Задачи за второ *домашно*[*] по Дизайн и анализ на алгоритми, спец. Компютърни науки, курс 2, поток 1

Задача 1.

В тази задача разглеждаме само крайни детерминирани автомати, в които всяко състояние е достижимо от началното и от всяко състояние има път до финално.

За краен детерминиран автомат $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ с $w_{\mathcal{A}}(n)$ бележим броя на думите от езика $\mathcal{L}(\mathcal{A})$ с дължина не по-голяма от n, тоест:

$$w_{\mathcal{A}}(n) = |\{\alpha \in \Sigma^* \mid |\alpha| \leq n \& \alpha \in \mathcal{L}(\mathcal{A})\}|$$

1. Нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ е краен детерминиран автомат. Да се докаже, че ако в $\mathcal{A}$ има състояния $p, q \in Q$, които участват в различни прости цикли и q е достижимо от p, то $w_{\mathcal{A}}(n) \in \Omega(c^n)$ за някоя константа $c > 1$.

2. Нека $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$ е краен детерминиран автомат. Да се докаже, че ако в $\mathcal{A}$ няма състояния $p, q \in Q$, които участват в различни прости цикли и q е достижимо от p, то $w_{\mathcal{A}}(n) \in O(P(n))$ за някой полином P.

3. Да се предложи алгоритъм със сложност $O(|Q||\Sigma|)$, който решава следния проблем ExponentialGrowth:

   `Вход:` $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$`, краен детерминиран автомат`
   `Изход: Да, ако` $w_{\mathcal{A}}(n) \in \Omega(c^n)$ `за някоя константа` $c > 1$ `и не, иначе.`

4. Да се предложи алгоритъм със сложност $O(n|Q||\Sigma|)$, който решава следния проблем NumberOfWords:

   `Вход:` $\mathcal{A} = \langle \Sigma, Q, s, \delta, F \rangle$`, краен детерминиран автомат и`
   `естествено число` $p > 2$`,`
   `Изход:` $w_A(n) \pmod{p}$`.`

[*] Задачите не се предават, в частност от оценъчна гледна точка няма значение дали са прочетени или не, дали са решени или не, дали са написани или не. За това, как ще бъде проверено домашното, може да прочете в документа "за оценяване" на страницата на курса.

Задача 2.

Дъска $n \times n$, част от чиито полетата са маркирани, ще представяме с матрица $A[1..n][1..n]$ от нули и единици, като:

$$A[i][j] = \begin{cases} 1, & \text{полето } (i,j) \text{ е маркирано} \\ 0, & \text{иначе.} \end{cases}$$

Област за такава дъска ще наричаме максимално по включване множество от клетки, което може да бъде обходено от шахматен цар, без да стъпва на маркирана клетка.

1. Да се предложи алгоритъм с времева сложност $O(n^2)$, който решава проблема Regions:

   `Вход:`$A[1..n][1..n]$ `представяне на дъска с маркирани клетки`
   `Изход:`r `брой различни области на дъската представена от` A.

2. Да се предложи алгоритъм с времева сложност $O(n^2)$, който решава проблема MoreRegions:

   `Вход:`$A[1..n][1..n]$ `представяне на дъска с маркирани клетки`
   `Изход:`$C[1..m][1..2]$, `където` $(C[i][1], C[i][2])$ `са всички клетки,`
   `за които ако се маркира и тя,`
   `броят на областите ще се увеличи.`

3. Да се предложи алгоритъм с времева сложност $O(n^2)$, който решава проблема OneMoreRegions:

   `Вход:`$A[1..n][1..n]$ `представяне на дъска с маркирани клетки`
   `Изход:`$C[1..m][1..2]$, `където` $(C[i][1], C[i][2])$ `са всички клетки,`
   `за които ако се маркира и тя,`
   `броят на областите ще се увеличи с точно` 1.

Задача 3.

Ще казваме, че граф $G = (V, E)$ е *почти хубав*, ако G е свързан, но премахването на кой да е прост цикъл в G с нечетна дължина от G го прави несвързан.

Казваме, че $c : V \to \{1, \dots, d\}$ е *оцветяване в d цвята* на върховете на граф $G = (V, E)$, ако за всяко ребро $\{u, v\} \in E$ е изпълнено, че $c(u) \neq c(v)$.

