дата звернення: 21.12.2020).
10. Теорія чисел – математичні основи розв'язування олімпіадних задач. URL: https://www.e-olymp.com/uk/blogs/posts/53 (дата звернення: 21.12.2020).
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.

ІНФОРМАЦІЙНА БЕЗПЕКА ІНТЕРНЕТУ РЕЧЕЙ

Юскович-Жуковська В. І.
кандидат технічних наук, доцент,
доцент кафедри інформаційних систем та обчислювальних методів
Міжнародного економіко-гуманітарного університету
імені академіка Степана Дем'янчука
м. Рівне, Україна

В сучасному світі пристрої Інтернету речей (IoT) стають звичними інструментами буденних справ, а безпровідне, online та голосове керування ними за допомогою девайсів визначає новітній розвиток цифровізації.

Згідно досліджень компанії Juniper Research у 2022 році очікується активних IoT пристроїв у світі понад 50 мільярдів [1]. У доповіді Fortune Business зазначається, що світовий ринок Інтернету речей до 2026 року