LFA Laborator 2

Miriam Costan

Martie 2020

Definitie. Un automat finit este un 5-tuplu $< Q, \Sigma, \delta, q_0, F >$, unde:

- Q reprezinta multimea finita a starilor.
- Σ reprezinta alfabetul.
- δ reprezinta functia de tranzitie. $\delta : Q \times \Sigma \to Q$, unde $\delta(x, \alpha) = y$ reprezinta faptul ca din starea x trecem in starea y la citirea caracterului α.
- q_0 starea initiala.
- $F \subset Q$, multimea starilor finale.

In laboratorul 2 vom scrie functia:

```
bool evaluate(automata, word) {
    if(word is accepted by the automata)
        return true;
    else
        return false;
}
```

Cu alte cuvinte, vrem un program care primeste ca input un automat(DFA, NFA sau λ-NFA) si un set de cuvinte, si decide daca respectivele cuvinte sunt acceptate de automat sau nu.

Formatul datelor de intrare va fi dupa cum urmeaza:

- n = numarul de stari. Starile vor fi numerotate de la 0 la $n - 1$.
- m = numarul de caractere din alfabet.
- m caratere reprezentand alfabetul.
- q_0 = starea initiala.
- k = numarul de stari finale.
- k numere diferite intre 0 si $n - 1$ reprezentand starile finale.

- l = numarul de tranzitii.
- l tranzitii de tipul x α y reprezentand faptul ca din starea x mergem in starea y cu caracterul α

Vom oferi indicatii de implementare pentru fiecare din cele trei cazuri, plus exemple de testare.

1 DFA accepter

1.1 Indicatii de implementare

1. Contruim matricea de tranzitie δ astfel:

δ	α_0	α_1	α_2	α_3
q_0	q_x	q_z	N/A	N/A
q_1	q_v	q_y	N/A	q_x
q_2	q_w	N/A	N/A	N/A
q_3	N/A	q_u	N/A	q_z

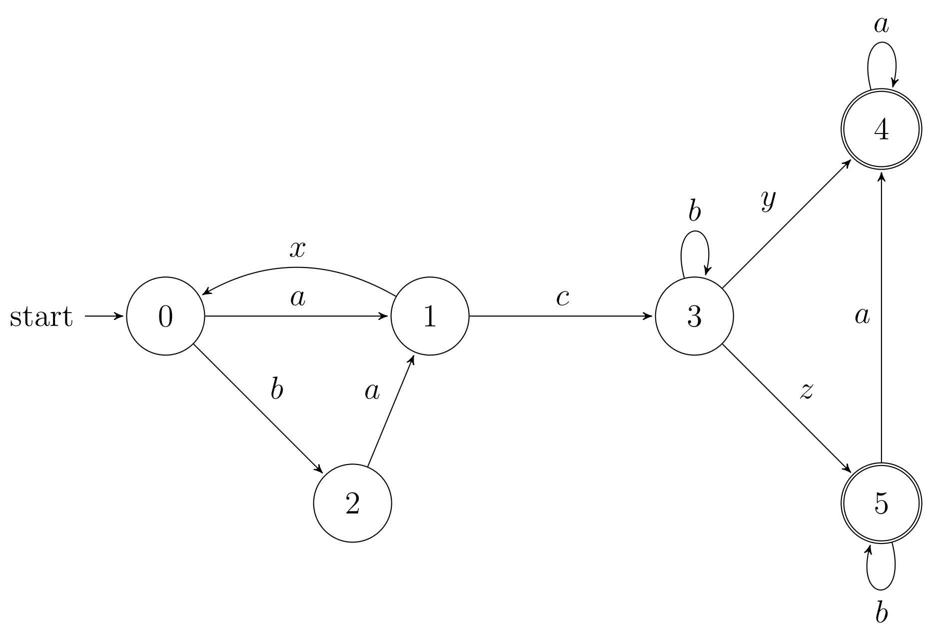
2. Retinem la fiecare pas starea curenta. Cand citim un nou caracter din cuvant, trecem la starea urmatoare corespunzator matricei de tranzitie.
3. Daca matricea de tranzitie nu are initializata celula care ne intereseaza δ (stare curenta, caracter curent), putem sa ne oprim, cuvantul nu poate fi acceptat de automat.
4. Ramane sa verificam atunci cand terminam de citit cuvantul, daca starea pe care ne-am oprit este finala.

