~ Seminar 3 ~

➢ *Algoritm:* Transformarea AFN-λ → AFN *[vezi curs 3, pag 16 – 19]*

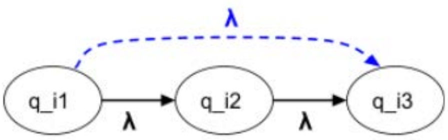
Se dă un automat AFN-λ. Se cere să se construiască un automat AFN echivalent.

Idee: Adăugăm tranziții cu litere din alfabet și stări finale astfel încât să simulăm comportamentul λ-tranzițiilor, pe care apoi le eliminăm.

→ Pas 1 („λ-completion”):

- Cât timp e posibil:

$$\forall q_{i_1}, q_{i_2}, q_{i_3} \in Q, \text{daca } \delta(\boldsymbol{q_{i_1}}, \boldsymbol{\lambda}) \ni \boldsymbol{q_{i_2}} \text{ si } (\boldsymbol{q_{i_2}}, \boldsymbol{\lambda}) \ni \boldsymbol{q_{i_3}} \Rightarrow (\boldsymbol{q_{i_1}}, \boldsymbol{\lambda}) \ni \boldsymbol{q_{i_3}}$$



Altfel spus, din fiecare stare **q** trebuie să adăugăm (dacă nu există deja) câte o λ-tranziție către fiecare stare **r** (cu $r \neq q$) din mulțimea <**q**> (λ-închiderea stării q).

(Adică dacă aveam de la starea q la starea r un drum format din două sau mai multe λ-tranziții, atunci trebuie să adaugăm o λ-tranziție directă de la q la r.)

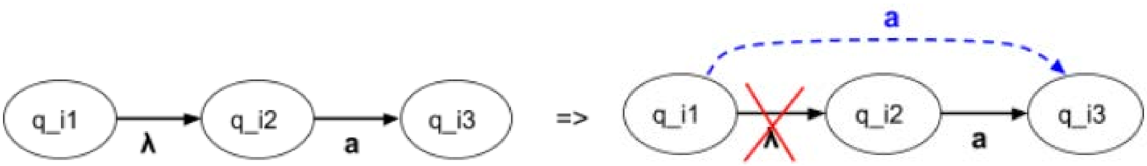
- Apoi trebuie să adăugăm la mulțimea stărilor finale acele stări din care cu λ ajungem într-o stare finală.

(Adică stările care conțineau în λ-închiderea lor cel puțin o stare finală vor deveni și ele stări finale.)

→ Pas 2 („λ-transition removal”):

- Adăugăm tranziții cu litere din alfabet care să înlocuiască efectul λ-tranzițiilor, astfel:

$$\forall q_{i_1}, q_{i_2}, q_{i_3} \in Q, \forall a \in \Sigma, \text{daca } \delta(\boldsymbol{q_{i_1}}, \boldsymbol{\lambda}) \ni \boldsymbol{q_{i_2}} \text{ si } (\boldsymbol{q_{i_2}}, \boldsymbol{a}) \ni \boldsymbol{q_{i_3}} \Rightarrow (\boldsymbol{q_{i_1}}, \boldsymbol{a}) \ni \boldsymbol{q_{i_3}}$$

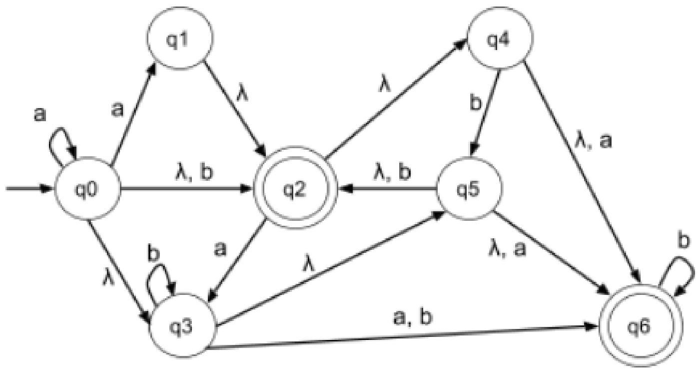


Adică dacă din starea **q** puteam ajunge în starea **r** citind ***λa*** ($\forall a \in \Sigma$), atunci adăugăm o tranziție cu litera **a** direct de la **q** la **r**.

