Wojciech "Sushi" Jasiński   Anna Karaś   Konrad Kleczkowski   Krzysztof Łyskawa
Mikołaj Motak   Grzegorz Winiarski

Algorytmy i Struktury Danych

egzamin poprawkowy, cz. I

wrzesień 2018

Zadanie 1. W algorytmie znajdowania mediany w zbiorze S poprzez próbkowanie w jednym z pierwszych kroków wybierany jest losowy podzbiór $R \subseteq S$. Ile maksymalnie procent elementów zbioru S może znaleźć się w zbiorze R?

Zadanie 2. Jaką złożoność ma uogólnienie algorytmu Karatsuby, w którym mnożone liczby dzielone są na trzy części? Odpowiedź uzasadnij.

Rozwiązanie (Sushi). Szkic dowodu.
Niech
$x = x1 + +x2 + +x3$
$y = y1 + +y2 + +y3$ (równej długości)
$x * y = x3 * y3 + ...$ (dużo rzeczy).
W takiej postaci mamy 9 mnożeń i to za dużo. W Karatsubie zmniejszamy liczbę mnożeń przedstawiając sumy $E(x_i * y_i)$ jakoś sprytnie (zamiast tego mamy więcej dodawań/odejmowań), ale nie pamiętam jak i ile wtedy zostaje mnożeń. Natomiast, zakładając, że 7:
$T(n) = 7 * T(n/3) + O(n)$
I wtedy
$T(n) = n^(log_3 7)$.

Zadanie 3. Sformułuj zasadę zero-jedynkową (albo inaczej lemat zero-jedynkowy) dla sieci sortujących.

Rozwiązanie. Sieć sortująca sortuje poprawnie dowolne ciągi, jeśli sortuje poprawnie wszystkie ciągi zer i jedynek.

Zadanie 4. Podaj jaki jest pesymistyczny czas wykonywania operacji `find`, gdy operacja `union` wykonywana jest w sposób:

(a) zbalansowany,
(b) niezbalansowany.

Rozwiązanie (Sushi).

(a) $O(n)$
(b) $O(logn)$

Zadanie 5. Opisz (albo zapisz w pseudokodzie), w jaki sposób wykonywana jest operacja znajdowania minimum w drzewie van Emde Boasa. Jak kosztowna jest ta operacja?

Rozwiązanie (Sushi). W strukturze drzewa jest pole, w którym trzymamy minimum, więc możemy je od razu zwrócić. $O(1)$.

Zadanie 6. Czy usunięcie wierzchołka z drzewca o n wierzchołkach może wymagać $\Omega(n)$ operacji rotacji? Jeśli tak, narysuj taki drzewiec i wskaż w nim ten usuwany wierzchołek. Jeśli nie, przedstaw uzasadnienie.

Zadanie 7. Opisz, w jaki sposób sieci sortujące ciągi bitoniczne są użyte w konstrukcji sieci scalającej dwa uporządkowane ciągi.

Zadanie 8. Podaj optymalny (pod względem liczby wykonywanych porównań) algorytm jednoczesnego znajdowania minimum i maksimum w ciągu. Ile porównań wykonuje ten algorytm?

Rozwiązanie (Łyskawa). Porównujemy ze sobą parami kolejne elementy zbioru. Większy z nich przenosimy do zbioru $S_{\max}$, większy do zbioru $S_{\min}$. $n/2$ porównań.

Wyszukujemy element minimalny zbioru $S_{\min}$ i element maksymalny zbioru $S_{\max}$, każde wyszukiwanie to kolejnych $n/2 - 1$ porównań. Razem $3n/2 - 2$ porównań.

Zadanie 9. Opisz, w jaki sposób wykonywana jest operacja usuwania klucza ze słownika zapamiętywanego w tablicy haszującej z adresowaniem otwartym, w której kolizje rozwiązywane są za pomocą metody podwójnego haszowania.