1.2 Exemplu

```
6
6
a b c x y z
0
2
4 5
11
0 a 1
0 b 2
1 c 3
1 x 0
2 a 1
3 b 3
3 y 4
```

```
3 z 5
4 a 4
5 a 4
5 b 5

axbacbbzbbaaa - TRUE
axccbya - FALSE
axbac - FALSE
bacy - TRUE
bacyaaac - FALSE
```



2 NFA accepter

2.1 Indicatii de implementare

1. Matricea de tranzitie δ se modifica astfel:

δ	α_0	α_1	α_2	α_3
q_0	$\{\}$	$\{\}$	$\{\}$	$\{\}$
q_1	$\{\}$	$\{\}$	$\{\}$	$\{\}$
q_2	$\{\}$	$\{\}$	$\{\}$	$\{\}$
q_3	$\{\}$	$\{\}$	$\{\}$	$\{\}$

   In loc de o stare, matricea de tranzitii indica catre o multime de stari(posibil vida) in care putem sa ajungem la citirea fiecarui caracter.

2. In loc sa mai retinem starea curenta, retinem un set de stari curente in care putem ajunge.

3. La final este de ajuns ca una din starile in care ne-am oprit sa fie finala.

3 λ-NFA accepter

3.1 Indicatii de implementare

1. Matricea de tranzitie δ se modifica astfel:

δ	α_0	α_1	α_2	λ
q_0	$\{\}$	$\{\}$	$\{\}$	$\{\}$
q_1	$\{\}$	$\{\}$	$\{\}$	$\{\}$
q_2	$\{\}$	$\{\}$	$\{\}$	$\{\}$
q_3	$\{\}$	$\{\}$	$\{\}$	$\{\}$

   Introducem in matricea de trazitii caracterul λ

2. De tinut cont ca λ nu citeste niciun caracter din cuvant pentru a il folosi.

3. la fiecare pas cand actualizam starile curente, trebuie sa tinem cont si de starile in care ajungem cu λ tranzitie.

