Теми за Държавен Изпит

29 юли 2024 г.

Дискретни структури

Множества. Декартово произведение. Релации. Функции.

Аксиома (за обема)
$\forall x(x \in A \iff x \in B) \implies A = B$

Аксиома (за отделянето)
$A' = \{x | x \in A, \pi(x)\}$ е множество

Аксиома (за степенното множество)
Съвкупността от всички подмножества на множеството X е множество

Аксиома (за индукцията)
$X =< X_0, \mathcal{F} >$ е множество, където X_0 е непразно множество, а $\mathcal{F}$ е множество от операции

Теорема (математическа индукция)
За всеки елемент x на индуктивно дефинираното мн-во M е в сила $\pi(x)$

Доказателство.

- База: За всеки базов елемент x_0 от M_0 проверяваме верността на $\pi(x_0)$
- Индукционно предположение: Допускаме, че $\pi(x)$ е в сила за всеки елемент x, включен до определен момент в множеството M
- Индукционна стъпка: Показваме, че при направеното предположение, за всеки построен с помощта на операциите от $\mathcal{F}$ елемент y на M, също е в сила $\pi(y)$
- Заключение: π е в сила за всеки елемент на множеството M

□

Основни операции върху множества и техните свойства:

$$A \cup B = \{x | x \in A \text{ или } x \in B\}$$

$$A \cap B = \{x | x \in A \text{ и } x \in B\}$$

$$A \backslash B = \{x | x \in A \text{ и } x \notin B\}$$

$$A \triangle B = \{x | (x \in A \text{ и } x \notin B) \text{ или } (x \notin A \text{ и } x \in B)\}$$

1. Идемпотентност - $A \cup A = A, A \cap A = A$
2. Комутативност - $A \cup B = B \cup A, A \cap B = B \cap A, A \triangle B = B \cup A$
3. Асоциативност - $(A \cup B) \cup C = A \cup (B \cup C), (A \cap B) \cap C = A \cap (B \cap C), (A \triangle B) \triangle C = A \triangle (B \triangle C)$
4. Дистрибутивност - $(A \cup B) \cap C = (A \cap C) \cup (B \cap C), (A \cap B) \cup C = (A \cup C) \cap (B \cup C)$
5. Свойства на празното и универсалното множество - $A \cup \varnothing = A, A \cap \varnothing = \varnothing, A \cup U = U, A \cap U = A$
6. Свойства на допълнението - $A \cup \overline{A} = U, A \cap \overline{A} = \varnothing, \overline{\overline{A}} = A$
7. Закони на Де Морган - $\overline{A \cup B} = \overline{A} \cap \overline{B}, \overline{A \cap B} = \overline{A} \cup \overline{B}$

Дефиниция Нека a и b са произволни елементи. Означаваме с (a, b) множеството $\{a, \{a, b\}\}$ и го наричаме **наредена двойка** от елементите a и b

Дефиниция Нека A и B са множества. Множеството $A \times B = \{(a, b) | a \in A, b \in B\}$ наричаме **Декартово произведение** на множествата A и B

Дефиниция Нека $n \in \mathbb{N}$ и $A_1, ..., A_n$ е фамилия от множества. Всяко $R \subseteq A_1 \times ... \times A_n$ наричаме **n-местна релация** над декартовото произведение $A_1 \times ... \times A_n$ или просто **n-местна релация**. Множествата $A_1, ..., A_n$ наричаме домейни. В случай, че $n = 2$, релацията наричаме двуместна, или просто **релация**.

Дефиниция Нека $R \subseteq A \times A$. Казваме, че релацията R е:

- **Рефлексивна** - ако $\forall a \in A, (a, a) \in R$
- **Симетрична** - ако $\forall a, b \in A, a \neq b, (a, b) \in R \implies (b, a) \in R$
- **Транзитивна** - ако $\forall a, b, c \in A, (a, b) \in R, (b, c) \in R \implies (a, c) \in R$
- **Антирефлексивна** - ако $\forall a \in A, (a, a) \notin R$
- **Антисиметрична** - ако $\forall a, b \in A, a \neq b, (a, b) \in R \implies (b, a) \notin R$
- **Силно антисиметрична** - ако $\forall a, b \in A, a \neq b$, точно едно от $(a, b) \in R$ или $(b, a) \in R$ е в сила

Дефиниция Релацията $R \subseteq A \times A$ наричаме **еквивалентност**, ако е рефлексивна, симетрична и транзитивна.

Дефиниция Нека $R \subseteq A \times A$ е релация на еквивалентност, $a \in A$ и $[a] = \{b | b \in A, aRb\}$. Нека фамилията $\mathcal{R} = \{[a_i] | i \in I\}$ се състои от всички различни множества $[a_i]$, където I е подходящо индексно множество. Тогава $\mathcal{R}$ е разбиране на A. Множествата $[a_i]$ наричаме **класове на еквивалентност**.

Дефиниция Релацията $R \subseteq A \times A$ наричаме **частична наредба**, ако е рефлексивна, антисиметрична и транзитивна.

Дефиниция Частичната наредба $R \subseteq A \times A$ наричаме **пълна(линейна)**, ако силно антисиметрична.

