Złożoność obliczeniowa algorytmów (ang. *algorithmic complexity*)

Jest wiele problemów algorytmicznych, dla których nie istnieją dokładne matematyczne odpowiedzi. Nie zawsze jest to jednak konieczne, ponieważ ważniejsze może być przybliżone oszacowanie. Takie uproszczenie jest dopuszczalne z kilku względów, gdyż czas wykonania algorytmu i tak jest różny dla różnych komputerów, zależy od języka programowania, zastosowanego kompilatora, używania pamięci zewnętrznej, a nawet od umiejętności programisty.

Dla porównania dwóch różnych algorytmów, rozwiązujących ten sam problem, wystarczy porównanie tempa ich wzrostu w miarę wzrostu ilości danych do przetworzenia. Jeśli liczba danych przybiera duże rozmiary (zmierza do nieskończoności), to analiza algorytmów opiera się na asymptotyce (ang. *asymptotic*), która zajmuje się metodami przybliżonymi.

Różne funkcje liczbowe w różnym tempie zmierzają do nieskończoności. Porównując ze sobą dwie funkcje można ogólnie stwierdzić, że

$$f(n) \prec g(n) \Leftrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

gdzie:
symbol „$\prec$” oznacza „rośnie wolniej”
symbol *lim* (łac. *limes* - granica) oznacza granicę wyrażenia

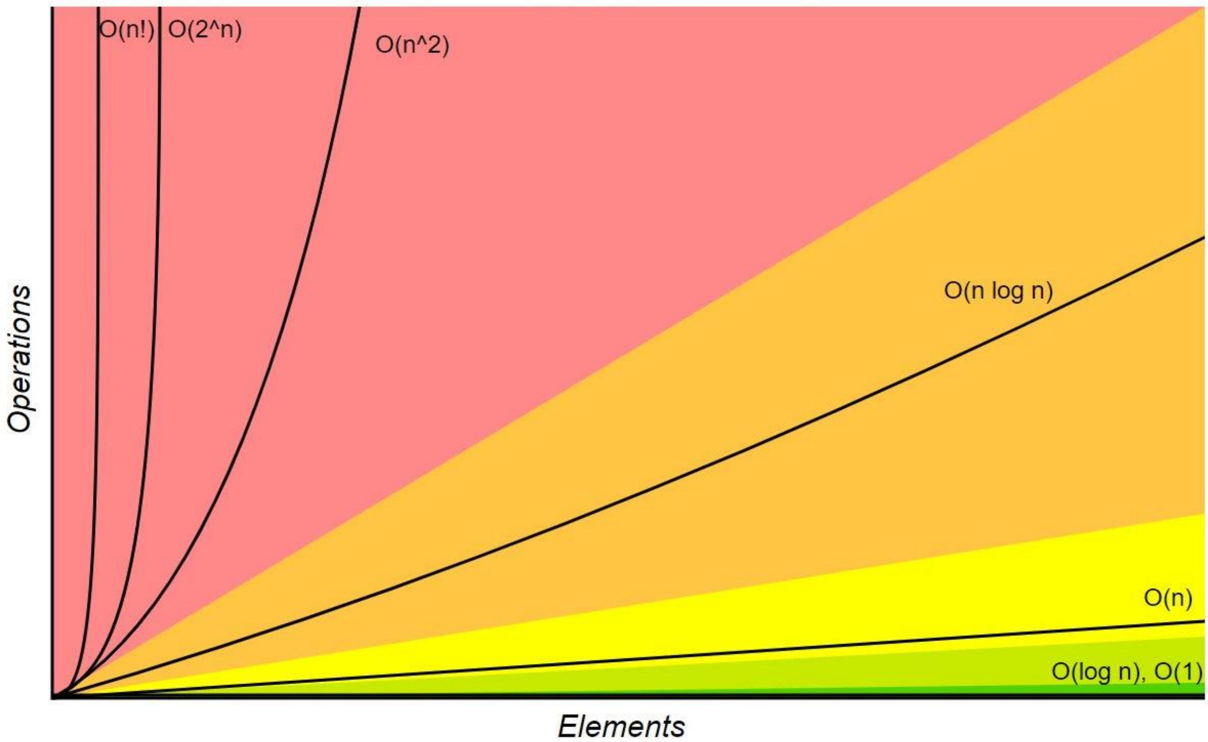
Zatem funkcja *f(n)* rośnie wolniej niż *g(n)* wtedy i tylko wtedy, gdy granica ilorazu tych funkcji przy *n* dążącym do nieskończoności zmierza do zera.

Relacji „$\prec$” można użyć do określenia rzędu różnych funkcji i ich uszeregowania w porządku asymptotycznym, zawierającym np. pozycje

$$1 \prec loglogn \prec logn \prec n^{\varepsilon} \prec n^{c} \prec n^{logn} \prec c^{n} \prec n! \prec n^{n}$$

gdzie: $0 < \varepsilon < 1 < c$

Za wyjątkiem liczby 1 wszystkie funkcje zmierzają do nieskończoności dla $\mathrm{n} \to \infty$.