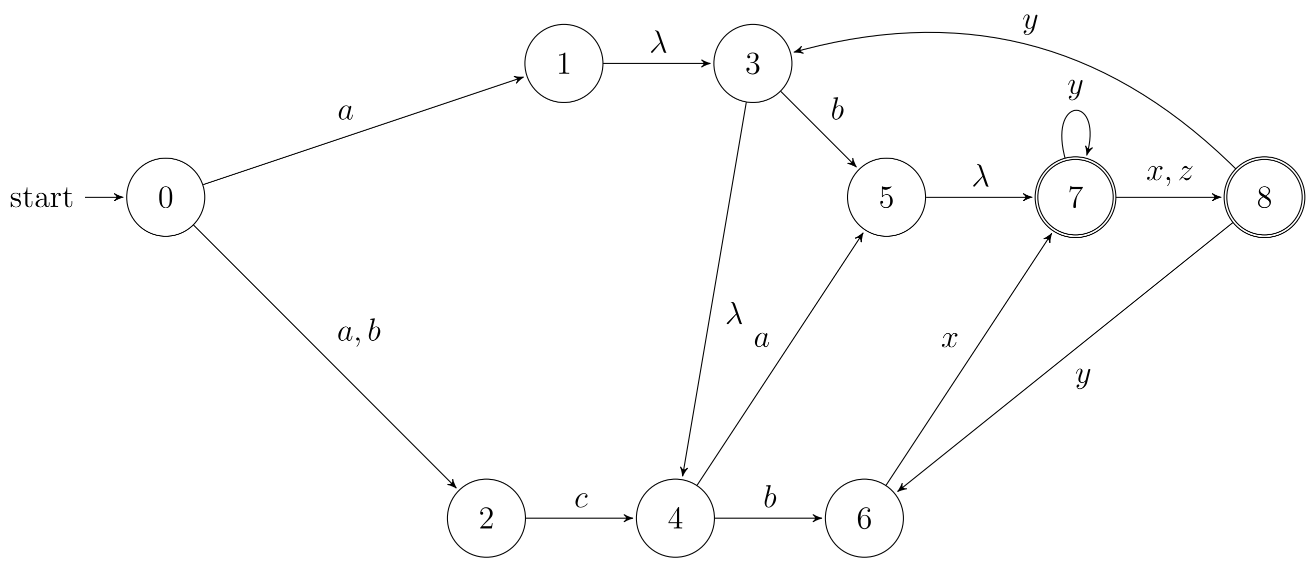
3.2 Exemplu

- **In fisierul din input λ va fi reprezentat cu ajutorul caracterului \$.**
- **NU exista λ - cicluri**.

```
9
6
a b c x y z
0
2
7 8
16
0 a 1
0 a 2
0 b 2
1 $ 3
2 c 4
3 b 5
3 $ 4
4 a 5
4 b 6
5 $ 7
6 x 7
```

```
7 y 7
7 x 8
7 z 8
8 y 6
8 y 3

abxyyyxyby - TRUE
bcax - TRUE
bcbxxy - FALSE
abyyxz - FALSE
abyyxyx - TRUE
```



LFA Laborator 3

Miriam Costan

Martie 2020

In urmatoarele doua laboratoare vom face conversia unui $\lambda - NFA$ la un DFA_{min}. Cu alte cuvinte vrem un program care primeste ca intrare un $\lambda - NFA$ si afiseaza DFA-ul echivalent cu un numar minim de stari.

0.1 Pasii conversiei.

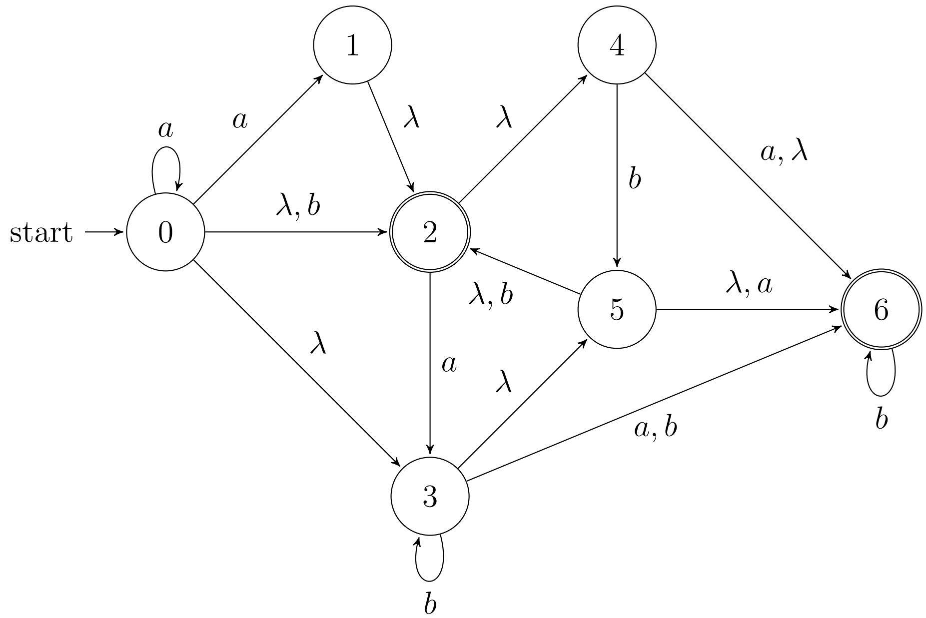
1. Laboratorul 3 - Conversie de la $\lambda - NFA$ la NFA.
2. Laboratorul 3 - Conversie de la NFA la DFA.
3. Laboratorul 4 - Conversie de la DFA la DFA_{min}.

