Примерни решение на поправителния изпит по ДАА на 30.08.2022 г.

Зад. 1 Даден е масив $A[1,\ldots,n]$ от цели числа. Известно е, че съществува $k \in \{1,\ldots,n\}$, такова че

- за всеки i и j, такива че $1 \leq i < j \leq k$, в сила е $A[i] < A[j]$,
- за всеки i и j, такива че $k \leq i < j \leq n$, в сила е $A[i] > A[j]$.

Докажете, че $\Omega(\lg n)$ е долна граница за сложността на всеки алгоритъм, базиран на директни сравнения, който намира k. **(10 точки)**

Предложете алгоритъм със сложност $\Theta(\lg n)$, който намира k. Допустимо е да опишете алгоритъма на български, а не с псевдокод, стига описанието да е напълно ясно и недвусмислено. Съвсем накратко обосновете коректността и сложността по време на Вашия алгоритъм. **(20 точки)**

Решение: Какъвто и алгоритъм да ползваме, щом е базиран на директни сравнения, той поражда дърво T на вземане на решения. Съществуват n възможности за отговора: $k = 1$, $k = 2$, и така нататък, $k = n$. Ерго, T има поне n листа, инак не би могло да даде точно един (коректен) отговор. Щом T има поне n листа и разклоненост 2, за неговата височина h очевидно е в сила $h \geq \log_2 n$. Доказахме долната граница.

Ето един оптимален в асимптотичния смисъл алгоритъм, решаващ тази задача.

- Ако $n \leq 2$, намираме желаното k в константно време.
- Ако $n \geq 3$, изчисляваме $\texttt{mid} = \left\lceil \frac{n}{2} \right\rceil$. Спрямо тази стойност, масивът $A[1,\ldots,n]$ се разбива на "долна част" $A[1,\ldots,\texttt{mid}-1]$, $A[\texttt{mid}]$ и "горна част" $A[\texttt{mid}+1,\ldots,n]$. Това, че $\texttt{mid}-1, \texttt{mid}, \texttt{mid}+1 \in \{1,\ldots,n\}$, е очевидно. Също така е очевидно, че и долната, и горната част притежават свойството на $A[1,\ldots,n]$ да имат по точно един индекс, да речем съответно k' и k'', спрямо който са изпълнени съответните неравенства.

  Разглеждаме трите числа $A[\texttt{mid}-1]$, $A[\texttt{mid}]$ и $A[\texttt{mid}+1]$.

  - Ако $A[\texttt{mid}-1] < A[\texttt{mid}] > A[\texttt{mid}+1]$, връщаме $\texttt{mid}$.
  - Ако $A[\texttt{mid}-1] < A[\texttt{mid}] < A[\texttt{mid}+1]$, правим рекурсивно викане върху "горната част" $A[\texttt{mid}+1,\ldots,n]$.
  - Ако $A[\texttt{mid}-1] > A[\texttt{mid}] > A[\texttt{mid}+1]$, правим рекурсивно викане върху "горната част" $A[1,\ldots,\texttt{mid}-1]$.

  Поради свойствата на $A[1,\ldots,n]$, невъзможно е $A[\texttt{mid}-1] > A[\texttt{mid}] < A[\texttt{mid}+1]$, така че тези три възможности са изчерпателни и взаимно изключващи се.

  В първата възможност, $k = \texttt{mid}$, което очевидно следва от свойството на $A[1,\ldots,n]$. Във втората възможност, $k = k''$, понеже $A[1,\ldots,\texttt{mid}+1]$ е строго растящ. Огледално, в третата възможност $k = k'$, понеже $A[\texttt{mid}-1,\ldots,n]$ е строго намаляващ. Доказателството за коректност е по индукция по размера на входния масив: и във втория, и в третия случай, подмасивите, върху които викаме рекурсивно, са по-малки, така че съгласно индуктивното допускане, върнатото k' или k'' е коректно. Базата за $n = 1$ и $n = 2$ е очевидна.

  Сложността е $\Theta(\lg n)$, което се доказва по точно същия начин, по който се доказва за двоичното търсене.

Зад. 2 Представете си множество от отсечки $\{\ell_1, \ell_2, \ldots, \ell_n\}$. За всяка отсечка ℓ_i, единият ѝ край a_i е върху абсцисата $y = 0$, а другият ѝ край b_i е върху правата $y = 1$. Освен това, ако $1 \leq i < j \leq n$, то $a_i \neq a_j$ и $b_i \neq b_j$.