Zadanie 10. Podaj definicję problemu plecakowego bez powtórzeń i przedstaw pseudowielomianowy algorytm rozwiązujący ten problem. Uzasadnij stwierdzenie, że jest on pseudowielomianowy.

Zadanie 11. Rozwiąż poniższe równanie rekurencyjne:

$$T(n) = \begin{cases} 1 & \text{dla } n = 2 \\ 4T(\sqrt{n}) + \mathcal{O}(1) & \text{dla } n > 1 \end{cases}$$

Rozwiązanie (K. Kleczkowski). Niech $n > 1$.

Należy podstawić $n = 2^k$, by pozbyć się pierwiastka, przy czym $\mathbb{N} \ni k \geqslant 0$. Stąd otrzymujemy, że $T(2^k) = 4T(2^{k/2}) + \mathcal{O}(1)$. Podstawmy $U(k) = T(2^k)$. Mamy więc $U(k) = 4U(k/2) + \mathcal{O}(1) = 4U(k/2) + \mathcal{O}(k^0)$.

Ponieważ $\log_2(4) = 2 > 0$, innymi słowy, rozgałęzianie kosztuje więcej niż scalanie, to z tw. o rekursji uniwersalnej $U(k) = \mathcal{O}(k^{\log_2 4}) = \mathcal{O}(k^2)$; stąd że $T(n) = U(k)$, to $T(n) = \mathcal{O}(\log^2 n)$.

Zadanie 12. Ile wyniesie pesymistyczna złożoność algorytmu magicznych piątek, jeśli ciąg będziemy dzielić na podciągi trójelementowe zamiast na podciągi pięcioelementowe? Czy w optymistycznym przypadku algorytm będzie działał tak samo (asymptotycznie) jak w przypadku pesymistycznym?

Rozwiązanie (K. Kleczkowski). Przeanalizujmy algorytm mediany piątek i przełóżmy go na przypadek mediany „trójek".

Podstawową obserwacją w oszacowaniu czasu algorytmu magicznych piątek jest fakt, że spośród $n/5$ grup połowa z nich zawiera elementy mniejsze od mediany median. Ta sama ilość zawiera elementy większe, bądź równe od mediany median.

Zadanie 13. Zapisz w pseudokodzie algorytm Knutha-Morrisa-Prata. Załóż, że masz już wyliczoną funkcję π.

Zadanie 14. Podany na wykładzie algorytm obliczający minimalny koszt policzenia iloczynu n macierzy, o wymiarach odpowiednio $d_0 \times d_1, d_1 \times d_2, \ldots, d_{n-1} \times d_n$, wylicza pewne wartości $m_{i,j}$. Podaj ich definicje i sposób wyliczenia. Jeśli nie pamiętasz tego algorytmu, podaj inny algorytm o nie gorszej złożoności, który rozwiązuje ten problem.

Zadanie 15. Algorytm Karpa-Millera-Rosenberga (KMR) nadaje podsłowom tekstu i wzorca numery, by wyszukiwanie wzorca w tekście sprowadzić do sprawdzenia, czy w tekście występuje m-literowe podsłowo o numerze równym numerowi wzorca. Opisz, w jaki sposób algorytm KMR przypisuje numery podsłowom trzydziestoliterowym.

Zadanie 16. Chcemy dla danego grafu nieskierowanego (bez wag na krawędziach) o n wierzchołkach i m krawędziach obliczyć sumę odległości wszystkich wierzchołków od wyróżnionego wierzchołka (powiedzmy od wierzchołka o numerze 0). Opisz ideę optymalnego algorytmu rozwiązującego ten problem. Uzasadnij, że jest on optymalny.

Uwaga 16.1. Graf może mieć cykle. Jak nie umiesz rozwiązać dla grafu z cyklami, to będzie taka mniejsza bombka.

Uwaga 16.2. Tyle że nie ma już podziału na małe i duże bombki.