0.2 Punctajul pe cea de-a doua tema.

- 5 **- Pentru oricare din cei** 3 **pasi.**
- 7 **- Pentru oricare** 2 **din cei** 3 **pasi.**
- 10 **- Pentru toti cei** 3 **pasi.**

1 $\lambda - NFA \to NFA$.

Avand dat un $\lambda - NFA$, vom construi NFA-ul echivalent prin eliminarea tranzitiilor de tip λ.



δ	a	b	λ
0	{0, 1}	{2}	{2, 3}
1	{}	{}	{2}
2	{3}	{}	{4}
3	{6}	{3, 6}	{5}
4	{6}	{5}	{2, 6}
5	{6}	{2}	{2, 6}
6	{}	{6}	{}

1.1 Pasul 1. Calcularea λ-inchiderii.

λ-inchiderea(λ^*) unei stari q reprezinta multimea de stari in care se poate ajunge plecand din q cu 0 sau mai multe tranzitii de tip λ.

Observatie 1. Orice stare face parte din propria sa λ-inchidere. Practic putem considera ca pentru orice stare exista o λ-tranzitie implicita catre ea insasi.

Observatie 2. λ-inchiderea unei multimi de stari este egala cu reuniunea λ-inchiderii fiecarei stari din multime.

λ-inchiderea automatului dat este:

	λ^*
0	{0, 2, 3, 4, 5, 6}
1	{1, 2, 4, 6}
2	{2, 4, 6}
3	{2, 3, 4, 5, 6}
4	{4, 6}
5	{2, 4, 5, 6}
6	{6}

1.2 Pasul 2. Calcularea functiei de tranzitie δ^*.

Cu ajutorul λ-inchiderii, putem calcula functia de tranzitie a NFA-ului pe care dorim sa il construim.

O stare q poate ajunge cu caracterul α in starile ce rezulta dupa concatenarea la stanga si la dreapta cu λ^*. Cu alte cuvinte starile rezultate din calcularea $\lambda^*\alpha\lambda^*$.

Exemplu pentru calcularea tranzitiilor cu caracterul a:

	λ^*	a	λ^*
0	{0, 2, 3, 4, 5, 6}	{0, 1, 3, 6}	{0, 1, 2, 3, 4, 5, 6}
1	{1, 2, 4, 6}	{3, 6}	{2, 3, 4, 5, 6}
2	{2, 4, 6}	{3, 6}	{2, 3, 4, 5, 6}
3	{2, 3, 4, 5, 6}	{3, 6}	{2, 3, 4, 5, 6}
4	{4, 6}	{6}	{6}
5	{2, 4, 5, 6}	{3, 6}	{2, 3, 4, 5, 6}
6	{6}	{}	{}

- Prima coloana reprezinta λ-inchiderea starii de pe linia respectiva.
- A doua coloana reprezinta reuniunea starilor accesibile cu caracterul a din fiecare din starile din λ-inchidere.
- A treia coloana reprezinta λ-inchiderea multimii de pe a doua coloana. Aceasta din urma va fi tranzitia cu caracterul a a NFA-ului construit.

Dupa calcularea caracterului b, matricea de tranzitii va arata dupa cum urmeaza:

δ^*	a	b
0	{0, 1, 2, 3, 4, 5, 6}	{2, 3, 4, 5, 6}
1	{2, 3, 4, 5, 6}	{2, 4, 5, 6}
2	{2, 3, 4, 5, 6}	{2, 4, 5, 6}
3	{2, 3, 4, 5, 6}	{2, 3, 4, 5, 6}
4	{6}	{2, 4, 5, 6}
5	{2, 3, 4, 5, 6}	{2, 4, 5, 6}
6	{}	{6}

1.3 Pasul 3. Calcularea starilor finale si initiale.

- Starea initiala ramane aceasi cu cea a automatlui initial, in cazul nostru 0.
- Starile finale vor fi toate starile care contin o stare finala din automatul initial in λ-inchidere, in cazul nostru toate starile sunt in aceasta situatie.