1. Нека $G = (V, E)$ е почти хубав граф, а $A \dot\cup B = V$ е разбиване на върховете на графа G, за което броят на ребрата, свързващи връх от A с връх от B е възможно най-голям. Да се докаже, че ако

   $$E_A = \{\{a_1, a_2\} \in E \,|\, a_1, a_2 \in A\},$$

   то графът (A, E_A) е двуделен.

2. Да се предложи алгоритъм с времева сложност $O(m^2)$, който решава проблема 4Colours:

   Вход: $G = (V, E)$ почти хубав граф с $V = \{1, \dots, n\}$ и $|E| = m$
   Изход: $c[1..n]$ оцветяване на G в 4 цвята.

Задача 4.

Нека $\Sigma = \{1, \dots, \sigma\}$, със $\sigma \in \mathbb{N}$, $\sigma > 0$, е азбука. С 0 ще бележим празната дума. За думи $v, w \in \Sigma^*$, казваме, че $\alpha = (a_1, b_1)(a_2, b_2) \dots (a_m, b_m)$, където $a_i, b_i \in \Sigma \cup \{0\}$, е *подравняване* на v и w, ако:

$$v = a_1 \cdot a_2 \cdot \dots \cdot a_m \text{ и } w = b_1 \cdot b_2 \cdot \dots \cdot b_m.$$

Ако $c : (\Sigma \cup \{0\}) \times (\Sigma \cup \{0\}) \to \mathbb{N}$, дефинираме *цена* на подравняването α като:

$$cost_c(\alpha) = \sum_{i=1}^{m} c((a_i, b_i)), \text{ където } \alpha = (a_1, b_1)(a_2, b_2) \dots (a_m, b_m).$$

Разглеждаме проблема EditDistance:

Вход: $v[1..k], w[1..n] \in \Sigma^*$ и $c[0..\sigma][0..\sigma]$ двумерен масив от естествени числа
Изход: $dist(v, w) = \min\{cost_c(\alpha) \mid \alpha$ е подравняване на v и $w\}$

1. Да се предложи алгоритъм с времева сложност $O(nk)$, който решава EditDistance.

2. Да се предложи алгоритъм с времева сложност $O(nk)$, който решава EditDistance и намира примерно подравняване α, което има възможно най-малка цена.

3. Да се предложи алгоритъм с времева сложност $O(nk(n + k))$, който решава решава EditDistance и измежду всички подравнявания, които имат възможно най-малка цена, намира такова, което използва най-малко операции от вида $\Sigma \times \{0\} \cup \{0\} \times \Sigma$.

Задача 5.

Разглеждаме пълен граф $G = (V, E)$ с върхове $V = \{1, 2, \dots, n\}$ и инективна ценова функция $c : E \to \mathbb{Q}$.

1. Нека $\pi_1, \pi_2, \dots, \pi_n$ са n пътя в G, за които $\pi_i : v_i \to_G^* i$ и никой път не използва едно ребро повече от веднъж. Да се докаже, че ако $c(e) > c(\{i, j\})$ за всяко ребро e, което се среща в пътищата $\pi_1, \pi_2, \dots, \pi_n$, то:

   $$\pi'_j = \pi_i \circ (j) \text{ и } \pi'_i = \pi_j \circ (i),$$

   са пътища, които завършват в j и i съответно и използват всяко ребро не повече от веднъж.

2. Да се докаже, че в G има път $\pi = (v_1, v_2, \ldots, v_n)$, за който $c(\{v_i, v_{i+1}\}) > c(\{v_{i+1}, v_{i+2}\})$ за всяко $i < n-1$.

3. Да се предложи алгоритъм с времева сложност $O(n^2 \log n)$, който решава следния проблем:

   `Вход:` $C[1..n][1..n]$ `матрица с цените на ребрата` $\{i,j\}$`, като` $C[i][j] = C[j][i]$ `и` $C[i][i] = 0$
   `Изход:` $v[1..n]$ `редица от цели числа между` 1 `и` n`, за които` $C[v[i]][v[i+1]] < C[v[i+1]][v[i+2]]$ `за` $i < n-1$`.`

Задача 6.

За матрица $A \in M_{n,n}(\mathbb{Q})$ разглеждаме следните операции:

C_j избираме стълб j на матрицата A и сменяме знаците на всички елементи в j-ия стълб на A.

R_j избираме ред i на матрицата A и сменяме знаците на всички елементи в i-ия ред на A.

1. Да се докаже, че ако никоя от операциите R_i и C_j не увеличава сумата на елементите на матрица A, то сумата от елементите във всеки ред и във всеки стълб на A е неотрицателен.