Дефиниция Нека $R \subseteq A \times A$. Елементът $a \in A$ наричаме **минимален** в R, ако $\nexists b \in A, b \neq a$ и такъв, че bRa. Елементът $a \in A$ наричаме **максимален** в R, ако $\nexists b \in A, b \neq a$ и такъв, че aRb.

Алгоритъм (топологично сортиране)

Дадено: Множество $A = \{a_1, a_2, ..., a_n\}$ и частична наредба $R \subseteq A \times A$.
Резултат: Наредена n-торка $(a_{i_1}, a_{i_2}, ..., a_{i_n})$ от различните елементи на A, задаваща пълна наредба $R' \subseteq A \times A, R \subseteq R' : \forall m \in I_n, \forall k \in I_n, m \leqslant k \to (a_{i_m}, a_{i_k}) \in R'$.
Процедура:

1. $j = 1, T = R, M = A$
2. Намираме a_{i_j} - минимален в M за релацията T
3. $T = T \backslash \{(a_{i_j}, a_l) | (a_{i_j}, a_l) \in T\}; M = M \backslash \{a_{i_j}\}; j = j + 1$
4. Ако $M \neq \varnothing$ преминаваме към 2., в противен случай алгоритъмът спира.

Дефиниция Релацията $f \subseteq X \times Y$ наричаме **частична функция**, ако $\forall a \in X \exists$ не повече от едно $b \in Y : (a, b) \in f$

Дефиниция Релацията $f \subseteq X \times Y$ наричаме **тотална функция**, ако $\forall a \in X \exists$ точно едно $b \in Y : (a, b) \in f$

Дефиниция Ако $f(x_1) \neq f(x_2), \forall x_1 \neq x_2 \in X$, тогава функцията f наричаме **инекция**.

Дефиниция Ако $\forall b \in Y, \exists a \in X, f(a) = b$, тогава функцията f наричаме **сюрекция** на X върху Y.

Дефиниция Ако тоталната функция $f : X \to Y$ е инекция и сюрекция едновременно, казваме, че f е **биекция**.

Дефиниция Множеството A е **крайно**, ако $A = \varnothing$ или $\exists n \in N, n \geqslant 1$ и биекция $f : A \to I_n$ $|A| = 0$, ако $A = \varnothing$ и $|A| = n$ в противен случай, наричаме брой на елементите на A (**кардиналност**).

Дефиниция Казваме, че множеството A е **изброимо безкрайно**, ако $\exists$ биекция $f : A \to N$. Казваме, че множеството A е **изброимо**, ако е крайно или изброимо безкрайно.

Принцип на Дирихле Нека X и Y са крайни множества и $|X| > |Y|$. Тогава за всяка тотална функция $f : X \to Y$ съществуват $x_1 \neq x_2 \in X$ такива, че $f(x_1) = f(x_2)$

Основни комбинаторни принципи и конфигурации. Рекурентни уравнения.

Теорема (принцип на Дирихле)
Нека X е множество с n елемента (предмета), а Y е множество с m елемента (чекмеджета) и $n > m$. Както и да поставим всички предмети в чекмеджетата, поне в едно чекмедже ще има поне два предмета.

Теорема (принцип на биекцията)
Нека X и Y са крайни множества, $|X| = n, |Y| = m$. Съществува биекция $f : X \to Y \iff n = m$

Теорема (принцип на събирането)
Нека A е крайно множество, а $\mathcal{R} = \{S_1, ..., S_n\}$ е разбиване на A. Тогава $|A| = \sum_{i=1}^{k} |S_i|$

Теорема (принцип на изваждането)
Нека A е крайно множество, $A', A'' \subseteq A, A' = A\ A''$. Тогава $|A'| = |A| - |A''|$

Теорема (принцип на умножението)
Нека X и Y са крайни множества, $|X| = n, |Y| = m$. Тогава $|X \times Y| = |X|.|Y| = nm$

Теорема (принцип на делението)
Нека A е крайно множество и $B = A \times C$, където C също е крайно, $|C| \neq 0$. Тогава $|A| = |B|/|C|$

Теорема (принцип за включване и изключване)
Нека A е крайно множество и $A_1, ..., A_n \subseteq A$. Тогава:

$$|A_1| = |A| - \sum |A_i| + \sum |A_i \cap A_j| - \sum |A_i \cap A_j \cap A_k| + ... + (-1)^n |A_1 \cap A_2 \cap ... \cap A_n|$$

Комбинаторни конфигурации с наредба и повторение

Нека $0 < m \in \mathbb{Z}$ и $\mathcal{K}_{\text{н,п}}(n,m)$ е множеството, в което всеки елемент е наредена m-торка с елементи от образуващото множество A, като един елемент на A може да участва произволен брой пъти в m-торката. $|\mathcal{K}_{\text{н,п}}(n,m)| = |A^m| = n^m$

