

## Съотност на алгоритми за крайни автомати

Сега ще направим оценка на съотността на алгоритми за крайни автомати и регулрни езици. Да отбележим, че всички доказателства, които сме разгледали, свидетелстват и за небивко алгоритмична процедурата.

В началото ще отбележим нещо, което е необходимо при всеко такова разглеждане. Когато разглеждаме алгоритмите за рефлексивно и транзитивно затваряне на рефлексивна релация, както и последващите алгоритми за затваряне на множество относно реалния и релации, трябва да си обезвреждаме, че основното множество е крайно с и на друг елемент. Това и участвува в оценката на съотността на алгоритмите.

Когато говорим за крайни автомати (KDA или НКА) отпътуваме към този че бъде дробът на елементите на  $K\cup\Sigma$  и бъде същността от това да ли е KDA или НКА че разглеждане и дробът  $|A|$  на реалната за преходите че броят на ел. на  $\Sigma$ . Всички обаче могат да ограничат броя на елементите на  $A$  отгоре с броя на  $K\cup\Sigma$ , защото  $|A| \leq |K|^{|K|}|\Sigma|$ . Когато говорим за регулрни езици, онежеят от регулрни изрази, отпътуваме към на регулрните изрази че бъде неговата граница  $|A|$ , разглеждана като дума в азбука  $\Sigma = \{\phi, \circ, U, (, )\}$ . Сега сме готови да покажем следната

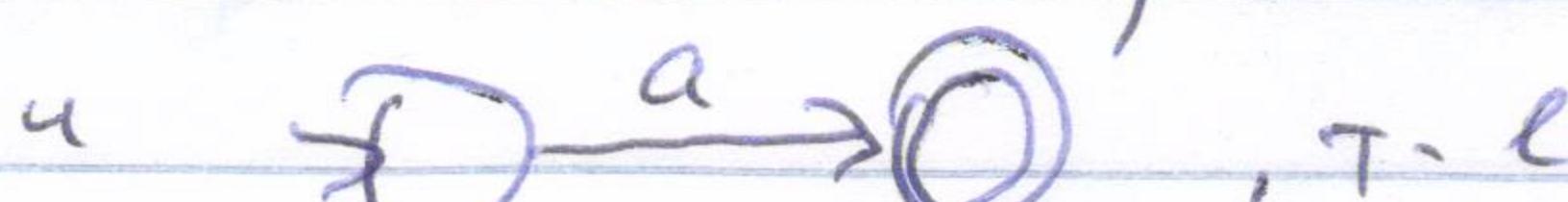
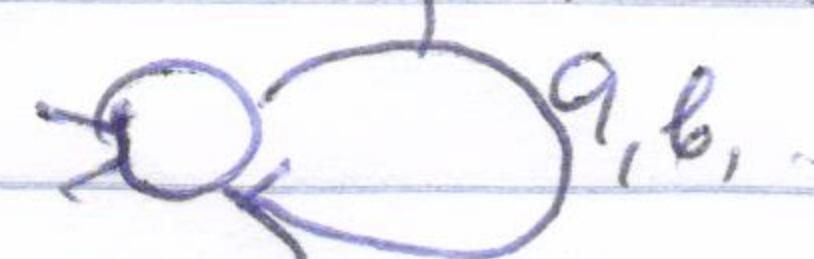
- Теорема. а) Съществува едноненадален алгоритъм, който на гаден НКА, дава еквивалентен KDA;
- б) Съществува покритиялен алгоритъм, който на гаден регулрни изрази дава НКА  $M$ , такъв, че  $L(M) = L(A)$ ;
- в) Съществува едноненадален алгоритъм, който на гаден НКА  $M$ , дава регулрни изрази, такива, че  $L(M) = L(A)$ ;
- г) Съществува покритиялен алгоритъм, който на гаден KDA дава минимален KDA, еквивалентен на изризи;
- д) Съществува покритиялен алг., който на гадени KDA дава отговор дали са еквивалентни или не;

e) Съществува едн. алг., който по задади гба НКА  
габа отговор да са съвместни и не;  
Погодно и за пер. изрази.