1.4 Pasul 4. Eliminarea starilor redundante.

Doua stari sunt identice daca au tranzitiile sunt identice pentru orice caracter din alfabet si daca amandoua sunt sau nu sunt stari finale. In cazul nostru $1, 2, 5$ sunt identice.

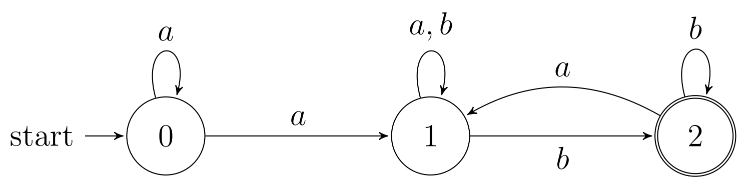
Observatie 3. Daca q si r sunt identice putem sa stergem starea r si sa inlocuim r cu q peste tot in tabelul de tranzitii.

Dupa stergerea starilor $2, 5$ si inlocuirea lor in tabelul de tranzitii, functia δ^* va arata dupa cum urmeaza:

δ^*	a	b
0	{0, 1, 3, 4, 6}	{1, 3, 4, 6}
1	{1, 3, 4, 6}	{1, 4, 6}
3	{1, 3, 4, 6}	{1, 3, 4, 6}
4	{6}	{1, 4, 6}
6	{}	{6}

2 $NFA \to DFA$.

Avand dat un NFA, vom construi DFA-ul echivalent prin eliminarea nedeterminismului.



δ	a	b
0	{0, 1}	{}
1	{1}	{1, 2}
2	{1}	{2}

2.1 Pasul 1. Eliminarea nedeterminismului.

Pornim cu o coada in care adaugam doar starea initiala q_0. Apoi pentru fiecare stare din coada q si fiecare caracter din alfabet α facem urmatorul calcul:
daca $\delta(q, \alpha) = q_{x_0}, ...q_{x_k}, k \geq 0$ atunci:

creem starea $q_{x_0...x_k}$(poate fi si o stare care nu este compusa)

Daca noua stare fromata $q_{x_0...x_k}$ nu a mai fost vizitata, atunci o adaugam in coada. Tranzitia acestei stari cu un caracter α va fi reuniunea starilor accesibile cu caracterul α din toate starile componente.

Repetam acest calcul pana cand coada devine vida.

Dupa adaugarea starilor $01, 12$ tabelul de tranzitii arata in felul urmator:

δ^*	a	b
0	01	
1	1	12
01	01	12
12	1	12

Observatie 4. Chiar daca numarul starilor poate creste exponential, asta nu se intampla de obicei in practica.

2.2 Pasul 2. Calcularea starilor initiale si finale.

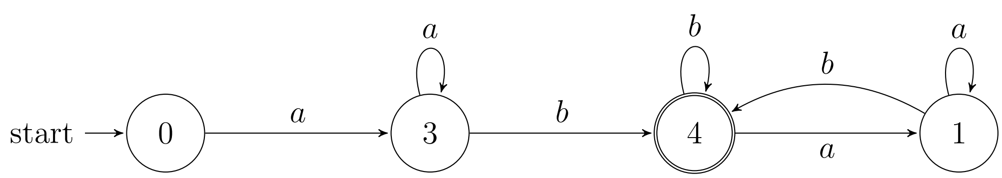
- Starea initiala ramane aceasi cu cea a automatlui initial, in cazul nostru 0.
- Starile finale vor fi toate starile care au in componenta o stare finala din automatul initial, in cazul nostru 2, 12.

2.3 Pasul 3. Redenumirea starilor.

Putem sa redenumim starile fara a afecta functionalitatea.