Комбинаторни конфигурации с наредба и без повторение

Нека $1 \leqslant m \leqslant n \in \mathbb{Z}$ и $\mathcal{K}_{\text{н}}(n,m)$ е множеството от наредените m-торки с елементи от образуващото множество A, в които всеки елемент участва не повече от веднъж. Нека $m \geqslant 2$. Тогава на първо място в m-торката може да се постави кой да е от елементите на A, които са n на брой, а на второ място всеки от останалите $n-1$. Съгласно принципа за умножението, за първите позиции в m-торката получаваме $n(n-1)$ възможности. За всяка от тях можем да изберем трети елемент по $(n-2)$ възможни начина и съгласно Принципа за умножението, за първите 3 позиции имаме $n(n-1)(n-2)$ възможности. Разсъждавайки индуктивно, получаваме за броя на конфигурациите с наредба и без повтаряне $\mathcal{K}_{\text{н}}(n,m) = n(n-1)...(n-m+1)$. В случай, че $m = 1$ имаме точно n възможности и тъй като $n = \mathcal{K}_{\text{н}}(n,1)$, формулата е в сила и в този случай.
Комбинаторните конфигурации от m елемента, построени от множество с n елемента, $1 \leqslant m \leqslant n$, с наредба и без повтаряне се наричат още **вариации** на n елемента от m-ти клас и се означават с V_n^m
При $m = n$, комбинаторната конфигурация се нарича **пермутация** и $|P_n| = \mathcal{K}_{\text{н}}(n,n) = n(n-1)...2.1 = n!$

Комбинаторни конфигурации без наредба и без повторение

Нека $0 > n, 0 \leqslant m \leqslant n \in \mathbb{Z}$ и $\mathcal{K}(n,m)$ е множеството от наредените m-торки, без повтаряне, от елементи на образуващото множество A с n елемента. Всъщност отсъствието на наредба и повтаряне означава, че разглежданите конфигурации са m-елементните подмножества на A. Наричаме ги **комбинации** на n елемента от m-ти клас и ги означаваме с C_n^m. За да намерим броя на комбинациите на n елемента от m-ти клас, ще използваме Принципа на делението. Ако вместо ненаредените m-торки без повтаряне разгледаме наредените, множеството $\mathcal{K}(n,m)$ ще се разшири до $\mathcal{K}'(n,m) = \mathcal{K}_{\text{н}}(n,m)$, при това всяка ненаредена m-орка ще се среща в новото множество по толкова начина, по колкото можем да подредим всичките ѝ елементи, т.е. по $m!$ начина. Следователно $|\mathcal{K}(n,m)| = \frac{\mathcal{K}_{\text{н}}(n,m)}{m!} = \frac{n(n-1)...(n-m+1)}{m!} = \frac{n!}{m!(n-m)!}$.
Така получения израз означаваме с $\binom{n}{m}$ и наричаме биномен коефициент.

Теорема (Нютон)

$$\forall n \in \mathbb{N}, \forall x, y \in \mathbb{R} : (x+y)^n = \sum_{m=0}^{n} \binom{n}{m} x^m y^{n-m}$$

Доказателство. При повдигане бинома $x+y$ на n-та степен всеки едночлен от вида $x^m y^{n-m}$ ще се получи толкова пъти, по колкото начина можем да изберем m множители измежду n-те, които взимаме x (от останалите $n-m$ избираме y), а това е точно броят на ненаредените конфигурации без повторение $\binom{n}{m}$ □

В комбинаториката едно тъждество може да бъде доказано както по формален път, така и по комбинаторен път, чрез преброяване на елементите на подходящо избрана конфигурация по два различни начина. Тази техника е известна като **принцип на двукратното броене**.

Теорема
Нека $n, m \in \mathbb{N}, n > m$. Тогава $\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1}$ и $\binom{n}{m} = \binom{n}{n-m}$

Теорема
Нека $n, m \in \mathbb{N}$. Тогава $\binom{n+m}{m} = \sum_{k=0}^{m} \binom{n+k-1}{k}$

Комбинаторни конфигурации без наредба и с повторение

Графи. Дървета. Обхождания на графи.

Дефиниция Нека $V = \{v_1, ..., v_n\}$ е крайно множество, елементите на което наричаме върхове, а $E = \{e_1, ..., e_m\}$ е крайно множество, елементите на което наричаме ребра. Функцията $f_G : E \to V \times V$ съпоставяща на всяко ребро наредена двойка от върхове, наричаме **краен ориентиран мултиграф**.

Дефиниция Нека $G(V, E, f_G)$ е краен ориентиран мултиграф и функцията f_G е еднозначна, т.е. $f(e_i) \neq f(e_j), i \neq j$. Тогава $G(V, E, f_G)$ наричаме **краен ориентиран граф**.

Дефиниция Нека $G(V, E)$ е краен ориентиран граф, такъв че релацията $E \subseteq V \times V$ е антирефлексивна и симетрична. Тогава $G(V, E)$ наричаме **краен неориентиран граф** или просто **граф**.

Дефиниция Нека $G(V, E)$ е мултиграф. Последователността от върхове $v_{i_0} ... v_{i_l}$ наричаме **път** в G от v_{i_0} до v_{i_l}, ако $(v_{i_j}, v_{i_{j+1}}) \in E$ и $v_{i_{j-1}} \neq v_{i_{j+1}}, \forall j$. Числото l наричаме **дължина на пътя**. В случая, когато $v_{i_0} = v_{i_l}$, пътя наричаме **цикъл**.

Дефиниция За всеки краен неориентиран граф $G(V, E)$ дефинираме релацията $\mathcal{P}_G \subseteq V \times V$ така: $(v_i, v_j) \in \mathcal{P}_G \iff \exists$ път в G от v_i до v_j. Релацията $\mathcal{P}_G$ е релация на еквивалентност.