2-бо. а) Първият алгоритъм за крайни автомати, който разглеждае в курса бие за детерминизација на НКА. КДА  $M' = \langle K', \Sigma, \delta', S', F' \rangle$ , който построихме на базата на съврка. Първо  $K' = 2^K$  - и то не събогото от поганото съврка на  $K$ . Тук достъп на елементарните операции, който направихме за построване на всички крайни идентифицира, е  $2^{IK'}$ . За построването на  $F'$  използвахме също така идентифициране на  $\Delta'$  и също тук също съвсем идентифициране на  $\Delta$ . Сега това за  $E(p)$ -та използвахме за всичко  $\Delta$  и  $p$  относно  $R = \{ (p_1, p_2) \mid (p_1, \epsilon, p_2) \in \Delta \} \subseteq K^2$ . Тук същинската  $O(1K^3)$ . Сега това определихме  $\delta'(Q, q) = \cup \{ E(p) \mid \text{същ. } q \in Q, p \in K : (q, q, p) \in \Delta \}$  за фикс.  $Q \in K'$ ,  $q \in \Sigma$ . Същинската тук е  $O((1K^2) |\Delta| \cdot 1K^3)$ . Очевидно, сложността на алгоритъма е  $O(2^{IK'} \cdot |K|^3 \cdot |K|^2 \cdot |\Delta| \cdot |K|^3 \cdot |\Sigma|)$ . Така можем да видим, че алгоритъмът е експоненциален израз  $2^{IK'}$ .

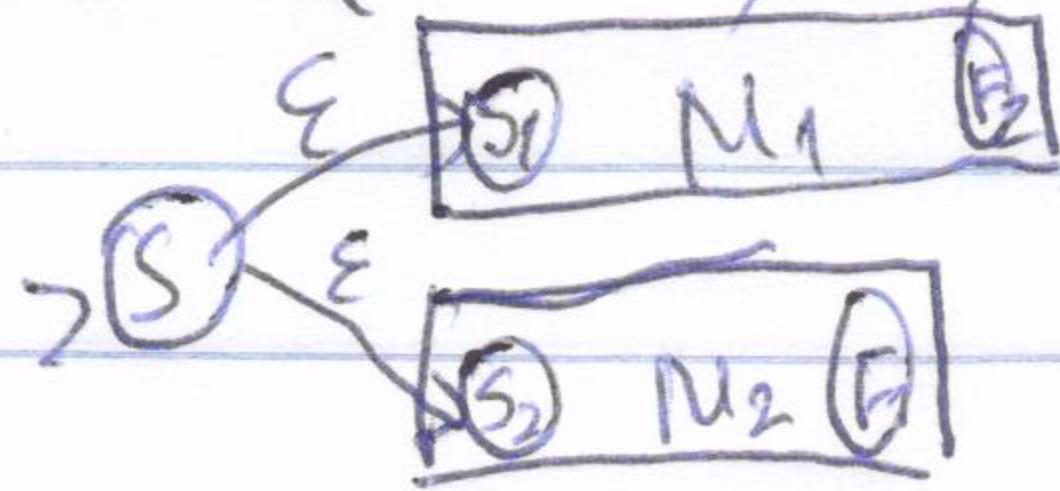
б) Първият доказателство, че за всички пер. израз  $\lambda$  (с идентични обозначения) има НКА  $M$ , такава че  $L(\lambda) = L(M)$ , заеднстване, че сложността на алг. е  $O(4|\lambda|^2)$ . С идентични обозначения.

1)  $|\lambda| = 1$ , т.е.  $\lambda = \phi$  или  $\lambda = a$ ,  $a \in \Sigma$  имахме



Съчинението на  $\lambda$  не нарушава  $\lambda = 2 \cdot 1$ , а провежда на  $\lambda$  на  $|\Sigma| \cdot \lambda$ , т.е.  $O(4|\lambda|^2) = O(|\lambda|^2)$

2)  $\lambda = (Q, U\lambda_2)$ ,  $|\lambda| = |\lambda_1| + |\lambda_2| + 1 + 2$



$$M_1: L(M_1) = \lambda[\lambda_1]$$

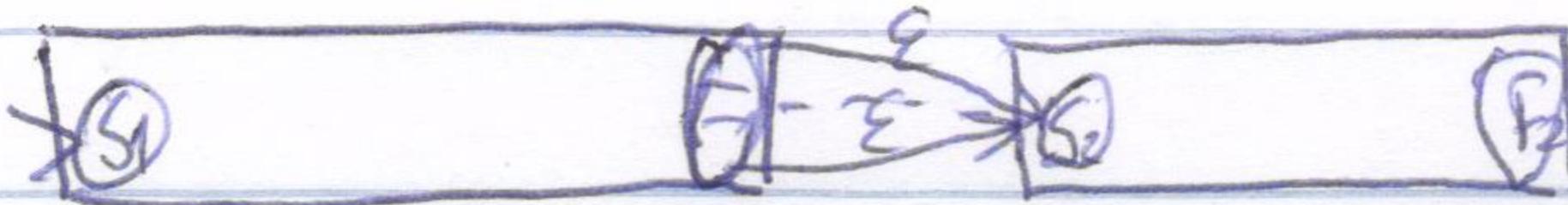
$$M_2: L(M_2) = \lambda[\lambda_2]$$

Сложността на  $\lambda$  не нарушава  $O(4|\lambda_1|^2)$

$$4(\lambda)^2 = 4(|\lambda_1| + |\lambda_2| + 3)^2 \geq 4|\lambda_1|^2 + 4|\lambda_2|^2 + O(4|\lambda|^2) = O(|\lambda|^2)$$