Redenumim $01 = 3, 12 = 4$ si obtinem:

δ^*	a	b
0	3	
1	1	4
3	3	4
4	1	4



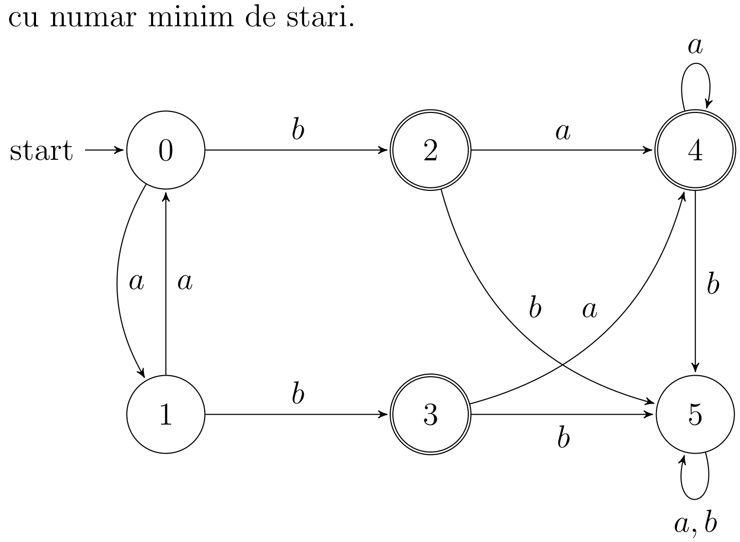
LFA Laborator 3

Miriam Costan

Aprilie 2020

1 $DFA \to DFA_{min}$.

Avand dat un DFA, vom construi DFA_{min}-ul echivalent, care accepta acelasi set de cuvinte dar cu numar minim de stari.



δ	a	b
0	1	2
1	0	3
2	4	5
3	4	5
4	4	5
5	5	5

1.1 Pasul 1. Determinarea starilor echivalente.

Doua stari sunt **echivalente** daca si numai daca pentru orice cuvant am alege, plecand din cele doua stari, ajungem in doua stari fie finale sau nefinale.

$\forall q, r \in Q, q \equiv r \iff [\forall \omega \in \Sigma^*, \delta(q, \omega) \in F \leftrightarrow \delta(r, \omega) \in F]$

Vom calcula starile echivalente in felul urmator:

1. Construim matricea de echivalenta si o marcam pe toata cu $TRUE$(consideram ca toate sunt echivalente).

≡	0	1	2	3	4	5
0	-	-	-	-	-	-
1	TRUE	-	-	-	-	-
2	TRUE	TRUE	-	-	-	-
3	TRUE	TRUE	TRUE	-	-	-
4	TRUE	TRUE	TRUE	TRUE	-	-
5	TRUE	TRUE	TRUE	TRUE	TRUE	-

Observatie 1. Marcam doar partea stanga jos, matricea fiind simetrica.

2. Marcam cu $FALSE$ toate perechile (q, r), unde q stare finala si r stare nefinala.

3. Marcam cu $FALSE$ toate perechile (q, r) pentru care $(\delta(q, \alpha), \delta(r, \alpha))$ sunt marcate cu $FALSE$, $\alpha \in \Sigma$.

4. Repetam 3 pana nu mai apar modificari.

≡	0	1	2	3	4	5
0	-	-	-	-	-	-
1	TRUE	-	-	-	-	-
2	FALSE	FALSE	-	-	-	-
3	FALSE	FALSE	TRUE	-	-	-
4	FALSE	FALSE	TRUE	TRUE	-	-
5	FALSE	FALSE	FALSE	FALSE	FALSE	-

Observam ca grupurile de stari echivalente sun $\{0, 1\}, \{2, 3, 4\}$ si $\{5\}$.