2. Да се докаже, че има редица от не повече от $2n$ от позволените операции, след които сумите от елементите на A във всеки ред и всеки стълб са неотрицателни.

3. Да се предложи алгоритъм с времева сложност $O(n^3 L)$, който решава проблема ChangeSigns:

   `Вход:` $A[1..n][1..n]$ `матрица от цели числа, ненадминаващи по модул` L`,`
   `Изход:` $C[1..2n]$ `масив от операции` C_j `и` R_i`, след които сумата от елементите на` A `във всеки ред и всеки стълб са неотрицателни`

Задача 7.

Нека $\Gamma = \langle \Sigma, \mathcal{N}, S, P \rangle$ е контекстносвобдна граматика.

1. Ако $\mathcal{N}_{\neq\emptyset}$ е множеството от нетерминали, които извеждат поне една дума от Σ^*, да се докаже, че:

$$\begin{aligned}
\mathcal{N}_{\neq\emptyset} &= \mathcal{N}_{\neq\emptyset}^{(n)} \text{ за } n = |\mathcal{N}|, \text{ където} \\
\mathcal{N}_{\neq\emptyset}^{(0)} &= \emptyset \\
\mathcal{N}_{\neq\emptyset}^{(k+1)} &= \mathcal{N}_{\neq\emptyset}^{(k)} \cup \{A \in \mathcal{N} \mid \exists \gamma \in (\mathcal{N}_{\neq\emptyset}^{(k)} \cup \Sigma)^* (A \to \gamma \in P)\}.
\end{aligned}$$

2. Ако $\Sigma = \{1, 2 \ldots, k\}$, $\mathcal{N} = \{k+1, k+2, \ldots, k+n\}$. Да се опише представяне на множеството от правила P на граматика с терминали Σ и нетерминали $\mathcal{N}$.

3. При така избраното представяне, да се предложи линеен алгоритъм, който решава следния проблем:

   `Вход:`$\Gamma = \langle \Sigma, \mathcal{N}, S, P \rangle$ `контекстносвободна граматика`
   `Изход:`$\mathcal{N}_{\neq\emptyset}$`.`

4. Ако $\mathcal{N}_\varepsilon$ е множеството от нетерминали, които извеждат ε, да се докаже, че:

$$\begin{aligned} \mathcal{N}_\varepsilon &= \mathcal{N}_\varepsilon^{(n)} \text{ за } n = |\mathcal{N}|, \text{ където} \\ \mathcal{N}_\varepsilon^{(0)} &= \emptyset \\ \mathcal{N}_\varepsilon^{(k+1)} &= \mathcal{N}_\varepsilon^{(k)} \cup \{A \in \mathcal{N} \mid \exists \gamma \in (\mathcal{N}_\varepsilon^{(k)})^* (A \to \gamma \in P)\}. \end{aligned}$$

5. При избраното по-горе представяне на контекстносвободни граматики, да се предложи линеен алгоритъм, който решава следния проблем:

   `Вход:`$\Gamma = \langle \Sigma, \mathcal{N}, S, P \rangle$ `контекстносвободна граматика`
   `Изход:`$\mathcal{N}_\varepsilon$`.`

6. Да се докаже, че за всяка контекстносвободна граматика с нетерминали $\mathcal{N}$, релацията $\sim \subseteq \mathcal{N} \times \mathcal{N}$, за която:

$$A \sim B \iff A \Rightarrow_\Gamma^* B \& B \Rightarrow_\Gamma^* A$$

   е релация на еквивалентност.

7. Да се предложи линеен алгоритъм, който решава следния проблем:

   `Вход:`$\Gamma = \langle \Sigma, \mathcal{N}, S, P \rangle$ `контекстносвободна граматика, за която`
   `ако` $A \to \gamma \in P$`, то` $|\gamma| \leq 2$
   `Изход:`$\{[A]_\sim \mid A \in \mathcal{N}\}$

Задача 8.

Остап Бендер се намира пред обменно бюро в непозната страна. Той има 1 долар в джоба си и вижда обменните курсове на таблото на обменното бюро като между някои валути се извършват трансакции само в едната посока, а между други изобщо не се извършват.

Разбира се, първият въпрос на Вечния комбинатор е дали може да фалира обменното бюро. Едва вторият би бил колко най-много пари от местната валута може да си купи с наличния му 1 долар. Помогнете на Остап Бендер[1] да си отговори на тези житейски въпроси.