Дефиниция Върховете от всеки клас на еквивалентност $V' \subseteq V$ на релацията $\mathcal{P}_G$ индуцират в $G(V, E)$ по един подграф, който наричаме **свързана компонента**. Графът $G(V, E)$ се нарича **свързан**, ако $\forall v_i, v_j \in V$, $\exists$ път в G от v_i до v_j.

Дефиниция Свързан граф без цикли наричаме **дърво**, а несвързан граф без цикли - **гора**.

Дефиниция Графът $D(\{r\}, \{\})$ с един връх r и без ребра наричаме дърво с корен r (**кореново дърво**). Единственият връх r е единствен лист на това кореново дърво.

Теорема Всяко дърво с корен е дърво.

Доказателство. Прилагаме индукция по дефиницията на дърво с корен.

- Тривиалното дърво $D(\{r\}, \{\})$ е очевидно, че е свързан граф $\implies$ тривиалното кореново дърво е дърво.
- Допускаме, че дървото $D(V, E)$, с корен r е свързан граф без цикли.
- Ще докажем, че дървото $D'(V', E') = D'(V \cup \{w\}, E \cup \{(u, w)\})$ с корен r е свързан граф.
  За произволни два върха v_i и v_j от $V \exists$ път от v_i до v_j в D, защото според ИП D е свързан граф $\implies$ $\exists$ такъв път и в D'. Път от произволен връх $v_i \in V$ до w в D' получаваме, като към съществуващия в D път от v_i до u добавим реброто (u, w), с което w е u, защото $w \notin V \implies \exists$ път в D' между всеки два върха и D' е свързан граф.
  Съгласно ИП в D няма цикли, а за да има цикъл в D', той трябва да съдържа реброто (u, w). Това не е възможно, тъй като w е край само на реброто $(u, w), |w| = 1$ и съгласно дефиницията не може да участва в цикъл.

$\square$

Теорема Нека $D(V, E)$ е дърво с корен r. Тогава $|V| = |E| + 1$.

Доказателство.

- За тривиалното кореново дърво $D(V = r, E = \{\})$ имаме $|V| = 1, |E| = 0 \implies |V| = |E| + 1$
- Нека $D(V, E)$ е кореново дърво и $|V| = |E| + 1$
- Присъединяваме w към върха $u \in V$ и получаваме кореновото дърво $D'(V', E'), V' = V \cup w, E' = E \cup (u, w)$. Сега $|V'| = |V| + 1, |E'| = |E| + 1 \implies |V'| - |E'| = |V| - |E| = 1$ и $|V'| = |E'| + 1$

$\square$

Дефиниция Нека $G(V, E)$ е граф, а $D(V, E'), E' \subseteq E$ е дърво. Тогава D наричаме **покриващо дърво** на G.

Алгоритъм (обхождане в ширина)

Дадено: Свързан граф $G(V, E)$
Резултат: Покриващо дърво $D(V, E')$ на G.
Процедура:

1. Коренът r на покриващото дърво ще изберем за начален връх на обхождането. Затова $L_0 = \{r\}$. Образуваме дървото $D_0(V_0, E_0), V_0 = L_0, E_0 = \varnothing$ и $l = 0$.

2. Ако $V_l = V$, тогава алгоритъмът спира и търсеното покриващо дърво е D_l. В противен случай преминаваме към 3.

3. Нека $D_l(V_l, E_l)$ е дървото, построено след l-тата стъпка и L_l са върховете от l-то ниво. Образуваме следващото ниво $L_{l+1} = \{v | v \notin V_l, \exists w \in L_l, (w, v) \in E\}$. Образуваме дървото $D_{l+1}(V_{l+1}, E_{l+1})$, $V_{l+1} = V_l \cup L_{l+1}$, $E_{l+1} = E_l \cup \{(w, v) | (w, v) \in E, w \in L_l, v \in L_{l+1}\}$ с присъединяване на всеки $v \in L_{l+1}$ към съответния $w \in L_l$. Нека $l = l + 1$ и преминаваме към 2.

Алгоритъм (обхождане в дълбочина)

Дадено: Свързан граф $G(V, E)$
Резултат: Покриващо дърво $D(V, E')$ на G.
Процедура:

1. Коренът r на покриващото дърво ще изберем за начален връх на обхождането. Затова $L_0 = \{r\}$. Образуваме дървото $D_0(V_0, E_0), V_0 = \{r\}, E_0 = \varnothing$. Нека $t = r, i = 0$ и $p(t)$ е неопределен.

2. Ако $V_l = V$, тогава алгоритъмът спира и търсеното покриващо дърво е D_l. В противен случай преминаваме към 3.

3. Нека е построено дървото $D_i(V_i, E_i)$. Търсим $v \in V_i$ такъв, че $(t, v) \in E$ и

   - ако има такъв, построяваме дървото $D_{i+1}(V_{i+1}, E_{i+1}), V_{i+1} = V_i \cup \{v\}, E_{i+1} = E_i \cup \{(t, v)\}$ Сега $p(v) = t, t = v, i = i+1$ и преминаваме към 2.

   - в противен случай:

     – ако $t \neq r$, тогава $t = p(t)$ и преминаваме към 2.

     – иначе - край. Търсеното покриващо дърво е D_i.