3)  $\lambda = (\lambda_1, \lambda_2)$   $(\lambda) = (\lambda_1) + (\lambda_2) + 3$

$$M_1: L(M_1) = \lambda[\lambda_1], M_2: L(M_2) = \lambda[\lambda_2]$$



Рак је  $\Delta$  и  $\Delta_1 + \Delta_2 + K_1$  и так са. е  $O(|\Delta|^2)$

4)  $\Delta = \Delta_1^*$  – Алгоритм

$$M_1: L(M_1) = L[\Delta_1]: \quad | \Delta | \leq |\Delta_1| + |K_1| + 1$$

Оттога е лесно, че  $4|\Delta|^2 \geq 4|\Delta_1|^2 + |K_1| + 1$  и

Сложността е  $O(|\Delta|^2)$ .

6) За доказателството на теоремата, че за всички НКА  $M = (K, \Sigma, \Delta, S, F)$  има рекурентен израз  $L$ , такъв, че  $L(M) = L[\Delta]$  използвахме  $R(i, j, k)$  за  $i, j = 1, \dots, n = |K|$ ,  $k = 0, 1, \dots, n$ . От рекурентната формула

$$R(i, j, k+1) = R(i, j, k-1) \cup R(i, k, k-1) R(k, k, k-1)^* R(k, j, k-1)$$

може да видим, че на всяка стъпка  $R(i, j, k)$  се употребява, а освен това  $R(i, j, 0) \leq 3(|\Sigma|+1)$ . Следователно,

$$|L(M)| = |\{R(i, j, n) \mid q_j \in F\}| \leq |K| \cdot 3(|\Sigma|+1) \cdot 3^{n-1}, \text{ т.е.}$$

Сложността е  $O(|K| \cdot (|\Sigma|+1) \cdot 3^{n-1})$ . Т.е. алгоритъмът

е итеративен. Да означим здадено че може да се итерира

$$R(i, j, 0) \leq \sum_{q \in F} \dots$$

и следователно (затворено)

п. изразът за  $R(i, j, 0)$  в Наш-Ломан изразът ще бъде

$$((\dots(q_1 q_2) q_3) \dots q_n) \phi.$$

2) За инициализация на КДА има използвахме

същия алгоритъм:

Започнем с инициализация на съв.  $K \neq F \wedge F \neq \Xi_0$ ;

repeat for  $n := 1, 2, \dots$

Нашите инициализации на съв.  $H \in \Xi_n$  от тези на

иначеве  $H \in \Xi_{n-1}$

until  $\Xi_n$  и  $\Xi_{n-1}$  събият.

Тук използваме лемата за едината думба и сложността на алгоритъма е  $O(|\Sigma| \cdot |K|^3)$ , която е може да се покаже че е сложността.

g) Тук използваме идейната мотивация. Нека са дадени две КДА. Първо ги инициализираме и

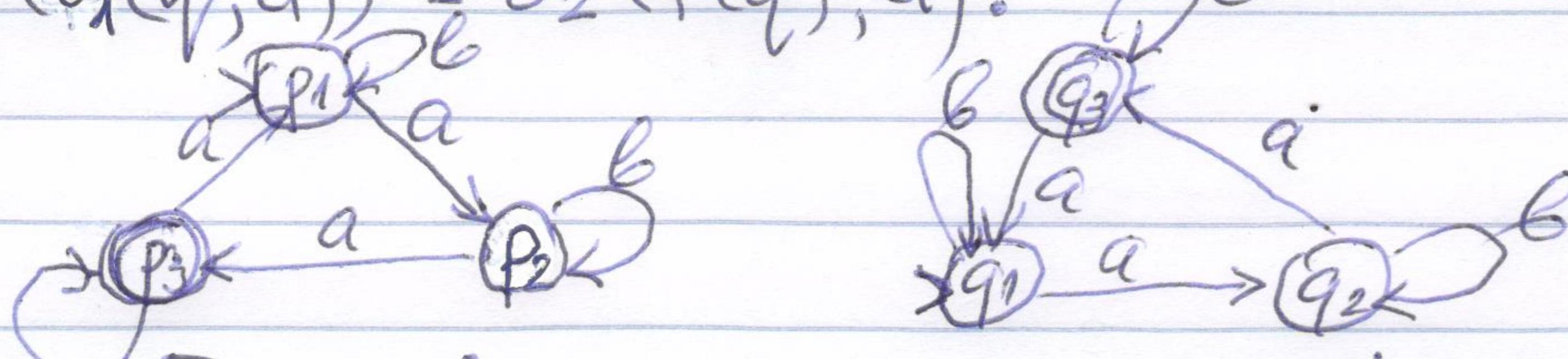
същият съв. ще бъде зам. са изоморфни.

