Algorytmy i struktury danych (Sortowanie i kopce)

Przyjmując, że $t1[] = 1, 2, 3, 4, 5, 6, 7$ oraz $t2[] = 7, 6, 5, 4, 3, 2, 1$ i stosując algorytmy sortujące ściśle wg procedur z pliku sorty2020.cc i wykonaj polecenia:

Zadanie 1 Ile dokładnie porównań (między elementami tablicy) wykona insertionSort(t2) a ile insertionSsort(t1)?

Posortowana: n-1 porównań. Nieposortowana: $\frac{n(n-1)}{2}$ porównań.
Dla t1 wykona 6 porównań. Dla t2 wykona 21 porównań.

Zadanie 2 Ile co najwyżej porównań (między elementami tablic) wykona procedura scalająca merge dwie tablice n-elementowe?

W najgorszym przypadku będzie musiała wykonać $2n - 1$ porównań, gdzie lementy rosnące wystepują naprzemiennie, więc będziemy porównywać każdy element do momentu aż nie trafimy na większy i go nie wpiszemy do tablicy wyjściowej.

Zadanie 3 Jaka jest pesymistyczna złożoność czasowa procedury mergeSort? Odpowiedź uzasadnij.

Dla wersji rekurencyjnej i iteracyjnej: $O(nlogn)$

Twierdzenie o rekurencji uniwersalnej: $T(n) = 2T(\frac{n}{2}) + n$,
gdzie liczba podproblemów to $a = 2$, rozmiar podproblemu to $b = 2$, i mamy n jako liczbę porównań.

$f(n) = \theta(n^{log_2 2}) = \theta(n)$

$T(n) = \theta(nlogn)$

Porównujemy elementy i scalamy je w jedną tablicę. W każdym kroku wykonujemy n porównań. W każdym kroku dzielimy tablicę na dwie części o połowie długości. W sumie wykonujemy $logn$ kroków, scalań. Złożoność czasowa zawsze jest $O(nlogn)$.

Zadanie 4 Ile co najwyżej porównań (między elementami tablicy) wykona procedura partition?

Przy jednym wywołaniu partition wykona się maksymalnie $n + 1$ porównań. To przypadek, gdy nasze odwrócone elementy są najbliżej pivota lub musimy zamienić stronami każdy z elementów. W obu przypadkach musimy każdą wartość po stronie prawej i lewej porównać z pivotem. Mimalnie są zawsz 2 porównania.

Zadanie 5 Jak jest średnia a jaka pesymistyczna złożoność quickSort. Odpowiedź uzasadnij.

Średnia złożoność czasowa: $O(nlogn)$

Pesymistyczna złożoność czasowa: $O(n^2)$

Pesymistyczną złożoność osiągamy, gdy jako pivot zostanie wybrana wartość najmniejsza lub największa w tablicy. Złożoność kwadratowa wynika ze wzoru

$T(n) = n + 1 + T(n-1) -> T(n) = \frac{n(n-1)}{2}$

czyli możemy posłużyć się sumą ciągu arytmetycznego.

$T(n) = O(n^2)$

Zadanie 6 Jaka jest złożoność funkcji buildheap? Przeprowadź dowód - uzasadnij swoją odpowiedź.

Złożoność buildheap wynosi $O(n)$.

Jeśli tablica ma n elementów to $\frac{n}{2}$ elementów jest liśćmi. Do porównań potrzebujemy węzła z dziećmi, więc ich nie mamy z czym porównywać. Pozosłe węzły możemy porównywać z dziećmi, a przy przesiewaniu z dalszymi potomkami. Stąd:

$2(\frac{n}{2} \cdot 0 + \frac{n}{4} \cdot 1 + \frac{n}{8} \cdot 2 + ...)$

Mnożenie przez dwa wynika z sytuacji gdy porównujemy oboje dzieci z rodzicem.

$$2\sum_{i=1}^{h-1} \frac{n}{2^{i+1}} \cdot i$$
$$2(\frac{n}{4} \cdot 1 + \frac{n}{8} \cdot 2 + \frac{n}{16} \cdot 3 + \frac{n}{32} \cdot 4 + \dots)$$
$$\frac{n}{2}(\frac{1}{1} + \frac{2}{2} + \frac{3}{4} + \frac{4}{8} + \dots)$$
$$\frac{1}{1} + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots = \frac{\frac{1}{1}}{1 - \frac{1}{2}} = 2$$
$$\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots = 1$$
$$\frac{1}{4} + \frac{1}{8} + \dots = 0.5$$
$$\frac{1}{8} + \dots = 0.25$$
$$\dots$$

$$\frac{n}{2} \cdot \frac{2}{1 - \frac{1}{2}} = \frac{n}{2} \cdot 4 = 2n = O(n)$$

Zadanie 7 Ile dodatkowej pamięci wymaga posortowanie tablicy n-elementowej za pomocą algorytmu:

a. mergesort, złożoność pamięciowa

b. quicksort

c. heapsort, złożoność pamięciowa O(n)
nie potrzebujemy dodatkowej pamięci -> O(1), ponieważ sortujemy w miejscu

d. insertionsort

e. countingsort

f. bucketsort

g. radixsort W punktach (e), (f), (g) zakładamy, że ilość kubełków jest m, a liczby do posortowania mają nie więcej niż k cyfr.

Zadanie 8 Jaka jest średnia a jaka pesymistyczna złożoność czasowa algorytmu:

a mergesort

b quicksort

c heapsort

d insertionsort

e countingsort

f bucketsort

g radixsort W punktach (e), (f), (g) zakładamy, że ilość kubełków jest m, a liczby do posortowania mają nie więcej niż k cyfr.

Zadanie 9 Udowodnij, że wysokość (ilość poziomów na których występują węzły) kopca n-elementowego wynosi $\lfloor log_2 n \rfloor + 1$

Zadanie 10

Zadanie 11 Czy ciąg 23, 17, 14, 6, 13, 10, 1, 5, 7, 12 jest kopcem?
23
/
17 14
/ /
6 13 10 1