Дефиниция **Ойлеров път** в свързания мултиграф $G(V, E)$ наричаме път, който минава еднократно през всяко ребро на мултиграфа. Ако Ойлеровият път има съвпадащи начало и край, тогава той се нарича **Ойлеров цикъл**.

Теорема
Свързаният мултиграф $G(V, E)$ е Ойлеров $\iff$ всеки връх на G е с четна степен

Теорема
Свързаният мултиграф $G(V, E)$ съдържа Ойлеров път, който не е Ойлеров цикъл $\iff$ има точно 2 върха с нечетна степен

Доказателство. $\Leftarrow$) Нека v_i и v_j са върховете с нечетна степен. Добавяме в мултиграфа ребро $e \notin E$ и додефинираме $f_{G'}(e) = (v_i, v_j)$ Полученият мултиграф G' е Ойлеров $\implies$ можем да построим Ойлеров цикъл. Премахваме добавеното ребро и получаваме път от v_i до v_j, който съдържа всички ребра на графа $G(V, E)$ точно по веднъж $\implies$ е Ойлеров път.

$\Rightarrow$) Нека ребрата на $G(V, E)$ образуват Ойлеров път от v_i до v_j. Добавяме ребро $e \notin E$ и додефинираме $f_{G'}(e) = (v_i, v_j)$. Пътят се превръща в Ойлеров цикъл за новополучения мултиграф G' $\implies$ всички върхове са с четна степен. Добавянето на реброто (v_i, v_j) е увеличило с 1 само степените на v_i и v_j $\implies$ в G всички върхове са с четна степен, с изключение на тези два върха. $\square$

Езици, автомати и изчислимост

Крайни автомати. Регулярни езици.

Дефиниция **Краен детерминиран автомат** наричаме петорката $A =< Q, \Sigma, q_0, \delta, F >$, където Q е крайно множество от състояния, Σ е крайна входна азбука, $q_0 \in Q$ е начално състояние, $\delta : Q \times \Sigma \to Q$ е частична функция на преходите, пресмятаща следващото състояние, $F \subseteq Q$ са финални състояния

Дефиниция **Краен недетерминиран автомат** наричаме петорката $A =< Q, \Sigma, q_0, \delta, F >$, където Q е крайно множество от състояния, Σ е крайна входна азбука, $q_0 \in Q$ е начално състояние, $\delta : Q \times \Sigma \to 2^Q$ е частична функция на преходите, пресмятаща следващото състояние, $F \subseteq Q$ са финални състояния

Теорема
За всеки КНА A съществува КДА A' такъв, че $L_A = L_{A'}$

Доказателство. Нека $A =< Q, \Sigma, q_0, \delta, F >$. Построяваме КДА $A' =< Q', \Sigma, t_0, \delta', F' >$, където $Q' \subseteq 2^Q$. Нека множеството $\{q_{p_1}, q_{p_2}, ..., q_{p_l}\} \in Q'$ означим с $t_{[p_1,p_2,...,p_l]}$, където $t_0 = \{q_0\} = t_{[0]}$. Нека $F' = \{t_{[p_1,...p_l]} | q_{p_1}, ..., q_{p_l} \cap F \neq \varnothing\}$ $\square$

Дефиниция Нека $\widetilde{\Sigma} = \Sigma \cup \{\varepsilon, \varnothing, *, +, (,)\}$. Дефинираме **регулярни изрази** рекурсивно:

- Символите $\varnothing$, ε, както и всяка буква $a \in \Sigma$ са регулярни изрази
- Ако r_1 и r_2 са регулярни изрази, то думите $(r_1.r_2)$, $(r_1 + r_2)$ и $(r_1)^*$ са регулярни изрази

Дефиниция Казваме, че езикът L е **регулярен**, ако $L = L(r)$, за някой регулярен израз r над Σ, където езикът $L(r)$ се дефинира така:

- $L(\varnothing) = \varnothing, L(a) = \{a\}$ за $a \in \Sigma$
- $L(r_1.r_2) = L(r_1).L(r_2)$, $L(r_1 + r_2) = L(r_1) \cup L(r_2)$ и $L(r_1^*) = L(r_1)^*$

Теорема (Клини)
Всеки език, разпознаван от краен автомат, е регулярен.

Доказателство. Нека $L \subseteq \Sigma^*$ е автоматен език и $\mathcal{A} =< Q, \Sigma, q_0, \delta, F >$ е краен автомат такъв, че $L(\mathcal{A}) = L$. БОО $Q = \{0, 1, ..., n-1\}$ и $q_0 = 0$. Нека $\forall 0 \leqslant i, j \leqslant n-1$ и $\forall 0 \leqslant k \leqslant n$ с R_{ij}^k да означим множеството от всички думи над Σ, чрез които можем да се придвижим от състоянието i до състоянието j без да преминаваме през състояние $s \geqslant k$, т.е.

$$R_{ij}^k = \{w \in \Sigma^* | i \underset{\mathcal{A}}{\overset{w}{\longmapsto}} j, \text{ минавайки само през състояния } s < k\}$$

