Analiza algorytmów. Szacowanie złożoności obliczeniowej.

Złożoność obliczeniowa (czasowa) algorytmu - zależność pomiędzy rozmiarem danych wejściowych a liczbą operacji elementarnych wykonywanych w trakcie przebiegu algorytmu (podawana jako funkcja rozmiaru danych).

Operacje elementarne to np. liczba porównań dwóch liczb w algorytmie sortowania, liczba mnożeń w algorytmie obliczania silni, liczba dzieleń w algorytmie sprawdzania czy dana liczba jest pierwsza, ogólna liczba operacji arytmetycznych wykonywanych przez dany algorytm itp..

Złożoność obliczeniowa jest jednym z najważniejszych parametrów charakteryzujących algorytm. Decyduje ona o efektywności całego programu. Podstawowymi zasobami systemowymi uwzględnianymi w analizie algorytmów są czas działania (złożoność obliczeniowa algorytmu) oraz obszar zajmowanej pamięci (złożoność pamięciowa algorytmu). Złożoność pamięciowa wynika z liczby i rozmiaru struktur danych wykorzystywanych w algorytmie. Złożoność algorytmu może być rozumiana w sensie złożoności najgorszego przypadku lub złożoności średniej. Złożoność najgorszego przypadku nazywamy złożonością pesymistyczną - jest to maksymalna złożoność dla danych o zadanym rozmiarze n. Złożoność średnia lub oczekiwana to średnia wartość złożoności dla wszystkich problemów rozmiaru n. Najczęściej algorytmy mają złożoność czasową proporcjonalną do funkcji:

— $\log(n)$- złożoność logarytmiczna,
— n - złożoność liniowa,
— $n\log(n)$ - złożoność liniowo-logarytmiczna,
— n^2 - złożoność kwadratowa,
— n^k - złożoność wielomianowa, $(k > 2)$,
— $2^{\log n}$ - złożoność podwykładnicza,
— 2^n - złożoność wykładnicza,
— $n!$ - złożoność wykładnicza

Notacja asymptotyczna

Rząd wielkości służy do opisu czasu działania algorytmu. Istnieją 3 notacje służące do tego celu:

Funkcja asymptotycznie niewiększa od funkcji $g(n)$ to taka funkcja $f : \mathbb{N} \longrightarrow \mathbb{R}$, dla której istnieją $c > 0$ i $n_0 \in \mathbb{N}$, że $|f(n)| \leqslant c \cdot |g(n)|$ dla wszystkich $n \geqslant n_0$.

Będziemy też często mówić, że $|f(n)| \leqslant c \cdot |g(n)|$ zachodzi dla prawie wszystkich liczb naturalnych n. Zbiór funkcji asymptotycznie niewiększych niż $g(n)$ oznaczamy przez $O(g(n))$.

Funkcja asymptotycznie niemniejsza od funkcji $g(n)$ to taka funkcja $f : \mathbb{N} \longrightarrow \mathbb{R}$, dla której istnieją $c > 0$ i $n_0 \in \mathbb{N}$, że $c \cdot |g(n)| \leqslant |f(n)|$ dla wszystkich $n \geqslant n_0$.

Zbiór funkcji asymptotycznie niemniejszych niż $g(n)$ oznaczamy przez $\Omega(g(n))$.

Funkcja asymptotycznie podobna do funkcji $g(n)$ to taka funkcja $f : \mathbb{N} \longrightarrow \mathbb{R}$, dla której istnieją $c_0, c_1 > 0$ i $n_0 \in \mathbb{N}$, że $c_0 \cdot |g(n)| \leqslant |f(n)| \leqslant c_1 \cdot |g(n)|$ dla wszystkich $n \geqslant n_0$.

Zbiór funkcji asymptotycznie podobnych do $g(n)$ oznaczamy przez $\Theta(g(n))$. A zatem $\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$.

Odpowiednio w stosunku do wcześniej wypisanych funkcji powiemy, że złożoność obliczeniowa algorytmu wynosi (lub algorytm „działa w czasie") $O(log(n))$, $O(n)$, $O(n \log(n))$, itd.

Gdzie np. $\log n = O(n)$, $n^2 = \Omega(n)$, $n = O(n)$, $n = \Omega(n)$, $n = \Theta(n)$, $20n = \Theta(n)$.

Zadania na ćwiczenia

Zadanie 0

Jaka jest złożoność obliczeniowa algorytmu wyszukiwania maksimum ? Sumowania elementów macierzy?

Zadanie 1

Uporządkować rosnąco wg rzędu następujące funkcje:
$2^{n!}$, n^2 , $(2009!)n^2$, $n^3 - n^2$, n^3 , $n^3 + n^2$, 2^n , $n^{2009!}$, $n!$, $n \log n$, $\log n$.

Zadanie 2

Udowodnij przechodność:

$$f(n) = \Theta(g(n)) \text{ i } g(n) = \Theta(h(n)) \Rightarrow f(n) = \Theta(h(n))$$

Zadanie 3

Znajdź i wykaż (udowodnij) zależności pomiędzy funkcjami w notacji asymptotycznej:
— $2n$ w odniesieniu do $(20!)n$,
— $n^2 - n$ w odniesieniu do $n^2 + n$,
— $\log_2 n$ w odniesieniu do n,
— 3^n w odniesieniu do 2^n
— 2^{n+1} w odniesieniu do 2^n
— 2^{2n} w odniesieniu do 2^n
— $n!$ w odniesieniu do 2^n

Zadanie 4

Podaj złożoność funkcji rekurencyjnych:

$$T(n) = 2T(\frac{n}{2}) + 5n - 15$$

$$T(n) = 16T(\frac{n}{4}) + n \lg n$$

$$T(n) = 2T(\frac{n}{2}) + n \lg n$$

Podaj dokładnie asymptotyczne oszacowanie dla rekurencji

$$T(n) = T(n-1) + n + 1$$

Zadanie 5

Podaj złożoność algorytmów rekurencyjnych:
Wyliczającego silnię, fibonacciego? Wyszukiwania binarnego, sortowania przez scalanie?

Literatura

Wykład Asymptotyka