Notacja $O(\cdot)$

Dla celów szacowania rzędu wielkości metodami asymptotycznymi wprowadzono tzw. notację „duże O z kropką” (ang. *Big-O notation*) – w skrócie $O(\cdot)$, gdzie w miejsce kropki wstawia się właściwą wielkość. Korzystając z tej notacji można zapisać

$$\frac{2}{3}n^3 + \frac{1}{2}n^2 + \frac{1}{4}n = O(n^3) \ \ (\text{czyt.: "duże O od n do trzeciej})$$

Notacja $O(\cdot)$ narzuca równości jednostronne. Oznacza to, że przytoczonego równania nigdy nie wolno zapisać z zamienionymi stronami. Jego prawa strona dostarcza bowiem mniej informacji niż lewa i jest niejako jej „rozmyciem” . Znak równości między lewą i prawą stroną równania jest więc pewnym nadużyciem. Precyzyjniejszy byłby symbol zawierania zbiorowego ($L \subset P$), jednakże ze względów historycznych znak równości jest powszechnie stosowany.

Zapis np. *O(n)* stanowi, że pewna funkcja *f(n)* rośnie liniowo wraz ze zmienną *n*. Nie jest przy tym istotne, czy w rzeczywistości *f(n) = 100n* lub *f(n) = 0,01n*. Z definicji notacji *O(·)* wynikają następujące własności:

$$f(n) = O(f(n))$$

$$c \cdot f(n) = O\big(f(n)\big) \qquad\qquad \text{c} - \text{pewna stała}$$

$$O(O\big(f(n)\big)) = O(f(n))$$

$$O(f(n)) \cdot O(g(n)) = O(f(n) \cdot g(n))$$

Korzystając z notacji $O(\cdot)$ można zapisać np.

$$\frac{2}{3}n^3 + \frac{1}{2}n^2 + \frac{1}{4}n = \frac{2}{3}n^3 + \frac{1}{2}n^2 + O(n)$$

$$\frac{2}{3}n^3 + \frac{1}{2}n^2 + \frac{1}{4}n = \frac{2}{3}n^3 + O(n^2)$$

$$\frac{2}{3}n^3 + \frac{1}{2}n^2 + \frac{1}{4}n = O(n^3)$$

Teoretycznie dopuszczalny byłby również np. zapis

$$0 \cdot n^5 + 0 \cdot n^4 + \frac{2}{3}n^3 + \frac{1}{2}n^2 + \frac{1}{4}n = O(n^5)$$

Jednakże jest on nieprawidłowy, ponieważ powoduje on niepotrzebną utratę części informacji – w równaniu faktycznie nie występuje zmienna n w czwartej i piątej potędze.

Uwaga:
Oprócz notacji $O\big(f(n)\big)$ istnieją jeszcze dwie inne notacje, tj. $\Omega\big(f(n)\big)$ oraz $\Theta\big(f(n)\big)$, które poznamy w przyszłości.

Złożoność obliczeniowa algorytmu określa ilość czasu oraz zużycie zasobów komputera, potrzebnych do rozwiązania danego problemu.

Wyróżniamy następujące typy złożoności obliczeniowej:

- **złożoność pesymistyczna –** ilość zasobów komputerowych potrzebnych przy wprowadzeniu „najgorszych” danych wejściowych;

- **złożoność optymistyczna –** ilość zasobów komputerowych potrzebnych przy wprowadzeniu „najlepszych" danych wejściowych;
- **złożoność oczekiwana** – ilość zasobów komputerowych potrzebnych przy wprowadzeniu „typowych" danych wejściowych.

Ze względu na rodzaje zasobów komputerowych złożoność obliczeniową dzielimy na:

1. Złożoność czasową,
2. Złożoność pamięciową.

Złożoność czasowa obejmuje czas działania algorytmu, ponadto jest własnością algorytmu niezależną od komputera, języka programowania itp. Jednostką złożoności czasowej jest wykonanie jednej operacji dominującej. Mierzona jest więc liczbą operacji dominujących (zależną od danych wejściowych), które wykonywane są podczas realizacji algorytmu. Operacjami dominującymi w danej metodzie nazywamy podstawowe działania, na przykład porównanie dwóch elementów, zamianę wyrazów ciągu, operację przypisania, działania arytmetyczne: dodawanie, odejmowanie, mnożenie i dzielenie, które dla danego algorytmu są charakterystyczne i wykonywane najczęściej. Łączna liczba tych działań powinna być proporcjonalna do liczby wystąpień pozostałych operacji podstawowych wykonywanych w tej metodzie.

Złożoność pamięciowa określa, ile pamięci potrzeba do realizacji danej metody. Również w tym przypadku złożoność jest zależna od liczby wprowadzanych danych. Mierzona jest więc liczbą zmiennych wykorzystanych w algorytmie oraz zajmowaną przez nie pamięcią.