Тогава $R_{ii}^0 = \{\varepsilon\} \cup \{a | (i, a, i \in \delta)\}$ и $R_{ij}^0 = \{a | (i, a, j) \in \delta\}$ за $i \neq j \implies \forall 0 \leqslant i, j \leqslant n-1$ езикът R_{ij}^0 е регулярен. От друга страна, $\forall 0 \leqslant i, j \leqslant n-1$ и $\forall 0 \leqslant k \leqslant n-1$ е в сила $R_{ij}^{k+1} = R_{ij}^k \cup R_{ik}^k (R_{kk}^k)^* R_{kj}^k$. Оттук езиците R_{ij}^k са регулярни $\forall 0 \leqslant i, j \leqslant n-1$ и $\forall 0 \leqslant k \leqslant n$. От друга страна $L(\mathcal{A}) = \bigcup_{i \in F} R_{0i}^n \implies L(\mathcal{A})$ е регулярен. $\square$

Лема (разрастване на РЕ (uvw))
Нека $L \subseteq \Sigma^*$ е регулярен език. Тогава съществува $n \in \mathbb{N}, n \geqslant 1$ такова, че за всяка дума $w \in L, |w| > n$, съществуват $x, y, z \in \Sigma^*$ такива, че $w = xyz$ и

- $|xy| \leqslant n$
- $|y| \geqslant 1$
- $xy^i z \in L, \forall i$

Доказателство. Нека $\mathcal{A} =< Q, q_0, \delta, F >$ е краен автомат над Σ такъв, че $L(\mathcal{A}) = L$. Да положим $n = |Q|$. Нека сега $w \in L$ с $|w| = m > n$ и нека $w = a_1 ... a_m$, където $a_1 ... a_m \in \Sigma$. Тогава съществува път $q_0, a_1, q_1, ..., a_m, q_m$ в $G_{\mathcal{A}}$ с $q_m \in F$. Тъй като $m > n$, то поне две от състоянията в горната редица съвпадат (принцип на Дирихле). Нека s е най-голямото, за което

$q_i \neq q_j, \forall 0 \leqslant i < j \leqslant s$. Ясно е, че $s < n$ и освен това $q_k = q_s$ за някое $k < s$. Полагаме $x = a_1 ... a_k, y = a_{k+1} ... a_s, z = a_{s+1} ... a_m$. Тогава $|xy| = s \geqslant n$ и $|y| = s - k \geqslant 1$. При това $q_0 \overset{x}{\underset{\mathcal{A}}{\longmapsto}} q_k, q_k \overset{y}{\underset{\mathcal{A}}{\longmapsto}} q_k$ и $q_k \overset{y}{\underset{\mathcal{A}}{\longmapsto}} q_m$, откъдето $\forall i \geqslant 0, xy^i z \in L$ □

Примери за нерегулярни езици

1. $\{a^n b^n \mid n \geqslant 0\}$
2. $\{w \mid w = w^{Rev}\}$
3. $\{ww \mid w \in \Sigma^*\}$
4. $\{a^n b^n c^n \mid n \geqslant 0\}$

Теорема (Майхил-Нероуд)
Нека $L \subseteq X*$. Релацията $R_L \subseteq X * \times X*$ има краен индекс $\iff$ L е автоматен

Доказателство. □

Контекстносвободни граматики и езици. Стекови автомати.

Дефиниция **Контекстносвободна граматика** наричаме четворката $G = (\Gamma, \Sigma, R, S)$, където $\Sigma \subseteq \Gamma$ са крайни азбуки, $S \in \Gamma \backslash \Sigma$ и $R \subseteq (\Gamma \backslash \Sigma) \times \Gamma^*$ е крайно множество.

Дефиниция

Дефиниция **Недетерминиран стеков автомат** наричаме седморката $A =< Q, \Sigma, Z, q_0, z_0, \delta, F >$, където Q е крайно множество от състояния, Σ е крайна входна азбука, Z е крайна стекова азбука, $q_0 \in Q$ е начално състояние, $z_0 \in Z$ е начална стекова буква, $\delta : Q \times (X \cup \{\varepsilon\}) \times Z \to 2^{Q \times Z^*}$ е частична функция на преходите, пресмятаща следващото състояние, $F \subseteq Q$ са финални състояния

Лема (за разрастване на КСЕ)
Нека L е безконтекстен език. Тогава съществува число $n > 0$ такова, че за всяка дума $w \in L, |w| > n$, съществуват думи x, y, z, u, v такива, че $w = xyzuv$ и

- $|yzu| \leqslant n, |yu| \geqslant 1$
- $xy^i uv^i w \in L, \forall i \geqslant 0$

Доказателство. Нека G безконтекстна граматика в НФЧ, която генерира точно думите от L с дължина поне 2. ... □

Примери за неконтекстносвободни езици

1. $\{a^n b^n c^n \mid n \geqslant 0\}$
2. $\{ww^{Rev} \mid w \in \{a,b\}^*\}$
3. $\{ww \mid w \in \{a,b\}^*\}$
4. $\{a^n b^m c^m d^n \mid n, m \geqslant 0\}$

Дизайн и анализ на алгоритми

Сложност на алгоритъм. Асимптотично поведение на целочислени функции (O-, Ω-, Θ-, o- и ω-нотация). Сложност на рекурсивни програми.