- Apoi eliminăm toate λ-tranzițiile din automat.

- ***Exemplu:*** Se dă următorul AFN–λ.

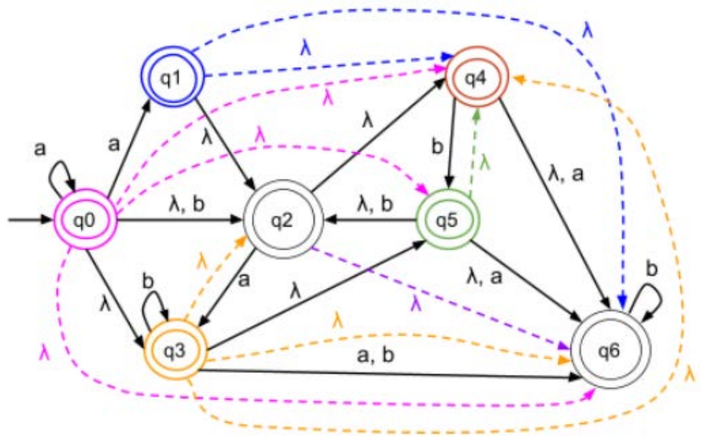
(Același din seminarul 2, pentru care calculasem deja λ-închiderile tuturor stărilor.)



→ Pas 1 („λ-completion"):

$< q_0 >= \{q_0, q_2, q_3, q_4, q_5, q_6\}$ => adaug λ de la q_0 spre q_4, q_5, q_6
$< q_1 >= \{q_1, q_2, q_4, q_6\}$ => adaug λ de la q_1 spre q_4, q_6
$< q_2 >= \{q_2, q_4, q_6\}$ => adaug λ de la q_2 spre q_6
$< q_3 >= \{q_3, q_5, q_2, q_6, q_4\}$ => adaug λ de la q_3 spre q_2, q_4, q_6
$< q_4 >= \{q_4, q_6\}$ => nu adaug nimic
$< q_5 >= \{q_5, q_2, q_4, q_6\}$ => adaug λ de la q_5 spre q_4
$< q_6 >= \{q_6\}$ => nu adaug nimic

Toate stările au în λ-închiderile lor cel puțin o stare finală, deci toate stările vor deveni finale.

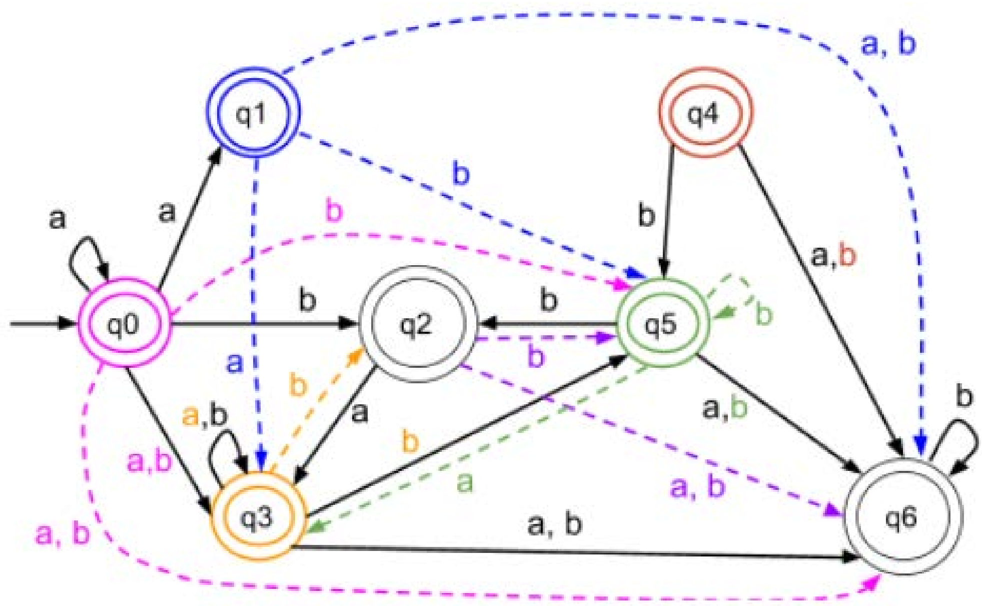


→ Pas 2 („λ-transition removal"):