Вашата задача е да конструирате колкото можете по-бърз, в асимптотичния смисъл, алгоритъм, който връща броя на двойките отсечки, които се пресичат. Входът на алгоритъма се състои от два масива $A[1, 2, \ldots, n]$, който е някаква пермутация на a_1, a_2, $\ldots$, a_n, и $B[1, 2, \ldots, n]$, който е такава пермутация на b_1, b_2, $\ldots$, b_n, че $A[i]$ и $B[i]$ са краищата на ℓ_i, за $1 \leq i \leq n$. Примерно, ако $A = [3, 1, 2]$ и $B = [4, 3, 1]$, Вашият алгоритъм трябва да върне 1, защото има една единствена двойка пресичащи се отсечки: отсечката с краища $(1, 0)$ и $(3, 1)$ и отсечката с краища $(2, 0)$ и $(1, 1)$ се пресичат.

Няма нужда да пишете псевдокод! Достатъчно е да опишете алгоритъма по начин, който е ясен и недвусмислен. Съвсем накратко анализирайте коректността и сложността по време.

Решение: Първо сортираме отсечките по a_i-стойност. Това обаче не става с просто сортиране на масива A! Ако сортираме само A, ще "загубим отсечките". Един начин да постигнем сортирането на отсечките е всеки път, когато сортирането на A прави $\text{swap}(A[i], A[j])$, да прави и съответното $\text{swap}(B[i], B[j])$. При това размяната остава действие в константно време, така че сложността на сортирането остава $\Theta(n \lg n)$, ако се ползва бърза сортировка. В нашия пример, сортираният A е $[1, 2, 3]$, при което B става $[3, 1, 4]$.

След сортирането е в сила следното: за всеки i и j, такива че $1 \leq i < j \leq n$, ℓ_i пресича ℓ_j тстк $B[i] > B[j]$. Този факт е очевиден. В примера, който разглеждаме, единствената такава двойка индекси е $i = 1$ и $j = 2$: наистина, $B[1] = 3$ е по-голямо от $B[2] = 1$, и това отговаря на пресичането на отсечката с краища $(1, 0)$ и $(3, 1)$ и отсечката с краища $(2, 0)$ и $(1, 1)$.

И така, след сортирането задачата се свежда до това, да се преброи за колко i и j, такива че $1 \leq i < j \leq n$, е изпълнено $B[i] > B[j]$. Но това е същото като да се преброят инверсиите в B. На лекции сме изучавали как с модифициран MERGESORT може да бъдат преброени инверсиите във време $O(n \lg n)$.

Цялата сложност е $\Theta(n \lg n)$ за сортирането плюс още $\Theta(n \lg n)$ за преброяването на инверсиите, което общо прави $\Theta(n \lg n)$ за алгоритъма.

Зад. 3 Подготвяйки лекцията си по ДАА за алгоритъма на Дийкстра, доцент Алгоритмов получава блестяща идея: да направи алгоритъма на Дийкстра хем по-бърз, хем по-лесен за имплементиране, като замени приоритетната опашка с обикновена опашка FIFO (First-In, First-Out). Какво бихте казали за тази идея?

Решение: Идеята е ужасна. Ако заменим приоритетната опашка с обикновена опашка, алгоритъмът престава да е коректен. Ето контрапример: неориентиран пълен граф

$$G = (\{s, a, b, \}, \{(s, a), (s, b), (a, b)\})$$

с тегловна функция w, такава че $w(s, a) = 1$, $w(s, b) = 100$ и $w(a, b) = 1$. Започваме от s. Да кажем, че b влиза в опашката преди a. Но тогава b излиза от опашката преди a. Тогава стойността на b е 100 в края на работата на алгоритъма, а това е очевидно погрешно: най-късият път между s и b е с дължина 2, а не 100.

Зад. 4 Дефинирайте формално задачата 3SAT: какъв е общият екземпляр и какъв е въпросът? **(2 точки)**

Какво гласи теоремата на Cook? Не е необходимо да я доказвате, само напишете какво е твърдението. **(2 точки)**

Докажете, че 3SAT е **NP**-пълна. Можете да ползвате наготово теоремата на Cook. **(21 точки)**

*Ако докажете теоремата на Cook, ще получите **25 точки** бонус.*

Решение: Това е преподавано на лекции.