Какъв ще бъде този изоморфизъм  $M_1$  и  $M_2$  ще бъде

изоморфни?

2. Казваме, че гба автомати  $M_1 = (K_1, \Sigma, \delta_1, S_1, F_1)$  и  $M_2 = (K_2, \Sigma, \delta_2, S_2, F_2)$  са изоморфни (нераразличими) ако същ. функция  $f: K_1 \rightarrow K_2$ , такава, че  $f(S_1) = S_2$  и  $f[F_1] = F_2$ , и  $f(\delta_1(q, a)) = \delta_2(f(q), a)$ .

Пример. Нека



са гба автомати. Тогава те са изоморфни защото функцията  $f: K_1 \rightarrow K_2$ , такава, че  $f(p_i) = q_i$ ,  $i = 1, 2, 3$  е функция с исклучителни свойства.

e) За тази част използваме предишните. Нека са гадети гваника. Първо определям изображение на гбата автомата (това е експоненциална сложност), след това ги линтилизираме (помага на линейна сложност) след това проверяваме дали са изоморфни (линейно). Основателно, сложността е линейна.

Означавамо, а) и б) имат експоненциална сложност, то

за останалите могат да се запазят интересни въпроси:

1) Дали има линийски алгоритъм, който дава отговор на въпроса дали гба НКА са еквивалентни;

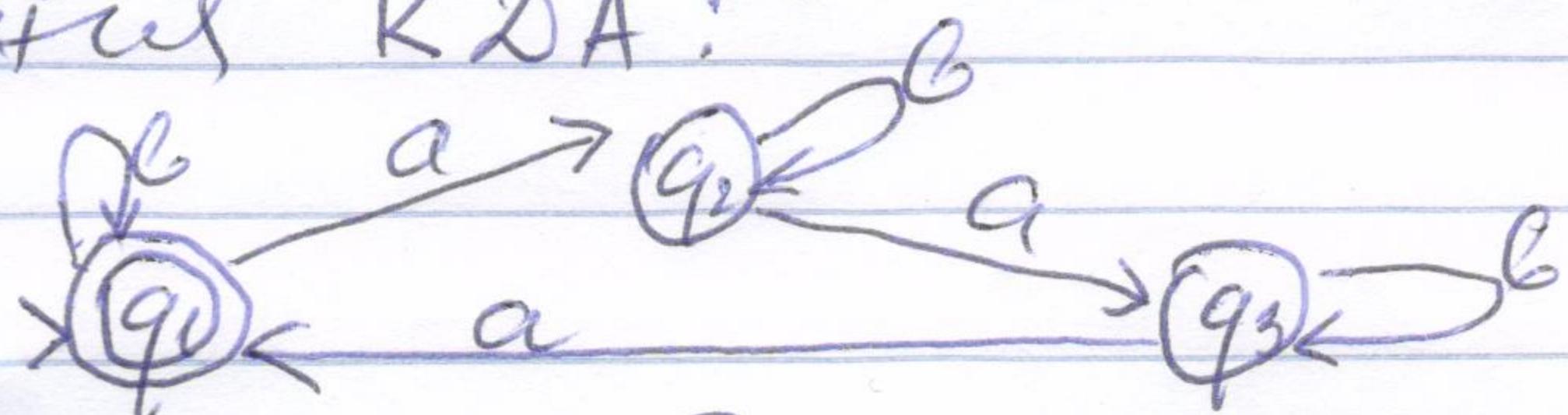
2) Дали има линийски алгоритъм, който да гаде НКА, дава НКА с най-малко съставки;

3) Дали не-линейният, дали има линийски алгоритъм, който да гаде НКА дава КДА еквивалентен на гадето с най-малко съставки. Значи експоненциален алгоритъм, който гади във видуващия алгоритъм, който гади във видуващия алгоритъм.