Дефиниция **Машина на Тюринг** (с безкрайна в едната посока лента) наричаме петорката $A =< Q, \Gamma, q_0, \delta, F >$, където Q е крайно множество от състояния, Γ е крайна азбука, $q_0 \in Q$ е начално състояние, $\delta : Q \times \Gamma \rightarrow (Q \cup F) \times \Gamma \times \{L, R, S\}$ е функция на преходите, F са финални състояния, $F \cap Q = \varnothing$

Дефиниция Нека $f_M : A^* \rightarrow A^*$ е изчислима с машината на Тюринг M тотална функция. Функцията

$$t_M(n) = \max_{\alpha \in A^*, d(\alpha)=n} [\text{ брой стъпки на M при работа върху } \alpha \text{ }]$$

наричаме **сложност по време** на M в най-лошия случай, а функцията

$$\tilde{t}_M(n) = \frac{\sum_{\alpha \in A^*, d(\alpha)=n} [\text{ брой стъпки на M при работа върху } \alpha \text{ }]}{|A^n|}$$

наричаме средна **сложност по време** на M.

Дефиниция Нека $f_M : A^* \rightarrow A^*$ е изчислима с машината на Тюринг M тотална функция. Функцията

$$s_M(n) = \max_{\alpha \in A^*, d(\alpha)=n} [\text{ брой използвани клетки от M при работа върху } \alpha \text{ }]$$

наричаме **сложност по памет** на M в най-лошия случай, а функцията

$$\tilde{s}_M(n) = \frac{\sum_{\alpha \in A^*, d(\alpha)=n} [\text{ брой използвани клетки от M при работа върху } \alpha \text{ }]}{|A^n|}$$

наричаме средна **сложност по памет** на M.

Нека f е реалнозначна функция, дефинирана в $\mathbb{R}$ или $\mathbb{Z}^+$. Функцията f се нарича асимптотично отрицателна, когато $\exists n_0 \in \mathbb{N} \forall n \geqslant n_0 : f(n) \geqslant 0$

Функцията се нарича асимптотично положителна, когато $\exists n_0 \in \mathbb{N} \forall n > n_0 :$ $f(n) > 0$

Дефиниция $\mathbf{O(g)}$ е множеството от всички функции, които растат асимптотично не по-бързо от g

$$O(g) = \{f \mid \exists c > 0 \;\; \exists n_0 \in \mathbb{N} \;\; \forall n \geqslant n_0 : 0 \leqslant f(n) \leqslant c.g(n)\}$$

Означение: $f \in O(g), f = O(g), f \preceq g$

Дефиниция $\mathbf{\Omega(g)}$ е множеството от всички функции, които растат асимптотично не по-бавно от g

$$\Omega(g) = \{f \mid \exists c > 0 \;\; \exists n_0 \in \mathbb{N} \;\; \forall n \geqslant n_0 : 0 \leqslant c.g(n) \leqslant f(n)\}$$

Означение: $f \in \Omega(g), f = \Omega(g), f \succeq g$

Дефиниция $\mathbf{o(g)}$ е множеството от всички функции, които растат асимптотично по-бавно от g

$$o(g) = \{f \mid \forall c > 0 \;\; \exists n_0 \in \mathbb{N} \;\; \forall n \geqslant n_0 : 0 \leqslant f(n) < c.g(n)\}$$

Означение: $f \in o(g), f = o(g), f \prec g$

Дефиниция $\omega(\mathbf{g})$ е множеството от всички функции, които растат асимптотично по-бързо от g

$$\omega(g) = \{f \mid \forall c > 0 \;\; \exists n_0 \in \mathbb{N} \;\; \forall n \geqslant n_0 : 0 \leqslant c.g(n) < f(n)\}$$

Означение: $f \in \omega(g), f = \omega(g), f \succ g$

Дефиниция $\mathbf{\Theta(g)}$ е множеството от всички функции, които растат асимптотично еднакво с g

$$\Theta(g) = \{f \mid \exists c_1 > 0 \;\; \exists c_2 > 0 \;\; \exists n_0 \in \mathbb{N} \;\; \forall n \geqslant n_0 : 0 \leqslant c_1.g(n) \leqslant f(n) \leqslant c_2.g(n)\}$$

Означение: $f \in \Theta(g), f = \Theta(g), f \asymp g$

Свойства

1. Ако $f \sigma g$ и $g \sigma h$, то $f \sigma h, \;\; \sigma \in \{\prec, \succ, \preceq, \succeq, \asymp\}$ - транзитивност
2. $f \sigma f, \;\; \forall \sigma \in \{\preceq, \succeq, \asymp\}$ (Θ, O и Ω са рефлексивни)
3. $f \succeq g$ и $g \succeq f \iff f \asymp g$ (O и Ω са антисиметрични)
4. $f \asymp g \implies g \asymp f$ (Θ е симетрична)
5. $f \preceq g \iff g \succeq f, f \prec g \iff g \succ f$

Теореми **(гранични)**

$$max\{f,g\} \asymp f+g$$

$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \iff f = o(g)$ $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty \iff f = \omega(g)$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = c > 0 \implies f = \Theta(g)$$

Теорема **(Мастър)**
Нека $a \geqslant 1, b > 1$ и $f(n)$ и $T(n)$ са асимптотично положителни функции и $T(n) = a.T(\frac{n}{b}) + f(n)$. Нека $k = \log_a b$. Тогава:

1. Ако $f(n) = O(n^{k-\varepsilon})$ за някое $\varepsilon > 0$, то $T(n) = O(n^k)$
2. Ако $f(n) = O(n^k)$, то $T(n) = \Theta(n^k logn)$
3. Ако $f(n) = \Omega(n^{k+\varepsilon})$, за някое $\varepsilon > 0$ и $\exists c \in (0,1)\ \exists n_0 \in \mathbb{N}\ \forall n \geqslant n_0 :$ $a.f(\frac{n}{b}) \leqslant c.f(n)$, то $T(n) = \Theta(f(n))$

Алгоритми в графи с тегла на ребрата. Оценки за сложност.