1.2 Pasul 2. Gruparea starilor echivalente si calcularea functiei de tranzitie δ^*.

Grupam starile echivalente rezultate din matricea de echivalenta intr-o unica stare. Tranzitiile vor fi aceleasi cu ale automatului initial dar tinand cont de aceasta grupare.

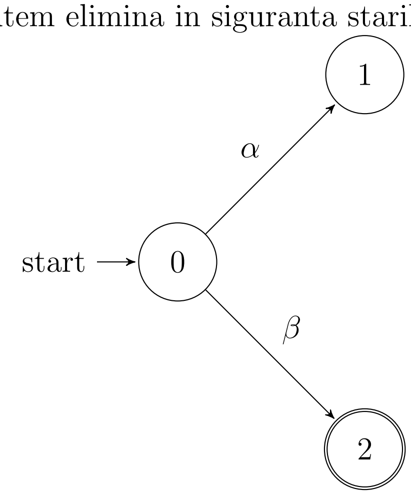
δ^*	a	b
01	01	234
234	234	5
5	5	5

1.3 Pasul 3. Calcularea starilor finale si initiale.

- Starea initiala devine starea ce contine starea initiala a automatului origina. In cazul nostru q_{01}.
- Starile finale sunt toate starile compuse din stari finale. In cazul nostru q_{234}.

1.4 Pasul 4. Eliminarea starilor dead-end.

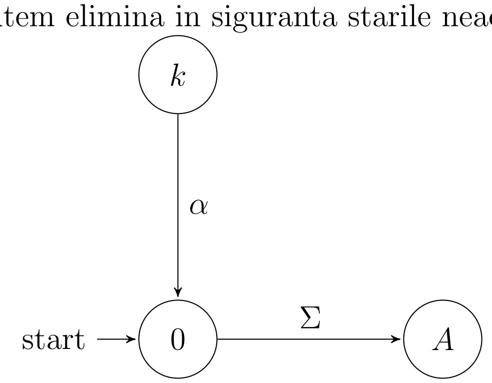
O stare q_k este dead-end daca nu exista niciun drum de la aceasta stare la o stare finala. Putem elimina in siguranta starile dead-end.



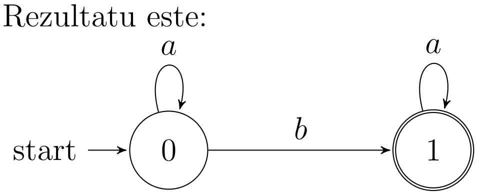
In exemplul de mai sus 1 este dead-end.
In automatul nostru, 5 este un dead-end si il putem elimina.

1.5 Pasul 5. Eliminarea starilor neaccesibile.

O stare q_k este neaccesibila daca nu exista niciun drum de la starea initala $q0$ pana la q_k. Putem elimina in siguranta starile neaccesibile.



In exemplul de mai sus, k nu este accesibil.
Rezultatu este:



LFA Laborator 5

Miriam Costan

Mai 2020

O gramatica este in **Forma Normala Chomsky(FNC)** daca are doar productii de forma:

$$\begin{aligned} A &\rightarrow a \\ A &\rightarrow BC \end{aligned} \tag{1}$$

Unde: A, B, C - neterminale
a - terminal

In laboratorul urmator vrem sa scriem o **Gramatica Independenta de Context(GIC)** in FNC. Scrierea se va face in 5 pasi. Punctajul va fi de 2 puncte pentru fiecare pas.

1 Eliminarea λ-productiilor.

Eliminam productiile care au ca membru cuvantul vid.

1. Gasim o productie $T \rightarrow \lambda$.
2. Daca T NU mai are si alte productii atunci:
   - Eliminam productia.
   - Il eliminam pe T din toate productiile in care apare ca membru drept in compuneri de mai mult de 2 simboluri. Ex. $A \rightarrow aT \Rightarrow A \rightarrow a$.
   - Transformam toate productiile in care T apare singur ca membru drept al unei productii din $U \rightarrow T$ in $U \rightarrow \lambda$

   Exemplu:

$$\begin{aligned} T &\rightarrow \lambda \\ A &\rightarrow T \\ B &\rightarrow CT \end{aligned} \tag{2}$$

   Dupa eliminatea lui $T \rightarrow \lambda$ rezulta: $A \rightarrow \lambda$ si $B \rightarrow C$.