[1]Не се надявайте на комисионна, но пък кой знае...

1. Нека наличните валути са номерирани $\{1, 2, \ldots, n\}$, като доларите са с индекс 1. Обменните курсове представляват наредени тройки $\langle buy, sell, rate\rangle$, където buy е индексът на валутата, която бюрото купува, $sell$ – индексът на валутата, която бюрото продава, $rate$ – каква стойност изплаща бюрото от валутата $sell$ за единица валута buy.

   Да се предложи алгоритъм с времева сложност $O(mn)$, който решава следния проблем:

   `Вход:`$ExRates[1..m]$`, масив от тройки `$\langle buy, sell, rate\rangle$
   `Изход:Не, ако при положение, че бюрото има достатъчна наличност`
   `от всяка от валутите, Остап Бендер не може да фалира обменното бюро.`
   `Схема `$s : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$`, примерна схема за това, как`
   `Остап Бендер да фалира обменното бюро, иначе.`

2. При уговорките от предишната подточка, да се предложи алгоритъм с времева сложност $O(mn)$, който решава следния проблем:

   `Вход:`$ExRates[1..m]$`, масив от тройки `$\langle buy, sell, rate\rangle$
   `Изход:`$money[1..m]$` -- максималната стойност от всяка една от валутите, която`
   `Остап Бендер може да си купи с наличния 1$, ако не може да фалира бюрото.`

Задача 9.

За пермутация $\pi \in S_n$, която е произведение на k независими цикъла $\pi = C_1 \circ C_2 \cdots \circ C_k$,*най-малко общо кратно* на π наричаме най-малкото общо кратно на дължините на циклите $C_1, \ldots, C_k$:

$$lcm(\pi) = lcm(|C_1|, |C_2|, \ldots, |C_k|).$$

1. Да се докаже, че ако $lcm(\pi) = \prod_{j=1}^{s} p_j^{\alpha_j}$ е каноничното разлагане на $lcm(\pi)$, то има пермутация $\pi' \in S_n$, за която $lcm(\pi') = lcm(\pi)$ и дължината на всеки цикъл в π' е $p_j^{\beta_j}$ за някое $\beta_j \leq \alpha_j$.

2. Нека $\pi^* \in S_n$ е пермутация, за която $lcm(\pi^*)$ е възможно най-голямо като всеки цикъл на π^* има дължина точна степен на просто число. Да се докаже, че за всяко просто p и всяко j, за което $p^j | lcm(\pi^*)$, но $p^{j+1} \not| \, lcm(\pi^*)$, π^* има най-много $(p-1)$ цикъла с дължина p^j.

3. Да се докаже, че ако $\pi^* \in S_n$ е пермутация, за която $lcm(\pi^*)$ е възможно най-голямо, то има и пермутация $\pi^{**} \in S_n$, за която:

   - $lcm(\pi^{**}) = lcm(\pi^*)$,
   - всички цикли в π^{**} имат дължини степени на просто число,
   - ако p е просто и $p^j | lcm(\pi^{**})$, то π^{**} има не повече 1 цикъл с дължина p^j.

4. Да се предложи алгоритъм, който при машинна дума с размер $\Theta(n \log n)$, с времева сложност $O(n^2 \log n)$ решава проблема Landau:

   `Вход:`n
   `Изход:`$\max\{lcm(\pi) \mid \pi \in S_n\}$

Задача 10.

Аритметичен граф ще наричаме (G, λ, ν), където $G = (V, E)$ е ориентиран граф, а $\lambda : E \to \{+, \times, \mod\}$ и $\nu : V \to \mathbb{N}$ са функции.

За път π в аритметичен граф (G, λ, ν), *стойност на* π ще наричаме $val(\pi) \in \mathbb{N}$, където:

$$val(\pi) = \begin{cases} \nu(v), \text{ ако } \pi = (v) \\ val(\pi')\lambda(u,v)\nu(v), \text{ ако } \pi = \pi \circ (u,v). \end{cases}$$

1. Да се предложи алгоритъм със сложност $O(|V| + |E|)$, който решава InfiVal:

   `Вход:`(G, λ, ν) `аритметичен граф с` $G = (V, E)$.
   `Изход:Да, ако` $\{val(\pi) \mid \pi$ `е път в` $G\}$ `е безкрайно.`
   `Не, иначе.`

2. Да се предложи алгоритъм със сложност $O(k \sum_{v \in V} \nu(v)(|V| + |E|))$, който решава Representable:

