Selekcja

Problem Selekcji

Dla elementów tablicy T znajdź k-ty największy element.

Proste przypadki

1. $k=1 \Rightarrow \Theta(n)$

   Wykonujemy $n-1$ porównań porównując z aktualną wartością maksymalną.

2. $k=2 \Rightarrow \Theta(n)$

   Można wykonać nawet $(n-1)+\lceil \log n \rceil - 1$ porównań

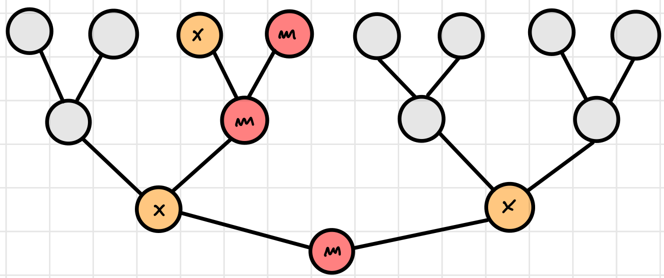
3. Przypadek ogólny $\Rightarrow \Theta(n)$

   Możemy wykonać algorytm selection.

Drugi największy element 🥈

Możemy oczywiście znaleźć największy ($n-1$ porównań) a następnie usunąć ten element i znowu znaleźć największy ($n-2$ porównań) razem wykonując $2n-3$ porównań.

Pytanie czy da się wykonać mniej porównań by znaleźć drugi największy element. Odpowiedź da się. Najpierw znajdujemy największy element za pomocą algorytmu dziel i zwyciężaj w którym zwracamy max(a, b). Wygrał element m. Następnie spośród elementów, które przegrały z m znowu wyznaczamy największy element - jest ich $\log n$.



m - zwycięzca ($n-1$ porównań)

x - przegrani z m ($\lceil \log n \rceil - 1$ porównań)

Razem $n-1+\lceil \log n \rceil - 1 = n + \lceil \log n \rceil - 2$ porównań

Selection

selection(T, k)

1. jeśli T jest małe, to posortuj i zwróć medianę
2. Wybierz jakiś piwot **P** z tablicy T.
3. Wrzuć do tablicy U elementy większe od **P**
4. Jeśli tych elementów jest $\geq k$

   then) return selection(U, k) ← Podobnie jak w Quick Sort

   else) return selection$(T \setminus U, k - |U|)$

Przypadek ogólny możemy zastosować do znajdywania mediany.

Wtedy $k = \lceil \frac{n}{2} \rceil$

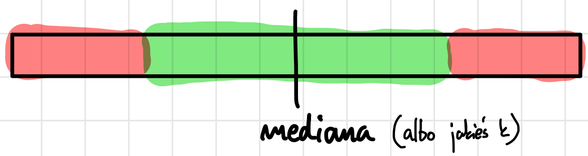
Selekcja z randomizacją 

Przeanalizujmy wariant algorytmu selection w którym piwot P znajdujemy losowo wybierając element z tablicy.

Dowód poprawności

Założmy, że randomizujemy wybór pivota.

Dla posortowanego ciągu 50% piwotów jest dobrych, gdzie dobry piwot definiujemy tak, że odrzucimy ze zbioru conajmniej $\frac{1}{4}$ danych.



Wtedy z prawdopodobieństwa mamy 50% szans na iteracje po $\frac{3}{4}$ danych lub po n danych.

$$T(n) = \frac{1}{2}T\left(\frac{3}{4}n\right) + \frac{1}{2}T(n) + O(n) \quad / - \frac{1}{2}T(n)$$

$$\frac{1}{2}T(n) = \frac{1}{2}T\left(\frac{3}{4}n\right) + O(n) \quad / \cdot 2$$

$$T(n) = T\left(\frac{3}{4}n\right) + O(n)$$

Z master Theorem $\rightarrow O(n)$

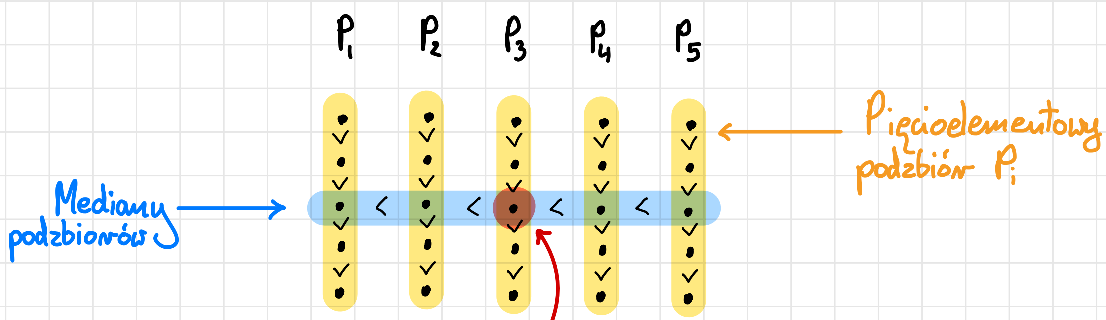
Magiczne Piątki 🙌

Algorytm magicznych piątek jest wykorzystywany w algorytmie selection (krok 2) do znajdowania sensownego pivotu **P**.

