

~ Seminar 2 ~

Automat Finit Nedeterminist cu λ -tranziții

AFN- $\lambda = (Q, \Sigma, q_0, \delta, F)$

Q mulțimea de stări

Σ alfabetul de intrare

$q_0 \in Q$ starea inițială

$F \subseteq Q$ mulțimea de stări finale

$\delta: Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$ funcția de tranziție („delta”)

Diferența față de AFN-ul simplu este că suntem într-o stare și putem citi fie un caracter din alfabetul Σ , fie cuvântul vid λ . Deci se poate întâmpla să ajungem într-o stare nouă fără să fi citit nicio literă nouă din cuvântul de intrare, ci doar aplicând λ -tranziții.

λ -închiderea unei stări

Mulțimea de stări în care se poate ajunge plecând din starea q și aplicând zero sau mai multe λ -tranziții se numește „ λ -închiderea stării q ” și se notează cu $\langle q \rangle$.

Obs: Orice stare face parte din propria λ -închidere (pentru că $\delta(q, \lambda^0) = q$; practic putem presupune că orice stare are o λ -tranziție *implicită* către ea însăși).

$$\langle q \rangle = \bigcup_{k \geq 0} \{r \mid r \in \delta(q, \lambda^k)\}$$

$$\langle q \rangle = \{q\} \cup \{q_i \mid q_i \in \delta(q, \lambda^1)\} \cup \{q_{ij} \mid q_{ij} \in \delta(q, \lambda^2)\} \cup \dots$$

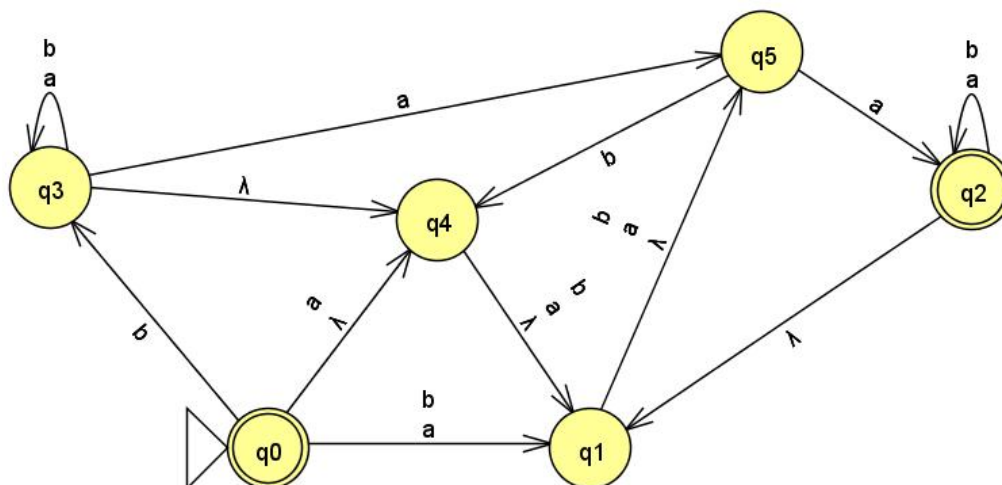
Observăm că mulțimile se pot calcula inductiv după puterea lui λ :

$$\{q_{ij} \mid q_{ij} \in \delta(q, \lambda^2)\} = \{q_{ij} \mid q_i \in \delta(q, \lambda^1), q_{ij} \in \delta(q_i, \lambda^1)\}.$$

$$\text{Sau în general } \{r \mid r \in \delta(q, \lambda^k)\} = \{r \mid s \in \delta(q, \lambda^{k-1}), r \in \delta(s, \lambda^1)\}.$$

Obs: λ -închiderea unei mulțimi de stări este egală cu reuniunea λ -închiderilor acelor stări.

$$\langle \{q_{i1}, q_{i2}, \dots, q_{in}\} \rangle = \langle q_{i1} \rangle \cup \langle q_{i2} \rangle \cup \dots \cup \langle q_{in} \rangle$$



EX_1: Să se calculeze λ -închiderile tuturor stărilor din automatul desenat mai sus.

$$\delta(q_0, \lambda^0) = \{q_0\}$$

$$\delta(q_0, \lambda^1) = \dots$$

$$\delta(q_0, \lambda^2) = \dots$$

...

$$\rightarrow \langle q_0 \rangle = \{q_0, \dots\}$$

Transformarea AFN- $\lambda \rightarrow$ AFD

Pas 1: Având dat un AFN- λ (reprezentat sub formă de graf), vom calcula funcția delta_AFN- λ completând un tabel asemănător cu cel de la algoritmul de transformare AFN \rightarrow AFD (*discutat în primul seminar*).

- La acest tabel, capetele de coloană vor fi caracterele din alfabet, dar concatenate la stânga și la dreapta cu λ^* .
- Pentru a calcula cele $|Q|$ linii ale primului tabel:
 - facem λ -închiderea stării de la capătul de linie curent
 - apoi în toate stările obținute încercăm să citim caracterul de pe coloana curentă și obținem o nouă mulțime de stări
 - facem λ -închiderea acestei mulțimi și ceea ce obținem scriem în tabel

Pas 2: Apoi facem al doilea tabel, pentru funcția delta_AFD, care va avea la capetele de coloană caracterele din alfabet.