   `Вход:`(G, λ, ν) `аритметичен граф с` $G = (V, E)$ `и` $k \in \mathbb{N}$.
   `Изход:Да, ако има път` π `в` G`, така че` $val(\pi) = k$.
   `Не, иначе.`

3. Да се предложи алгоритъм със сложност $O(k \sum_{v \in V} \nu(v)(|V| + |E|))$, който решава FindRepresentation:

   `Вход:`(G, λ, ν) `аритметичен граф с` $G = (V, E)$ `и` $k \in \mathbb{N}$.
   `Изход:`π `примерен път в` G`, така че` $val(\pi) = k$`, ако такъв има.`
   $()$`, иначе.`

Задача 11.

За масив от цели числа $A[1..n]$ и индекси $1 \le i \le j \le n$ с $\mu_A(i, j)$ и $\sigma_A(i, j)$ бележим:

$$\mu_A(i,j) = \frac{1}{j - i + 1} \sum_{k=i}^{j} A[k] \text{ и } \sigma_A(i,j) = \frac{1}{j - i + 1} \sqrt{\sum_{k=i}^{j} \sum_{\ell=k}^{j} (A[k] - A[\ell])^2}.$$

1. Да се предложат алгоритъм PREPROCESS и QUERY със следните свойства:

   - PREPROCESS($A[1..n]$) има времева сложност $O(n)$.
   - Ако D е резултатът от PREPROCESS($A[1..n]$), то за всеки $1 \leq i \leq j < n$ е изпълнено, че:

   $$\text{QUERY}(D, i, j) = (\mu_A(i, j), \sigma_A^2(i, j))$$

   и QUERY има сложност $O(1)$.

2. За масив $A[1..n]$ и индекси $1 \leq i \leq j \leq n$ с $max(i, j)$ бележим:

   $$max_A(i, j) = \begin{cases} \max_{i \leq k \leq j} \frac{A[k] - \mu_A(i,j)}{\sigma_A(i,j)}, \text{ ако } \sigma_A(i, j) > 0 \\ 0, \text{ иначе.} \end{cases}$$

   Да се предложи алгоритъм със сложност $O(n^2)$, който решава следния MAXMAX проблем:

   `Вход:`$A[1..n]$ `масив от числа,`
   `Изход:`(i, j)`, така че` $1 \leq i \leq j \leq n$ `и` $max_A(i, j)$ `е възможно най-голямо.`

Задача 12.

Нека $\Sigma = \{0, 1, \ldots, \sigma\}$ е фиксирана азбука. *Кореново дърво с етикети по ребрата* ще наричаме двойката (T, λ), където $T = (V, E, r)$ е кореново дърво, а $\lambda : E \to \Sigma$ е функция. Ще казвваме, че λ задава еднозначни етикети, ако:

$$\forall (u_1, v_1) \in E \forall (u_2, v_2) \in E(u_1 = u_2 \& \lambda(u_1, v_1) = \lambda(u_2, v_2) \Rightarrow v_1 = v_2).$$

1. Казваме, че кореново дърво с етикети $((V_1, E_1, r_1), \lambda_1)$ е изоморфно на кореново дърво с етикети $((V_2, E_2, r_2), \lambda_2)$ ако има биекция $f : V_1 \to V_2$, за която:

   $$f(r_1) = r_2 \text{ и } \forall (u_1, v_1) \in E_1(\lambda_1(u_1, v_1) = \lambda_2(f(u_1), f(v_1))).$$

   Да се докаже, че ако $T_1 = ((V_1, E_1, r_1), \lambda_1)$ и $T_2 = ((V_2, E_2, r_2), \lambda_2)$ са дървета с етикети по ребрата, като λ_1 задава еднозначни етикети, то тогава има най-много една биекция $f : V_1 \to V_2$, която задава изоморфизъм между T_1 и T_2.

2. Да се предложи алгоритъм с времева сложност $O(|V_1| + |V_2|)$, който решава проблема DETLABISOTREES:

   `Вход:`$T_1 = ((V_1, E_1, r_1), \lambda_1)$`,` $T_2 = ((V_2, E_2, r_2), \lambda_2)$ `коренови дървета с етикети, като` λ_1 `задава еднозначни етикети.`
   `Изход:`$f : V_1 \to V_2$ `изоморфизъм между` T_1 `и` T_2`, ако такъв има и` $\emptyset$ `ако няма такъв изоморфизъм.`