Vom elimina: $\delta(q_{i1}, \boldsymbol{\lambda}) \ni q_{i2}$	Avem: $\delta(q_{i2}, \boldsymbol{x}) \ni q_{i3}, \boldsymbol{x} \in \boldsymbol{\Sigma}$	Adăugăm: $\delta(q_{i1}, \boldsymbol{x}) \ni q_{i3}$
$\delta(q_0, \lambda) \ni q_2$	$\delta(q_2, a) \ni q_3$	$\delta(q_0, a) \ni q_3$
$\delta(q_0, \lambda) \ni q_3$	$\delta(q_3, a) \ni q_6$ $\delta(q_3, b) \supset \{q_3, q_6\}$	$\delta(q_0, a) \ni q_6$ $\delta(q_0, b) \supset \{q_3, q_6\}$
$\delta(q_0, \lambda) \ni q_4$	$\delta(q_4, a) \ni q_6$ $\delta(q_4, b) \ni q_5$	$\delta(q_0, a) \ni q_6$ $\delta(q_0, b) \ni q_5$
$\delta(q_0, \lambda) \ni q_5$	$\delta(q_5, a) \ni q_6$ $\delta(q_5, b) \ni q_2$	$\delta(q_0, a) \ni q_6$ $\delta(q_0, b) \ni q_2$
$\delta(q_0, \lambda) \ni q_6$	$\delta(q_6, b) \ni q_6$	$\delta(q_0, b) \ni q_6$

Vom elimina: $\delta(q_{i1}, \boldsymbol{\lambda}) \ni q_{i2}$	Avem: $\delta(q_{i2}, \boldsymbol{x}) \ni q_{i3}, \boldsymbol{x} \in \boldsymbol{\Sigma}$	Adăugăm: $\delta(q_{i1}, \boldsymbol{x}) \ni q_{i3}$
$\delta(q_1, \lambda) \ni q_2$	$\delta(q_2, a) \ni q_3$	$\delta(q_1, a) \ni q_3$
$\delta(q_1, \lambda) \ni q_4$	$\delta(q_4, a) \ni q_6$ $\delta(q_4, b) \ni q_5$	$\delta(q_1, a) \ni q_6$ $\delta(q_1, b) \ni q_5$
$\delta(q_1, \lambda) \ni q_6$	$\delta(q_6, b) \ni q_6$	$\delta(q_1, b) \ni q_6$
$\delta(q_2, \lambda) \ni q_4$	$\delta(q_4, a) \ni q_6$ $\delta(q_4, b) \ni q_5$	$\delta(q_2, a) \ni q_6$ $\delta(q_2, b) \ni q_5$
$\delta(q_2, \lambda) \ni q_6$	$\delta(q_6, b) \ni q_6$	$\delta(q_2, b) \ni q_6$
$\delta(q_3, \lambda) \ni q_2$	$\delta(q_2, a) \ni q_3$	$\delta(q_3, a) \ni q_3$
$\delta(q_3, \lambda) \ni q_4$	$\delta(q_4, a) \ni q_6$ $\delta(q_4, b) \ni q_5$	$\delta(q_3, a) \ni q_6$ $\delta(q_3, b) \ni q_5$
$\delta(q_3, \lambda) \ni q_5$	$\delta(q_5, a) \ni q_6$ $\delta(q_5, b) \ni q_2$	$\delta(q_3, a) \ni q_6$ $\delta(q_3, b) \ni q_2$
$\delta(q_3, \lambda) \ni q_6$	$\delta(q_6, b) \ni q_6$	$\delta(q_3, b) \ni q_6$
$\delta(q_4, \lambda) \ni q_6$	$\delta(q_6, b) \ni q_6$	$\delta(q_4, b) \ni q_6$
$\delta(q_5, \lambda) \ni q_2$	$\delta(q_2, a) \ni q_3$	$\delta(q_5, a) \ni q_3$
$\delta(q_5, \lambda) \ni q_4$	$\delta(q_4, a) \ni q_6$ $\delta(q_4, b) \ni q_5$	$\delta(q_5, a) \ni q_6$ $\delta(q_5, b) \ni q_5$
$\delta(q_5, \lambda) \ni q_6$	$\delta(q_6, b) \ni q_6$	$\delta(q_5, b) \ni q_6$
$\delta(q_6, \lambda) = \emptyset$ =>		=> Nu avem ce adăuga din q_6.

