Примерни решения на изпита по ДАА на Информатика и КН 2 поток, проведен на 21 юни 2025 г.

Зад. 1 Даден е ориентиран граф $G = (V, E)$. Задачата е, ако G е цикличен, това да се установи по някакъв начин. Професор Дълбоков предлага следното решение:

- пуска се BFS във варианта, който обхожда целия граф;
- както знаем, всеки връх на графа в даден момент влиза в опашката и в някакъв следващ момент бива изваден от опашката;
- ако за поне един връх u, когато бъде изваден от опашката и алгоритъмът "минава" през неговия списък на съседство $\mathrm{adj}[u]$, ако алгоритъмът попадне на черен връх, то G е цикличен граф.

Какво бихте казали за тази идея?

Решение: Идеята е погрешна. Ето контрапример. Нека $G = (\{1, 2, 3, 4\}, \{(1, 2), (2, 3), (3, 4), (1, 4)\})$. Нека BFS започне обхождането от връх 1. На първа итерация, връх 1 излиза от опашката, а неговите деца влизат в опашката. Да кажем, че опашката е $(2, 4)$ и връх 2 е влязъл първо. На втора итерация връх 2 излиза от опашката. Той има единствено дете връх 3 и опашката става $(4, 3)$. На трета итерация връх 4 излиза от опашката. Той няма деца, нищо не влиза в опашката и връх 4 става черен. На четвърта итерация връх 3 излиза от опашката. В списъка на 3 е връх 4. Когато алгоритъмът разглежда списъка на 3, връх 4 вече е черен. Но G е ацикличен граф.

Зад. 2 На лекции видяхме реализация на функцията Extract Max чрез двоична пирамида, която работи във време $\Theta(\lg n)$. Професор Дълбоков публикува статия, в която предлага много по-сложна реализация на Extract Max, за която твърди, че работи във време

$$O\left(\left(\lg\lg\lg\left(\sum_{k=0}^{n}\binom{n}{k}^2\right)\right)^2\right)$$

Статията е 199 страници и Вие нямате никакво желание да я четете. Какво бихте казали за цитирания резултат за сложността по време, без да четете статията?

Решение Доказателството на професора за сложността по време трябва да е грешно, ако алгоритъмът реализира Extract Max чрез сравнения.

Първо да опростим израза за сложността. Както знаем от курса Дискретни Структури, в сила е тъждеството $\sum_{k=0}^{n}\binom{n}{k}^2 = \binom{2n}{n}$. Следователно, професорът твърди, че неговата Extract Max работи във време $O\left(\left(\lg\lg\lg\left(\binom{2n}{n}\right)\right)^2\right)$, което ще напишем като

$$O\left(\left(\lg\lg\lg\binom{2n}{n}\right)^2\right) \tag{1}$$

Както знаем от лекции по ДАА, в сила е $\binom{n}{\frac{n}{2}} \asymp \frac{2^n}{\sqrt{n}}$. Тогава $\binom{2n}{n} \asymp \frac{2^{2n}}{\sqrt{2n}}$. Но $\frac{2^{2n}}{\sqrt{2n}} \asymp \frac{4^n}{\sqrt{n}}$. Тогава $\lg\binom{2n}{n} \asymp \lg\left(\frac{4^n}{\sqrt{n}}\right) \asymp n\lg 4 - \frac{1}{2}\lg n \asymp \lg n$, така че изразът (1) се опростява до $O((\lg\lg n)^2)$.

Сега ще докажем, че сложността по време на Extract Max не може да бъде $O((\lg\lg n)^2)$. В сила е $(\lg\lg n)^2 = o(\lg n)$, което следва веднага от факта, че всяка полилогаритмична функция расте асимптотично по-бавно от всяка полиномиална функция. Ерго, ако професорът е прав, той разполага с реализация на Extract Max, която работи във време, строго по-малко, в асимптотичния смисъл, от логаритмично.

Последното е невъзможно. Алгоритъмът Heapsort в началото строи двоична пирамида в линейно време и след това по същество прави Extract Max n пъти. Ако имаше реализация на Extract Max със сложност $o(\lg n)$, базирана на сравнения, щяхме да имаме сортиращ алгоритъм със сложност $o(n\lg n)$, базиран на сравнения. Това, както знаем от лекции, е невъзможно.