   Altfel, daca T mai are si alte productii:

   - Eliminam productia.

- Pentru toate productiile in care T apare ca membru drept in compuneri de mai mult de 2 simboluri adaugam si productia fara T. Ex. $U \to TV \Rightarrow U \to TV|V$.

Exemplu:

$$\begin{aligned} &T \to \lambda|ab \\ &A \to T \\ &B \to CT \end{aligned} \tag{3}$$

Dupa eliminarea lui $T \to \lambda$ rezulta:

$$\begin{aligned} &T \to ab \\ &A \to T \\ &B \to CT|C \end{aligned} \tag{4}$$

3. Repeta 1. cat timp mai exista productii $T \to \lambda$.

2 Eliminarea redenumirilor.

Inlocuim productiile de tipul:

$$\begin{aligned} &V \to W \\ &W \to \alpha \end{aligned} \tag{5}$$

cu: $V \to \alpha$. De exemplu:

$$\begin{aligned} &S \to A \\ &A \to ab|bc|Bc \\ &B \to b \end{aligned} \tag{6}$$

Eliminam redenumirea $S \to A$.

$$\begin{aligned} &S \to ab|bc|Bc \\ &A \to ab|bc|Bc \\ &B \to b \end{aligned} \tag{7}$$

3 Eliminarea productiilor inutile.

Eliminam productiile neaccesibile din simbolul de start.

$$\begin{aligned} &S \to ab \\ &A \to bc \end{aligned} \tag{8}$$

In exemplul de mai sus A nu este accesibil din S si poate fi eliminat in siguranta.

Eliminam si productiile care nu se termina, de tipul: $A \to aA$, acestea neputand face parte dintr-un cuvant.

Observatie 1. Atunci cand eliminam o productie inutila $T \to \alpha$, daca neterminalul din partea stanga T NU mai are si alte productii, atunci stergem toate productiile in care apare neterminalul.

$$\begin{aligned} &S \to aA|a \\ &A \to Aa \end{aligned} \tag{9}$$

In exemplul de mai sus productia $A \to Aa$ este inutila. A nu mai are si alte productii deci poate fi sters. Astfel, stergem si productia $S \to aA$. Rezultatul final va fi:

$$S \to a \tag{10}$$

4 Adaugarea de neterminale noi pentru terminalele din productii, daca acestea sunt insotite de cel putin un alt terminal sau neterminal.

Pentru toti terminalii care apar in productii in compuneri de mai mult de un simbol, inlocuim terminalul cu un neterminal nou, si adaugam o productie de la noul neterminal la terminal.

$$T \to aU|ab \tag{11}$$

Inlocuim a cu X_a si b cu X_b.

$$\begin{aligned} &T \to X_aU|X_aX_b \\ &X_a \to a \\ &X_b \to b \end{aligned} \tag{12}$$

5 Adaugarea de neterminale noi pentru productiile de mai mult de doua neterminale.

1. Gasim o productie $T \to T_0T_1...T_n$, cu $n \geq 2$.
2. Inlocuim $T_1...T_n$ cu un nou neterminal $T_{1,2,...,n}$.
3. Adaugam productia $T_{1,2,...,n} \to T_1...T_n$
4. Repeta 1 pana nu mai sunt productii $T \to T_0T_1...T_n$, cu $n \geq 2$.

$$T \to T_0T_1T_2 \tag{13}$$

Inlocuim T_2T_2 cu $T_{1,2}$ si adaugam productia $T_{1,2} \to T_1T_2$.

$$\begin{aligned} &T \to T_0T_{1,2} \\ &T_{1,2} \to T_1T_2 \end{aligned} \tag{14}$$