Am obținut un AFN echivalent cu AFN-λ dat.
(Observăm că starea q4 nu este accesibilă din starea inițială, deci ar putea fi eliminată împreună cu trazițiile ei fără a afecta limbajul recunoscut de automat.)



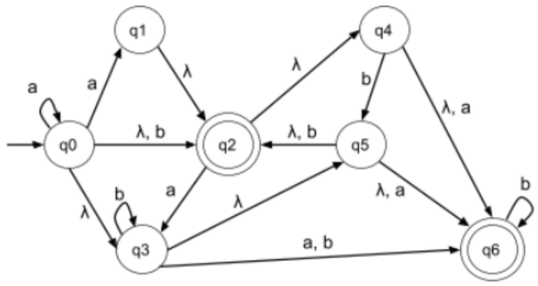
➢ *Algoritm:* Transformarea AFN-λ → AFD *[asemănător AFN → AFD, seminar 2]*

Idee: Dacă în AFN-λ din starea **q** citind $\boldsymbol{\lambda^* x \lambda^*}$ ($\forall x \in \Sigma$) se ajunge în mulțimea de stări **R**, atunci în AFN din starea **q** citind litera **x** se va ajunge în mulțimea de stări **R**.

Starea inițială pentru AFN este aceeași ca la AFN-λ. Stările finale ale AFN-ului sunt cele ale căror λ-închideri în AFN-λ conțin cel puțin o stare finală.

Obs: Dacă dorim să obținem AFD, atunci starea inițială din AFD este λ-închiderea stării inițiale din AFN-λ. Stările finale ale AFD-ului sunt cele care conțin cel puțin o stare care în AFN-λ avea în λ-închidere cel puțin o stare finală.

• ***Exemplu:*** Se dă următorul AFN–λ.
(Același din seminarul 2, pentru care calculasem deja λ-închiderile tuturor stărilor.)



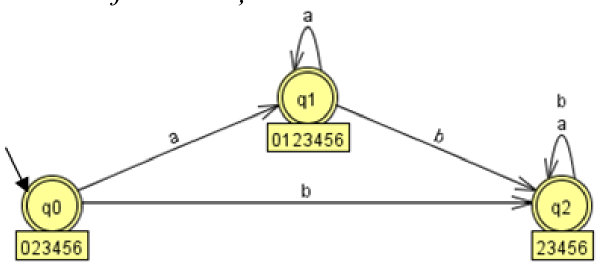
δ_AFN-λ	a	b	λ	λ* (λ-închiderea)
q0 init	{q0, q1}	{q2}	{q2, q3}	<q0> = {q0, q2, q3, q4, q5, q6}
q1	Ø	Ø	{q2}	<q1> = {q1, q2, q4, q6}
q2 in F	{q3}	Ø	{q4}	<q2> = {q2, q4, q6}
q3	{q6}	{q3, q6}	{q5}	<q3> = {q3, q5, q2, q6, q4}
q4	{q6}	{q5}	{q6}	<q4> = {q4, q6}
q5	{q6}	{q2}	{q2, q6}	<q5> = {q5, q2, q4, q6}
q6 in F	Ø	{q6}	Ø	<q6> = {q6}