Uwaga! Algorytm magicznych piątek nie znajduje mediany. Zamiast tego znajduje tzw. medianę median, która jest po prostu sensownym elementem do podziału tablicy.

Algorytm:

1. Jeśli tablica jest mała, to posortuj i zwróć medianę
2. Dzielimy zbiór na podzbiory P_i o długości 5
3. Sortujemy zbiory P_i. Sortowanie pojedynczego takiego zbioru zajmie nam $O(1)$ ponieważ ma rozmiar 5 (stały)
4. Następnie tworzymy zbiór median podzbiorów.



5. Znajdujemy medianę median rekurencyjnie wywołując Selection na zbiorze median podzbiorów, by znaleźć ich medianę.

Uwaga! Użytkownik nie wywołuje raczej tego algorytmu bezpośrednio. W rzeczywistości wywołuje funkcję selection, która w kroku drugim wywołuje funkcję magic-five.

Złożoność Obliczeniowa

1–3. $\Theta(n)$

4. $T\left(\frac{n}{5}\right)$

5. $T(\max(U, T \setminus U))$

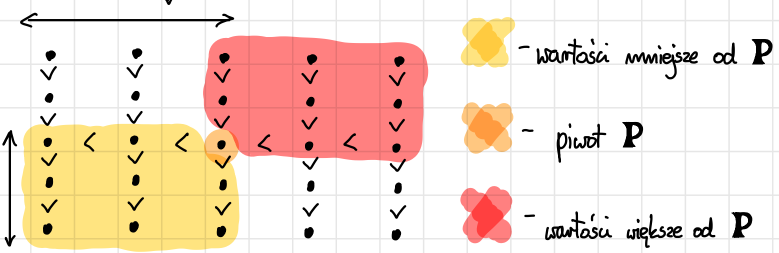
Tutaj zauważmy, że zbiór mniejszych elementów od mediany zawiera conajmniej mniejsze mediany oraz elementy od nich mniejsze. Takich elementów jest

$$3\left(\frac{1}{2}\left\lceil\frac{n}{5}\right\rceil\right)-1 \leftarrow \text{pivot } \mathbf{P}$$

po 3 elementy w każdym podzbiorze

połowa liczby podzbiorów

liczba podzbiorów



Analogicznie mamy dla wartości większych.

Zatem odrzucamy zawsze tyle elementów

$$3\left(\frac{1}{2}\left\lceil\frac{n}{5}\right\rceil - 1\right) = \frac{3n}{10} - 1 \geq \frac{3n}{10}$$

A wykonujemy operacje na $\frac{7n}{10}$ elementach

Podsumowując:

$$T(n) = T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) + \Theta(n)$$

$$T(n) = T\left(\frac{9n}{10}\right) + \Theta(n)$$

$$a = 1 \quad b = \frac{9}{10}$$

Zatem $a < b$:

$$T(n) = O(n)$$

<u>Kontynując</u>

Tak jak algorytm magicznych piątek poniższe algorytmy również znajdują tylko optymalny piwot P.

Mediana trzech elementów 🤟

Niedeterministyczna wersja

Idea

1. Wybieramy 3 liczby losowe
2. Wybieramy medianę z tych trzech liczb
3. Nowym pivotem **P** jest teraz ta mediana.

Ta optymalizacja istotnie zmniejsza prawdopodobieństwo wybrania złego pivota.

Deterministyczna wersja

Zamiast wybierania trzech losowych elementów - wybieramy trzy pierwsze z brzegu.

Laaaazy Select 🥱

1. Wybieramy losowo $n^{\frac{3}{4}}$ elementów z tablicy i umieszczamy w tablicy R.

$\frac{3}{4} n^{\frac{3}{4}} \cdot \underbrace{\log n}_{< n^{\frac{1}{4}}} < \frac{3}{4} n$

2. Sortujemy te elementy $O(n^{\frac{3}{4}} \log n^{\frac{3}{4}}) = O(n)$

3. $x = k \cdot n^{-\frac{1}{4}}$ gdzie $k \to k$-ty element
   $l = x - \sqrt{n}$
   $h = x + \sqrt{n}$

4. Porównując l i h ze wszystkimi elementami oblicz

   $P = \{ y \in S : l \leqslant p \leqslant h \}$ oraz $r(l) = |\{ y \in S : y < l \}|$

   zbiór w którym z dużym prawdopodobieństwem znajdziemy medianę

   Liczba elementów mniejszych od tych w P

5. Sprawdź czy k-ty element zbioru S znajduje się w P oraz czy $|P| \leqslant 4n^{\frac{3}{4}} + 2$

   then) zwróć $P[k - r(l) + 1]$

   else) wróć do punktu 1.

Dowód

Dowód wykracza poza materiał.