Зад. 3 Изчислителната задача за разпознаване Задача X се дефинира така:

- екземпляр $(A, \mathcal{F}, k)$, където A е множество, $\mathcal{F}$ е фамилия над A, а $k \in \mathbb{N}^+$;
- въпрос: дали съществува $\mathcal{F}' \subseteq \mathcal{F}$, такава че $\mathcal{F}'$ е покриване на A и $|\mathcal{F}'| \leq k$?

Докажете, че Задача X е **NP**-пълна.

Решение: Тази задача е широко известна по името Set Cover, накратко SC.

Това, че SC принадлежи на класа на сложност **NP-c**, се доказва елементарно: ако $(A, \mathcal{F}, k)$ е ДА-екземпляр на SC, то недетерминираната машина на Тюринг отгатва подмножество $\mathcal{F}'$ на $\mathcal{F}$ и след това проверява, че $\mathcal{F}'$ наистина е подмножество на $\mathcal{F}$, че $\mathcal{F}'$ наистина покрива A и че мощността на $\mathcal{F}'$ наистина не надхвърля k. Тази проверка очевидно може да се извърши в полиномиално време.

Сега ще покажем, че $VC \leq_p SC$. Нека (G, k) е екземпляр на VC. Нека $G = (V, E)$. За всеки връх $v \in V$, нека S_v е множеството от ребрата, инцидентни с v. Конструираме екземпляр $(A, \mathcal{F}, \ell)$ на SC, където

- $A = E$,
- $\mathcal{F} = \{S_v \mid v \in V\}$,
- $\ell = k$.

Ще докажем, че (G, k) е ДА-екземпляр на VC тстк $(A, \mathcal{F}, \ell)$ е ДА-екземпляр на SC.

В едната посока, нека (G, k) е ДА-екземпляр на VC. Тогава съществува $U \subseteq V$, такова че за всяко ребро e от E, поне единият край на e се намира в U, и освен това $|U| \leq k$. Нека $\mathcal{F}' = \{S_v \mid v \in U\}$. Очевидно $|\mathcal{F}'| \leq k$. Ще покажем, че

$$\forall e \in E \;\; \exists X \in \mathcal{F}' \;:\; e \in X \tag{2}$$

Да допуснем противното. Противното е

$$\exists e \in E \;\; \forall X \in \mathcal{F}' \;:\; e \notin X$$

Но тогава нито един от краищата на e не се намира в U, противно на заключението от конструкцията. Тогава (2) е истина. Следователно, $\mathcal{F}'$ покрива множеството $A = E$.

В другата посока, нека $(A, \mathcal{F}, \ell)$ е ДА-екземпляр на SC. Тогава съществува $\mathcal{F}' \subseteq \mathcal{F}$, такава че $\mathcal{F}'$ е покриване на A и $|\mathcal{F}'| \leq k$. Да разгледаме множеството $U \subseteq V$, състоящо се от върховете, чрез които е конструирана подфамилията $\mathcal{F}'$. Очевидно $|U| \leq k$. Твърдим, че за всяко ребро $e \in E$, поне единият край на e се намира в U. Ако допуснем противното, веднага стигаме до противоречие: ако има ребро $e = (a, b)$, такова че $a, b \notin U$, то e не се съдържа в никое множество-елемент на $\mathcal{F}'$, така че $\mathcal{F}'$ не покрива $A = E$.

Доказахме, че че (G, k) е ДА-екземпляр на VC тстк $(A, \mathcal{F}, \ell)$ е ДА-екземпляр на SC. Това, че конструкцията може са се извърши в полиномиално време, е очевидно.

Зад. 4 Дадена е редица $X = (x_1, x_2, \ldots, x_n)$ от положителни числа и положително число L. Иска се редицата X да бъде разбита на минимален брой подредици, във всяка от които сумата от числата не надхвърля L. "Подредица" е всяка редица от вида $(x_i, x_{i+i}, \ldots, x_j)$, където $1 \le i \le j \le n$.

1 т. **А)** Формулирайте необходимо и достатъчно условие, за да има решение. Не се иска доказателство, а само условието.

1 т. **Б)** Какъв е минималният брой редици, които може да се получат, ако има решение? Не се иска доказателство.

1 т. **В)** Какъв е максималният брой редици, които може да се получат, ако има решение? Не се иска доказателство.