în AFN-λ:	$\lambda^*\, a\, \lambda^*$	$\lambda^*\, b\, \lambda^*$
q0 init	$q_0 \xrightarrow{\lambda^*} q_{023456} \xrightarrow{a} q_{0136} \xrightarrow{\lambda^*} q_{0123456}$	$q_0 \xrightarrow{\lambda^*} q_{023456} \xrightarrow{b} q_{0356} \xrightarrow{\lambda^*} q_{23456}$
q1	$q_1 \xrightarrow{\lambda^*} q_{1246} \xrightarrow{a} q_{36} \xrightarrow{\lambda^*} q_{23456}$	$q_1 \xrightarrow{\lambda^*} q_{1246} \xrightarrow{b} q_{56} \xrightarrow{\lambda^*} q_{2456}$
q2 in F	$q_2 \xrightarrow{\lambda^*} q_{246} \xrightarrow{a} q_{36} \xrightarrow{\lambda^*} q_{23456}$	$q_2 \xrightarrow{\lambda^*} q_{246} \xrightarrow{b} q_{56} \xrightarrow{\lambda^*} q_{2456}$
q3	$q_3 \xrightarrow{\lambda^*} q_{23456} \xrightarrow{a} q_{36} \xrightarrow{\lambda^*} q_{23456}$	$q_3 \xrightarrow{\lambda^*} q_{23456} \xrightarrow{b} q_{2356} \xrightarrow{\lambda^*} q_{23456}$
q4	$q_4 \xrightarrow{\lambda^*} q_{46} \xrightarrow{a} q_6 \xrightarrow{\lambda^*} q_6$	$q_4 \xrightarrow{\lambda^*} q_{46} \xrightarrow{b} q_{56} \xrightarrow{\lambda^*} q_{2456}$
q5	$q_5 \xrightarrow{\lambda^*} q_{2456} \xrightarrow{a} q_{36} \xrightarrow{\lambda^*} q_{23456}$	$q_5 \xrightarrow{\lambda^*} q_{2456} \xrightarrow{b} q_{256} \xrightarrow{\lambda^*} q_{2456}$
q6 in F	$q_6 \xrightarrow{\lambda^*} q_6 \xrightarrow{a} \emptyset \xrightarrow{\lambda^*} \emptyset$	$q_6 \xrightarrow{\lambda^*} q_6 \xrightarrow{b} q_6 \xrightarrow{\lambda^*} q_6$

δ_AFN	*a*	*b*
q0 init, in F	$q_{0123456} = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$	$q_{23456} = \{q_2, q_3, q_4, q_5, q_6\}$
q1 in F	$q_{23456} = \{q_2, q_3, q_4, q_5, q_6\}$	$q_{2456} = \{q_2, q_4, q_5, q_6\}$
q2 in F	$q_{23456} = \{q_2, q_3, q_4, q_5, q_6\}$	$q_{2456} = \{q_2, q_4, q_5, q_6\}$
q3 in F	$q_{23456} = \{q_2, q_3, q_4, q_5, q_6\}$	$q_{23456} = \{q_2, q_3, q_4, q_5, q_6\}$
q4 in F	$\{q_6\}$	$q_{2456} = \{q_2, q_4, q_5, q_6\}$
q5 in F	$q_{23456} = \{q_2, q_3, q_4, q_5, q_6\}$	$q_{2456} = \{q_2, q_4, q_5, q_6\}$
q6 in F	Ø	$\{q_6\}$

δ_AFD	*a*	*b*
<q0> = q_{023456} init, in F	$q_{0123456}$	q_{23456}
$q_{0123456}$ in F	$q_{0123456}$	q_{23456}
q_{23456} in F	q_{23456}	q_{23456}

Am obținut un AFD echivalent cu automatele AFN și AFN-λ de mai sus.
Ce limbaj recunoaște acest AFD?



➢ ***Lema de pompare*** *pentru limbaje regulate (REG)* *[vezi curs 5, pag 19 – 21]*

Fie L un limbaj regulat. Atunci $\exists p \in \mathbb{N}$ (număr natural) astfel încât pentru $\forall \alpha \in L$ cuvânt, cu $|\alpha| \geq p$ și există o descompunere $\alpha = u \cdot v \cdot w$ cu proprietățile:

(1) $|u \cdot v| \leq p$

(2) $|v| \geq 1$

(3) $u \cdot v^i \cdot w \in L, \ \forall i \geq 0.$

- Vrem să demonstrăm că un limbaj NU este regulat.

Presupunem prin reducere la absurd că “limbajul este regulat” (predicatul P) și atunci rezultă că “afirmația din lemă este adevărată” (predicatul Q).

Obs: Știm de la logică faptul că $(P \rightarrow Q) \equiv (\neg Q \ \rightarrow \ \neg P)$. Așa că vom nega afirmația lemei $(\neg Q)$ și va rezulta că limbajul nu este regulat $(\neg P)$. Practic negarea constă în interschimbarea cuantificatorilor logici (∃ și ∀) între ei, iar la condiția (3) ∈ devine ∉.