Уже видяхме как Маню, че този въпрос не е линеен от съмисъл, когато разглеждаме темата за крайни автомати като алгоритъм

## Крайни автомати като алгоритми

Едно от забележителните неща на KDA е, че те могат да се разглеждат като ефективни алгоритми б/g гените от гадена азбука. Да разгледаме сървър KDA:



Три може да съдже разглеждането като алгоритм като сървър:

$q_1$ :  $\tau := \text{get-next-symbol};$   
if  $\tau = \text{end-of-file}$  then  $\text{пълни гума};$   
else if  $\tau = a$  then go to  $q_2$ ;  
else if  $\tau = b$  then go to  $q_3$ ;

$q_2$ :  $\tau := \text{get-next-symbol};$   
if  $\tau = \text{end-of-file}$  then  $\text{отхрани гума};$   
else if  $\tau = a$  then go to  $q_3$ ;  
else if  $\tau = b$  then go to  $q_2$ ;

$q_3$ :  $\tau := \text{get-next-symbol};$   
if  $\tau = \text{end-of-file}$  then  $\text{отхрани гума};$   
else if  $\tau = a$  then go to  $q_1$ ;  
else if  $\tau = b$  then go to  $q_3$ ;

За да се оценят сложността на алгоритъм от горните три за KDA  $M = (K, \Sigma, \delta, S, F)$  за всичко със сложното имаме да използваме  $|\Sigma| + 2$  временно място за промеждението, също како и при прочитането на сървъра. На гадена гума, отчитането на сървъра е също протича във времето  $|\Sigma| + 2$  временно място, и това времето е независимо от сървърите. Този алгоритъм има сложност  $O(|\Sigma| + 2) \cdot |W|$ . Така получихме сървърното търсение:

търсение. Ако  $L$  е персийски език, то съществува алгоритъм, който при прочитането на гадена гума във времето  $|L|$  има сложност  $O(|L|)$ . Да обясним това математично в теорията сложност-

Та забиси само от  $w$  и е  $O(|w|)$ . Това е таза защото сег като езикът е фиксиран, наричане автомат, който разпознава този език. Този автомат е фиксиран и за това всички останали параметри, размножени от  $w \in \Sigma^*$  са константи.

Как състои нечата за НКА? На пръв поглед трябва да организират експоненциална сложност, за разлика от полиномиалната сложност за КДА (дане и не е). Сега ще видим, че има алгоритъм, който е почти такъв ефективен, колкото и алгоритъма за КДА.

В този алгоритъм ние в извесен смисъл ще имплементираме детерминизација на НКА, но само за свидетелства, като са ни необходими за корекцията гума, а не за всички свидетели.

Нека си имаме  $M = (K, \Sigma, \Delta, S, F)$  е НКА и га разглеждаме алгоритъма:

$$S_0 := E(S), n := 0$$

repeat

$n := n + 1$ ,  $\sigma := n-th$  б ходът символ на гумата;

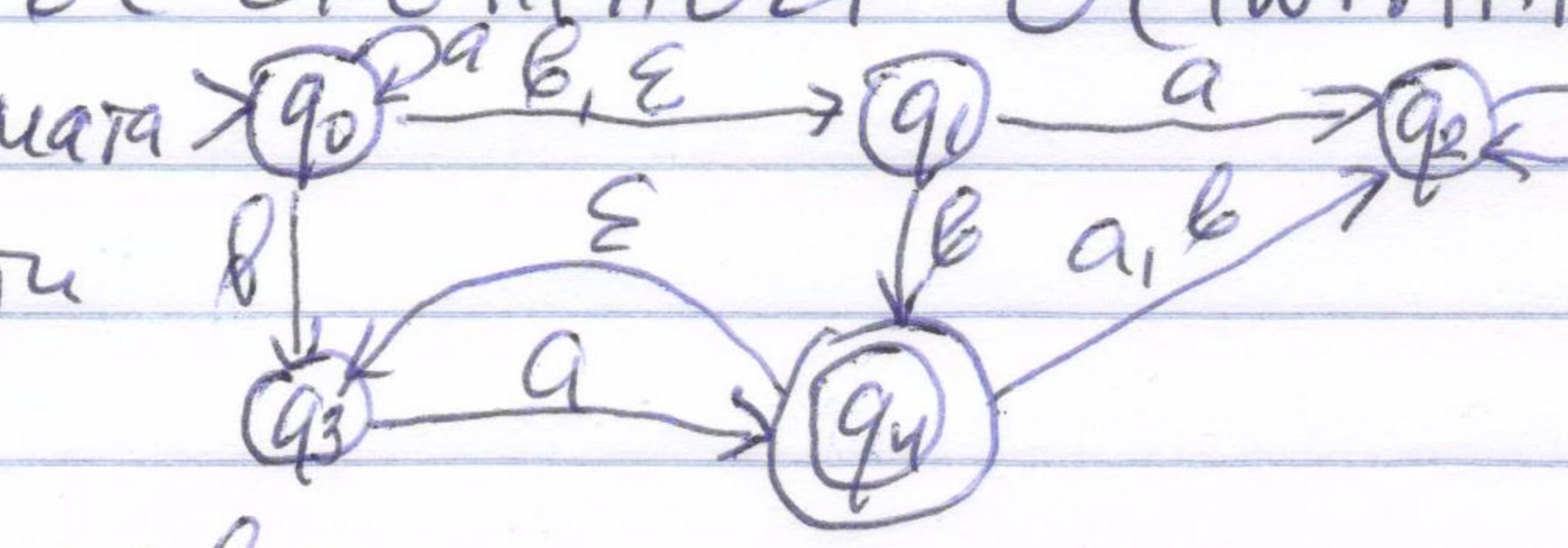
if  $\sigma \neq v_{pal}$  на гумата then