- Pornim de la starea inițială a AFD-ului, care va fi λ -închiderea stării inițiale a automatului AFN- λ .
- Pentru a calcula următoarele linii (corespunzătoare noilor stări-mulțime care apar), procedăm la fel ca la celălalt algoritm, adică reunim ce găsim în tabel pe aceeași coloană și pe liniile potrivite stărilor componente din starea-mulțime; iar dacă noile stări sunt simple, atunci le copiem liniile din primul tabel.
- Stările finale ale noului AFD obținut, vor fi acele stări-mulțime (pot avea și un singur element, nu neapărat mai multe) care conțin cel puțin una din stările care erau finale pentru automatul AFN- λ inițial.

EX_2: Să se transforme automatul din desen (pag 1) într-un AFD echivalent.

$$q_0 \xrightarrow{\lambda^*} \dots \xrightarrow{a} \dots \xrightarrow{\lambda^*} \dots$$

$\delta_{\text{AFN-}\lambda}$	$\lambda^* \cdot a \cdot \lambda^*$	$\lambda^* \cdot b \cdot \lambda^*$
$q_0 \in F$		
q_1		
$q_2 \in F$		
q_3		
q_4		
q_5		

δ_{AFD}	a	b
$\langle q_0 \rangle = \dots$		

AFD \rightarrow

Verificare acceptare cuvânt de către automat AFN- λ

Pentru a verifica dacă un cuvânt este sau nu acceptat de un automat AFN- λ :

Se procedează analog ca în cazul AFN-ului, doar că înainte de a căuta toate stările posibile de continuare cu tranziții cu caracterul curent, trebuie să facem λ -închiderea mulțimii curente de stări. Iar după ce a fost citit tot cuvântul de intrare, trebuie să facem o ultimă λ -închidere a stărilor curente, pentru a obține mulțimea finală de stări în care poate ajunge automatul pentru cuvântul dat.

Deci vom pleca din starea inițială și vom alterna

- pasul (1) în care facem λ -închiderea mulțimii curente de stări cu
 - pasul (2) în care din fiecare stare, cu litera curentă din cuvânt, căutăm în ce stare se ajunge
- La început și la sfârșit aplicăm pasul (1).

EX_3: Să se verifice dacă AFN- λ din desen (pag 1) acceptă sau nu cuvântul abbabba.

Obs: în loc de săgeată trebuie să folosiți simbolul \vdash (mie nu îmi merge când scriu formula).

AFN- λ :

$(q_0, abbabba) \xrightarrow{\lambda^*} (q_{0145}, \text{abbabba}) \xrightarrow{a} (q \dots, \text{bbabba})$
 $\xrightarrow{\lambda^*} (q \dots, \text{bbabba}) \xrightarrow{b} (q \dots, \text{babba}) \dots$
.....
.....
 $\xrightarrow{\lambda^*} (q \dots, \lambda)$

Gramatici

$G = (N, T, S, P)$

$N = \{A, B, C, \dots\}$ mulțimea de simboluri neterminale

$T = \{a, b, c, \dots\}$ mulțimea de simboluri terminale

$S \in N$ simbolul de start

P mulțimea de producții (sau reguli de producție)

Modul în care este definită funcția P determină diferite tipuri de gramatici.

Gramatici regulate

Gramaticile regulate au producții de tipurile

$A \rightarrow \lambda$

$A \rightarrow a$

$A \rightarrow aB, \text{ cu } a \in T \text{ și } A, B \in N$


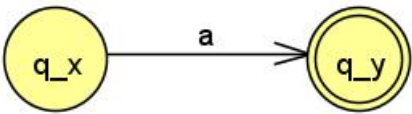
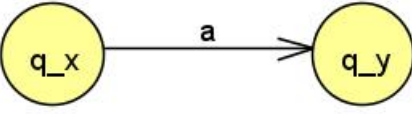
Obs: Putem grupa mai multe producții care au același membru stâng și putem scrie pe scurt

$A \rightarrow \lambda \mid a \mid aB, \text{ cu } a \in T \text{ și } A, B \in N$

Gramatici regulate vs Automate Finite:

- Gramaticile **generează cuvinte** pornind mereu de la *simbolul de start* (S). Pentru neterminalul (litera mare) din cuvânt se caută o **producție** în care apare el ca membru stâng și se înlocuiește în cuvânt cu membrul drept al producției alese. La fiecare pas se va genera încă un terminal (litera mică) (cuvântul va conține în continuare un singur neterminal la fiecare pas), iar la final (folosind o producție de tip “caz de oprire”) neterminalul va dispărea fiind înlocuit fie de o literă mică, fie de λ .

- Automatele **acceptă** (sau resping) cuvinte pornind mereu de la *starea inițială* (q_0). La fiecare pas se caută o **tranziție** din starea curentă cu litera curentă din cuvânt și automatul se mută în starea unde ajunge tranziția. Automatul cceptă cuvântul dacă a reușit să parcurgă toate literele și a ajuns într-o stare finală.

La automatele finite	La gramaticile regulate
	$A_F \rightarrow \lambda$ $A_F \in N$
	$A_x \rightarrow a$ $A_x \in N$ $a \in T$
	$A_x \rightarrow aA_y$ $A_x, A_y \in N$ $a \in T$

EX_4: Pentru următoarele limbaje, scrieți producțiile unei gramatici regulate: