Dziel i zwyciężaj
05-03-2020

Dane: X

1. Transformacja $X \to X_1, X_2, \ldots, X_k$ gdzie rozmiar$(X_i) <$ rozmiar(X)
2. Dla każdego i $W_i \leftarrow$ wynik X_i
3. Skonstruuj wynik dla X z $W_1, W_2, \ldots, W_k$

1 Sortowanie przez scalanie

$X_1 = x_1, x_2, x_3, \ldots, x_{\frac{n}{2}}$
$X_2 = x_{\frac{n}{2}+1}, \ldots, x_n$

1. Posortuj X_1 i $X_2 \to W_1, W_2$

2. Scal W_1, W_2

Algorithm 1 mergesort

mergesort$(T[1 \ldots n])$
if małe(n) **then**
    insertsort$(T[1 \ldots n])$
else
    $X[1 \ldots \lceil \frac{n}{2} \rceil] \leftarrow T[1 \ldots \lceil \frac{n}{2} \rceil]$
    $Y[1 \ldots \lfloor \frac{n}{2} \rfloor] \leftarrow T[1 + \lceil \frac{n}{2} \rceil \ldots n]$
    mergesort(X)
    mergesort(Y)
    $T \leftarrow$ scal(X, Y)
end if

2 Quicksort

Algorithm 2 quicksort

QS$(T[1 \ldots n])$
if małe(n) **then**
    insertsort$(T[1 \ldots n])$
else
    wybierz x – element dzielący
    przestaw elementy T tak aby $\forall_{i \leq k} T[i] \leq x$ (k – liczba elementów $\leq x$)
    QS$(T[1 \ldots k])$
    QS$(T[k+1 \ldots n])$
end if

3 Twierdzenie

$$T(n) = \begin{cases} b & \text{, gdy } n = 1 \\ aT(\frac{n}{c}) + bn & \text{, gdy } n > 1 \end{cases}$$

$a, b, c \in \mathbb{N}$

$$T(n) = \begin{cases} O(n) & \text{, gdy } a < c \\ O(n \log n) & \text{, gdy } a = c \\ O(n^{\log_c a}) & \text{, gdy } a > c \end{cases}$$

4 Mnożenie długich liczb

Dane: $A, B \in \mathbb{N}$ n – długość a (i b)
Wynik: $C = A \cdot B$

Mnożenie Karatsuby
$A = a_{n-1}a_{n-2}\ldots a_0$
$B = b_{n-1}b_{n-2}\ldots b_0$
$A = A_1 \cdot 2^{\frac{n}{2}} + A_0$
$B = B_1 \cdot 2^{\frac{n}{2}} + B_0$
$A \cdot B = (A_1 \cdot 2^{\frac{n}{2}} + A_0) \cdot (B_1 \cdot 2^{\frac{n}{2}} + B_0) = A_1 \cdot B_1 \cdot 2^n + (A_0 \cdot B_1 + A_1 \cdot B_0) \cdot 2^{\frac{n}{2}} + A_0 \cdot B_0$

Prosty algorytm:

Algorithm 3 mult

```
mult(A, B) :
W_1 = A_1 B_1
W_2 = A_1 B_0
W_3 = A_0 B_1
W_4 = A_0 B_0
zwróć W_1 2^N + ... + W_4
```

Korzystając z **Twierdzenie** $T(n) = 4 \cdot T(\frac{n}{2}) + O(n) = O(n^2)$.
Lepszy algorytm:

Algorithm 4 mult

```
mult(A, B) :
W_1 = (A_0 + A_1)(B_0 + B_1) = A_0 B_) + A_1 B_0 + A_0 B_1 + A_1 B_1
W_2 = A_0 B_0
W_3 = A_1 B_1
zwróć W_2 2^N + (W_0 − W_1 − W_2) 2^(n/2) + W_0
```

Korzystając z **Twierdzenie** $T(n) = 3 \cdot T(\frac{n}{2}) + O(n) = O(n^{1.58})$.
Oczywiście możemy podzielić całość na więcej kawałków. Wtedy dla ogólnego przypadku otrzymamy:
$T(n) = (2k-1)T(\frac{n}{k}) + O(n) = O(n^{\log \frac{2k-1}{k}})$