$$S_n := U \cup E(q) \mid \text{съвр. резултат} : (p, \sigma, q) \in \Delta \}$$

until  $\sigma = v_{pal}$  на гумата

if  $S_{n-1} \cap F \neq \emptyset$  then приеми гумата else отхвърли гумата

Този като тук напираме само тези  $E(q)$ , които са необходими за разпознаването на гумата, сложността не е експоненциална, а е  $O(|K|^2 \cdot |w|)$ . Така можем да тръгнем. Ако  $M = (K, \Sigma, \Delta, S, F)$  е НКА, то свидетелства алгоритъм, който може да гаджи гума  $w \in \Sigma^*$  проверка гуми  $w \in L(M)$  или не със сложност  $O(|w| \cdot |K|^2)$ .

За разглеждане на автомата  и га видим как работи алгоритъма ви. този автомат за гумата  $aaab$ .

$w = aaaba$

$S_0 = \{q_0, q_1\}$

$S_1 = \{q_0, q_1, q_2\}$

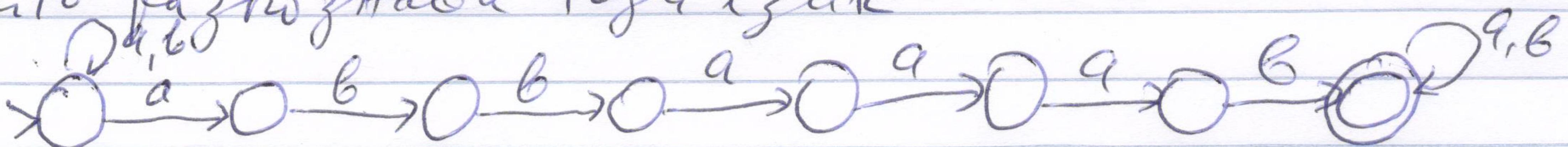
$S_2 = \{q_0, q_1, q_2\}$

$S_3 = \{q_0, q_1, q_2\}, S_4 = \{q_1, q_2, q_3, q_4\}, S_5 = \{q_2, q_3, q_4\}$

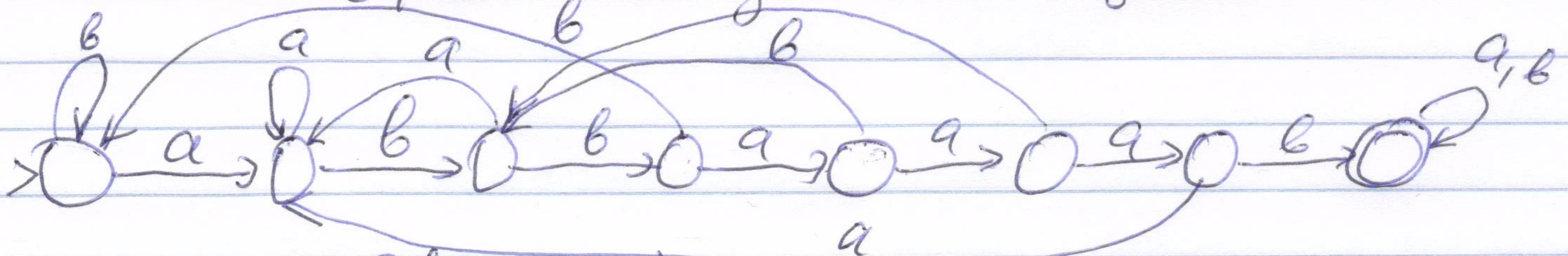
Така като  $S_5$  съдържа  $q_4$ , то  $w$  се разпознава от  $M$ .