4 т. **Г)** Предложете линеен алгоритъм, който или връща разбиване на X на минимален брой подредици, такива че във всяка от тях сумата от числата не надхвърля L, или връща индикация, че такова разбиване няма. Съвсем накратко обосновете сложността по време. Имате пълна свобода за това, как да бъде представено разбиването на подредици, стига да е ясно и недвусмислено.

13 т. **Д)** Докажете коректността на Вашия алгоритъм.

Решение: За краткост дефинираме, че *конформна подредица* е всяка подредица, сумата от елементите на която не надхвърля L.

А) $\forall i \in \{1, \ldots, n\} : x_i \le L$.

Б) 1.

В) n.

Г) Следният алчен алгоритъм решава задачата.

```
GREEDY((x_1, ..., x_n), L)
 1  if max(x_1, ..., x_n) > L
 2      return "няма такова разбиване"
 3  m ← 1, current ← x_1, part[1] ← 1
 4  for i ← 2 to n
 5      if current + x_i > L
 6          m ++
 7          part[m] ← i
 8          current ← x_i
 9      else
10          current ← current + x_i
11  return (part, m)
```

Сложността по време е линейна, защото по същество този алгоритъм изпълнява два цикъла (редове 1–2 и 4–10), като всеки цикъл е с линеен брой итерации и всяка итерация се изпълнява в константно време.

Д) Ще докажем, че GREEDY връща разбиване $\mathcal{U}$ на X на минимален брой конформни подредици. Числото m от изхода има смисъла на броя на редиците в $\mathcal{U}$, а масивът part$[1 \,..\, m]$ от изхода реализира $\mathcal{U}$ по следния начин: part[k], за $k \in \{1, \ldots, m\}$, е началото на k-тата редица в $\mathcal{U}$. Нека $\mathcal{U} = (s_1, \ldots, s_m)$. Очевидно s_1 започва с x_1 и s_m завършва с x_n.

Да допуснем, че съществува оптимално разбиване $\mathcal{V} = (z_1, \ldots, z_\ell)$ на X на конформни подредици, такова че $\ell < m$. Явно z_1 започва с x_1 и z_ℓ завършва с x_n. Нека максималният общ префикс на $\mathcal{U}$ и $\mathcal{V}$ е с дължина t, като $0 \le t < \ell$. С други думи, $s_1 = z_1$, $\ldots$, $s_t = z_t$, но $s_{t+1} \ne z_{t+1}$. Забележете, че $t < \ell$, защото $\mathcal{U}$ и $\mathcal{V}$ все някъде (в някой елемент с един и същи индекс $t+1$) трябва да започнат да се различават, ако ги гледаме отляво надясно.

БОО, нека $\mathcal{V}$ е оптимално разбиване, което има максимален общ префикс с $\mathcal{U}$.

Щом $s_t = z_t$, редиците s_{t+1} и z_{t+1} започват с един и същи елемент от X, да го наречем x_q. Обаче s_{t+1} и z_{t+1} завършват с различни елементи от X, инак биха били една и съща подредица на X. Да кажем, че $s_{t+1} = (x_q, \ldots, x_{q'})$ и $z_{t+1} = (x_q, \ldots, x_{q''})$. Ключовото наблюдение е, че $q'' < q'$, защото Greedy, започвайки от даден елемент на X, в случая x_q, слага в подредицата, започваща с него, максималният възможен брой елементи, чиято сума не надхвърля L. Ерго, ако допуснем, че $q'' > q'$, веднага достигаме противоречие: подредицата $(x_q, \ldots, x_{q''})$ би имала сума, по-голяма от L.

Но тогава можем да модифицираме $\mathcal{V}$, като прехвърлим в z_{t+1} всички елементи от $x_{q''+1}$ включително до $x_{q'}$ включително. При това сумата на числата в z_{t+1} ще остане не по-голяма от L, а на практика z_{t+1} ще стане същото като s_{t+1}. Останалите редици-елементи на $\mathcal{V}$ ще останат конформни. По този начин ще получим разбиване на X на конформни подредици, което има общ префикс с $\mathcal{U}$ с дължина $t+1$. Което директно противоречи на прежде направеното допускане, че $\mathcal{V}$ е оптимално разбиване, което има максимален общ префикс с $\mathcal{U}$. □