Дефиниция Нека $G(V,E)$ е свързан граф и $c : E \to R$ е функция с реални стойности, дефинирана по ребрата на G. Стойността $c(e), e \in E$ наричаме **тегло** (цена) на реброто e. Нека $D(V,E')$ е покриващо дърво на $G(V,E)$. **Тегло** (цена) на дървото наричаме сумата $c(D) = \Sigma_{e\in E'} c(e)$. Покриващото дърво $D_0(V,E_0)$ на $G(V,E)$ наричаме **минимално**, ако $c(D_0) \leqslant c(D)$ за всяко друго покриващо дърво D на G.

Теорема **(МПД-свойство)**
Нека $G(V,E)$ е свързан граф с теглова функция на ребрата $c : E \to R$ и $\varnothing \neq U \subset V$. Нека $e = (v_i, v_j) \in E$ е такова, че $v_i \in U, v_j \in V\backslash U$ и (v_i, v_j) има минимално тегло измежду всички такива ребра. Тогава съществува МПД $D(V,E')$ на G такова, че $e \in E'$

Алгоритъм **(Прим)**

Дадено: Свързан граф $G(V,E)$ и функция $c : E \to R$, задаваща тегла на ребрата му.
Резултат: Минимално покриващо дърво $D(V',E)$ с корен - зададен връх $r \in V$ на G.
Процедура:

1. Построяваме дървото $D_0(V_0,E_0), V_0 = \{r\}, E_0 = \varnothing, k = 0$.

2. Нека сме построили $D_k(V_k, E_k)$. Търсим реброто $e = (v_i, v_j), v_i \in V_k, v_j \in V \backslash V_k$ с минимално тегло и построяваме $D_{k+1}(V_{k+1}, E_{k+1}), V_{k+1} = V_k \cup \{v_j\}, E_{k+1} = E_k \cup \{\varepsilon\}, k = k + 1$

3. Ако $V_k = V$ - край, полученото дърво $D(V, E'), E' = E_k$ е оптималното. Иначе минаваме към 2.

Оценка на сложността: $T = O(V + E), S = O(V + E)$

Алгоритъм (Крускал)

Дадено: Свързан граф $G(V, E)$ и функция $c : E \to R$, задаваща тегла на ребрата му.
Резултат: Минимално покриващо дърво $D(V', E)$ на G.
Процедура:

1. Сортираме ребрата на G в нарастващ ред на цената и нека този ред е $e_1, ..., e_m$

2. От всеки връх v на графа образуваме тривиално дърво $D_v(\{v\}, \varnothing)$

3. За всяко ребро $e_i = (v_{i_1}, v_{i_2}), i \in I_m$ (по реда определен от сортирането), правим следното: ако v_{i_1}, v_{i_2} са в различни дървета, съответно $D'(V', E')$ и $D''(V'', E'')$, съединяваме двете в дървото $D(V' \cup V'', E' \cup E'' \cup \{(v_{i_1}, v_{i_2})\})$

Оценка на сложността: $T = O(V + E), S = O(V + E)$

Алгоритъм (Дейкстра)

Дадено: Свързан граф $G(V, E)$ и функция $c : E \to R^+$ и начален връх $v_0 \in V$.
Резултат: Дърво на минималните пътища от v_0 до всички останали върхове в G.
Процедура:

1. Разширяваме $c : E \to R^+$ до $c* : V \times V \to R^+$.

2. Нека $dist[0] = 0, part[0] = -1$ и $U = \{0\}$, а $dist[i] = c * (0, i)$ и $part[i] = 0, i \in I_n$

3. Повтаряме $n - 1$ пъти стъпките:

   - Избираме връх $j \notin U$, за който $dist[j]$ е минимално и $U = U \cup \{j\}$

   - За всеки $k \notin U$ пресмятаме $dist[k] = min(dist[k], dist[j] + c*(j, k))$. Ако min е $dist[j] + c * (j, k)$, тогава $part[k] = j$

Оценка на сложността: $T = O(V + E), S = O(V + E)$

Алгоритъм (Флойд)

Динамично програмиране. Оценки за сложност.

Динамичното програмиране е метод за решаване на задачи чрез комбиниране на решенията на подзадачи. Алгоритъмът решава всяка подзадача веднъж и запазва резултата в таблица, избягвайки нуждата от преизчисляване на отговора всеки път, когато решава някоя подзадача.

Задачи с линейна таблица на подзадачите (най-дълга растяща подредица)

Задачи с триъгълна таблица на подзадачите (оптимално разбиване на подредица)

Задачи с правоъгълна таблица на подзадачите (най-дълга обща подредица на две редици, задача за раницата)