Остава ли да проверите дали  $abbba$  се разпознава от  $M$  самостојително.

Един от най-болните проблеми в разглеждането на алгоритми, свързани с голям изоруци от думи в га-  
гов език (перфидни) е така наричаният проблем за „str-  
ing matching“. За разглеждане например алгоритъм  
за разпознаването езика  $L_x = \{w \mid w \in \Sigma^* \text{ и } x \in \text{подгру-}  
па на } w\}$ . За тази цел да разглеждаме думата  $x =$   
 $abbbaab$ . Много лесно можем да напишем НКА,  
които разпознава този език



Сложността на алгоритма, описан по-горе за  
НКА е  $O(|x|^2 \cdot |w|)$ . За целта на този алгоритъм е  
необходим много не-ефективен алгоритъм, като  
например написване на горния НКА да по сърочи  
KDA, който разпознава този език. Тогава съде  
много не ефективен за такива думи



а искато -  $O(|\Sigma| \cdot |w|)$  е неравна същност

## Контекстно-свободни езици

Контекстно свободните езици (KCE) или както още се срещат контекстно независимите езици са иззнати като отмът да анализират езикът във вид на езици като български, английски и т.н. Този отмът е бил приобщаван от стремежа за обработка на тези езици от компютри. Това е необходимо когато се очаква да се провери дали ИКС изреченията на български е правилно построено или пък искаже доводите да генерира правилно построени изречения. Така се стига до KCE и контекстно свободните граматики. Едновременно с това това да е следващата стапка: към усъвършенстване на разглежданите добуки изучавани модели като регуларните езици и крайните автомати. К тук и се създават оптимални модели както на те компютри, така и на разпознавачи. Може да се разглежда разпознавачите, наредени стекови автомати. Сега ще започнем със по-същественото от регуларните езици вариант на генератори, наричан KCE и свързаното с него контекстно свободните граматики (KCG). KCG ще определят правилата, но която ще построи съответните смислени изрази (изречения). Какво представлява KCG? Тогава ще фокусираме едното:

Д. KCG е четворката  $G = (V, \Sigma, R, S)$ , където  $V$  е фиксираната крайна азбука от символи,  $\Sigma \subseteq V$ , елементите на  $\Sigma$  се наричат терминирани символи или терминаци, елементите на  $V \setminus \Sigma$  се наричат нетерминаци,  $R$  е множество от правила,  $R \subseteq (V \setminus \Sigma) \times V^*$  и  $S \in V \setminus \Sigma$ ,  $S$  е начален символ.

Ако  $(A, u) \in R$ , то  $A$  е нетерминант, а  $u \in V^*$  е засега правилото  $(A, u)$  и ето го, означаване  $A \xrightarrow{u} u$  (правило от  $G$ )  
Д. Нека  $G = (V, \Sigma, R, S)$  е KCG. Тогава членът  $u \xrightarrow{G} V$  т.е.  $u, v \in V^*$  и съществуват  $x, y \in V^*$  и  $A \in V \setminus \Sigma$  и

С други деяни  $u \Rightarrow_G v$  когао икои негермашан  $A$ ,  
узастьеувану в деяни  $u$  заистим с деяни  $v'$ , така-  
ва, те  $A \rightarrow v'$  е правило в практиката  $G$ .

D. Hevia  $G = (V, \Sigma, R, S)$  e KCF. Toraba  $\in L(G)$  oz ita za-  
baile MHOHECSBON  $L(G) = \{w \mid w \in \Sigma^* \text{ u } S \Rightarrow_G^* w\}$  u ce  
Itapuzta eza k nrojeler os KCF G.