Rozwiązanie (Sushi). Zrobić BFSa, żeby policzyć odległości do wszystkich wierzchołków, a następnie je zsumować (albo sumować od razu w trakcie BFSa). $\mathcal{O}(|V| + |E|)$

Nie można lepiej, bo trzeba przejrzeć wszystkie wierzchołki i krawędzie - krawędzie dlatego, że gdybyśmy jakiejś nie rozpatrzyli, to mogłaby tworzyć krótszą ścieżkę do jakiegoś wierzchołka.

Zadanie 17. Porównaj następujące klasy funkcji:

(a) $\mathcal{O}((\log^* n)^{\log n})$,
(b) $\mathcal{O}((\log n)^{\log^* n})$,
(c) $\mathcal{O}(\log n)$.

Uwaga 17.1. Muszą być obliczenia i rozumowanie, nie może być wprost z definicji.

Rozwiązanie (K. Kleczkowski). Przypomnijmy więc, czym jest logarytm iterowany. Rekursywnie jest definiowany jako:

$$\log^* n = \begin{cases} 0 & \text{dla } n \leqslant 1 \\ 1 + \log^*(\log n) & \text{w p.w.} \end{cases} \tag{1}$$

Można również sformułować iteratywnie jako $\log^* n = \max\{k \in \mathbb{N} : \log^{(k)} n > 1\}$, gdzie $f^{(k)}$ to k-te złożenie funkcji f. Również, niech Φ_n będzie ciągiem danym rekursywnie:

$$\begin{cases} \Phi_0 = 1 \\ \Phi_{n+1} = 2^{\Phi_n} \end{cases} \tag{2}$$

Można zauważyć, że Φ jak i $\log^*$ są funkcjami wzajemnie odwrotnymi.

Policzmy granicę $\frac{\log n}{(\log^* n)^{\log n}}$ przy $n \to \infty$. Jeśli podciąg ma granicę, to ciąg również, stąd połóżmy $n = 2^k$. Przekształcając dla p.w. $k \in \mathbb{N}$:

$$\begin{aligned} \frac{\log n}{(\log^* n)^{\log n}} &= \frac{k}{(\log^*(2^k))^k} \\ &= \frac{k}{(1 + \log^* k)^k} \\ &\leqslant \frac{k}{2^k} \xrightarrow{k\to\infty} 0 \end{aligned}$$

bo $\log^* k \geqslant 1$ dla p.w. $k \in \mathbb{N}$. Czyli $\log n \in o\left((\log^* n)^{\log n}\right)$. Również możemy policzyć granicę $\frac{\log n}{(\log n)^{\log^* n}}$ przy $n \to \infty$. Połóżmy teraz $n = \Phi_k$.

$$\frac{\log n}{(\log n)^{\log^* n}} = \frac{\Phi_{k-1}}{\Phi_{k-1}^k}$$

Stąd, że $(\Phi_n)_{n\in\mathbb{N}}$ jest ciągiem rosnącym, to:

$$= \frac{1}{\Phi_{k-1}^{k-1}} \xrightarrow{k\to\infty} 0$$

Teraz należy porównać (a) i (b). Policzmy granicę wyrażenia $\frac{(\log^* n)^{\log n}}{(\log n)^{\log^* n}}$ przy $n \to \infty$. Połóżmy $n = \Phi_k$.

$$\begin{aligned} \frac{(\log^* n)^{\log n}}{(\log n)^{\log^* n}} &= \frac{k^{\Phi_{k-1}}}{\Phi_{k-1}^k} \\ &\leqslant \frac{k^{\Phi_{k-1}}}{1 + k(\Phi_{k-1} - 1)} \end{aligned}$$

Zadanie 18. Opisz ideę algorytmu Karpa-Rabina wyszukiwania wzorca w tekście.

Zadanie 19. Wyjaśnij, na czym polega operacja kaskadowego odcinania w kopcach Fibonacciego.

Zadanie 20. Podaj pseudokod optymalnego algorytmu tworzenia kopca (binarnego).