D. Egunz ezu L ce Itapura KCE T.T.K.  $L = L(G)$  za  
Hs kof KCF  $G = (V, \Sigma, R, S)$ .

Задо езидете от горният bug (кашто съзбъдните  
са матрици) се наричат KCE (KCF)? Причината е в  
преводата. Всичко правено в ефта KCF има bug  
 $A \rightarrow u$ , когато  $A$  е неприменим символ и  $u \in V^*$  и  
затвореното на  $A$  не зависи от некашб контекст,  
т.е. има правен bug  $Ax \rightarrow u$  или  $xAy \rightarrow u$ ,  
което га създаде "контекст"  $x$  и го пренесе като  
текст  $x, y$  за Има гуми  $x, y$  от  $V$ . Иде Има га  
се застапават с така наричани контексто-зависими  
(контекстуални) езици, които създадат подобен "контекст".  
Когато сме фокусирани на касаща Гадежно  
се става въпрос за  $G$ , вместо  $A \rightarrow u$  и  $u_1 \Rightarrow_G^* u_2$  ще им-  
ате вид  $A \rightarrow u$  и  $u \rightarrow_{G} u_2$

Всюка бе гулға  $W_0 \Rightarrow_G W_1 \Rightarrow_G W_2 \dots \Rightarrow_G W_n$  се Несуза избог  
б КСГ  $G$  на доказат  $W_n$  ос доказат  $W_0$ , ә и се Насуза  
жеки тиң на избога.

Пример 1. Имеа  $G = (V, \Sigma, R, S)$  е регуларна граматика:  
 $\Sigma = \{a, b\}$ ,  $V = \{S\}$ ,  $R = \{S \rightarrow \epsilon, S \rightarrow aSb, S \Rightarrow aSb\}$ . Тоба  
 $S \Rightarrow \epsilon$ ,  $S \Rightarrow aSb \Rightarrow ab$ ,  $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$   
и т.д., т.е.  $L(G) = \{a^n b^n \mid n \in \mathbb{N}\}$ . Тоба е пример на  
език, който се намира ос KCF, то не е персул НН,  
т.е.  $\{a^n b^n \mid n \in \mathbb{N}\} \notin KCE$ , то не е персул спр. Така ио-  
коато не е регуларен, за всички формални езици, т.е.

КСЕ разчитават регулрните езика. Това все пази  
гаве пример, свързан с изразите на изразителен на  
български език.

Пример 2.  $G = (W, \Sigma, R, S)$ , когато  $\Sigma = \{Христо,  
харесва, големи, зервена, & роги\}$ ,  $W = \{S, A, N, V, P\}$   
 $\Sigma, R = \{P \rightarrow N, P \rightarrow AP, S \rightarrow AP, S \rightarrow PVP, A \rightarrow \text{големи},  
A \rightarrow зервена, N \rightarrow & роги, N \rightarrow Христо, V \rightarrow харесва\}$ .

За обратния начин, за всички думи на български  
от  $\Sigma$  са същите като образни симболи (неграмати).

Обратно за  $S$  са имена на изразителни (sentences),  
 $P$  като фраза,  $V$  като пада,  $A$  като прилагателно,  $N$   
като съдържанието, Може да получим следните  
„изрази“ в тази КСГ:  $S \Rightarrow PVP \Rightarrow \text{Христо VP} \Rightarrow$   
 $\text{Христо харесва P} \Rightarrow \text{Христо харесва AP} \Rightarrow \text{Христо}$   
 $\text{харесва зервена P} \Rightarrow \text{Христо харесва зервена & роги}$ .

Но можем да получим и  $S \Rightarrow PVP \Rightarrow^* \text{Големи зер-}  
вени & роги харесва Големи Христо и т.н.$

Пример 3.  $G = (V, \Sigma, R, E)$ , когато  $\Sigma = \{+, \times, (,), id\}$ ,  
 $V = \Sigma \cup \{T, E, F\}$ ,  $R = \{E \rightarrow E + T, E \rightarrow T, T \rightarrow T * F, F \rightarrow F,  
F \rightarrow (E), F \rightarrow id\}$ . Тук за  $T$  са имена като ТЕМ,  
за  $E$  като израз, за  $F$  – идентитет.

$E \rightarrow T \Rightarrow T * F \Rightarrow T * (E) \Rightarrow T * (E + T) \Rightarrow T * (T + T) \Rightarrow$   
 $T * (F * T + T) \Rightarrow F * (F * F + T) \Rightarrow F * (F * F + F) \Rightarrow$   
 $id * (F * F + F) \Rightarrow id * (id * F + F) \Rightarrow id * (id * id + F) \Rightarrow$   
 $id * (id * id + id)$

Търпиме. Нека  $M = (K, \Sigma, \delta, S, F)$  е КДА. Тогава  
съществува КСГ  $G = (V, \Sigma, R, S)$  такава че  $L(M) = L(G)$ .  
Д-то. Нека  $G$  е същата:  $V = K \cup \Sigma$ ,  $S = S \cup R =$   
 $\{q \rightarrow qp | \delta(q, q) = p\} \cup \{p \rightarrow \varepsilon | p \in F\}$ . Следва да се  
изпровери, че  $L(G) = L(M)$ .