→ ***Schema demonstrației***

- Vrem să demonstrăm că L **nu este** limbaj regulat (adică nu se poate construi niciun automat finit care să-l recunoască pe L), folosind lema de pompare ***negată***.

- Presupunem prin reducere la absurd că L **este** limbaj regulat. Atunci $\exists p \in \mathbb{N}$ (unde $p = |Q|$ numărul de stări ale unui automat finit care-l recunoaște pe L) și putem aplica lema de pompare. *(În continuare negăm afirmația lemei.)*

- **Alegem** (adică $\exists$) un cuvânt $\boldsymbol{\alpha}$ din limbajul L, care să respecte ipoteza lemei de pompare, adică să aibă lungimea cel puțin p, deci $|\alpha| \geq p, \forall p \in \mathbb{N}$.

- **Pentru fiecare** (adică $\forall$) posibilă descompunere a cuvântului $\alpha = u \cdot v \cdot w$ care respectă condițiile (1) și (2) din lemă ($|u \cdot v| \leq p$ și $|v| \geq 1$):

- **alegem** (adică $\exists$) convenabil câte un număr natural $\boldsymbol{i} \geq 0$ pentru care să obținem o contradicție a condiției (3), adică să rezulte că cuvântul $\beta = u \cdot v^i \cdot w \notin L$ și deci presupunerea făcută este falsă.

Observație: Demonstrația este completă și corectă doar dacă se obține contradicție ($\beta \notin L$) pentru toate descompunerile posibile ale cuvântului α.

- ***Exemplu:*** $\boldsymbol{L_0} = \{\boldsymbol{a^n b^n} \mid \boldsymbol{n} \geq \boldsymbol{0}\} \notin \boldsymbol{REG}$ *(discutat la curs 4 și 5)*

Idee: Vrem să demonstrăm că L_0 **nu este limbaj regulat**. Observăm că L_0 conține cuvinte formate din *n* litere de ”a” urmate tot de *n* litere de ”b”. În demonstrație trebuie să obținem un cuvânt β care să **nu** respecte această proprietate (adică să fie de forma a^*b^*, dar să aibă număr *diferit* de a-uri și b-uri).

Obs: Dacă aveam condiția $n \geq x$ (cu $x \in \mathbb{N}$ o constantă), atunci ar fi trebuit să alegem cuvântul $\alpha = a^{p+x}b^{p+x} \in L_0$ (pentru a fi sigur un cuvânt din limbaj $\forall p \in \mathbb{N}$).

Demonstrație: Presupunem prin reducere la absurd că L_0 este limbaj regulat. Atunci $\exists p \in \mathbb{N}$ și putem aplica lema de pompare. *(În continuare negăm afirmația lemei.)*

Alegem cuvântul $\boldsymbol{\alpha} = \boldsymbol{a^p b^p} \in \boldsymbol{L_0}$, cu $|\alpha| = 2p \geq p, \forall p \in \mathbb{N}$ *(deci lungimea cuvântului respectă ipoteza lemei)*. Conform lemei, cuvântul poate fi scris sub forma $\alpha = u \cdot v \cdot w$.

Din condiția (1) avem $|u \cdot v| \leq p$. Rezultă că cuvântul $u \cdot v$ conține doar litere de ”a” (pentru că $u \cdot v$ este un prefix al primelor p caractere din α).

Atunci notăm $\boldsymbol{v} = \boldsymbol{a^k}$. Din condițiile (1) și (2) avem $1 \leq |v| \leq p$ *(pentru că se poate ca $u = \lambda$, adică $|u| = 0$)*. Deci $1 \leq |a^k| \leq p$, adică $1 \leq k \leq p$ (*).

De asemenea, condiția (3) spune că $u \cdot v^i \cdot w \in L_0$, $\forall i \geq 0$. Alegem $\boldsymbol{i} = \boldsymbol{2}$ și avem cuvântul $\boldsymbol{\beta} = u \cdot v^2 \cdot w = \boldsymbol{a^{p+k}b^p} \notin \boldsymbol{L_0}$ (pentru că în relația (*) avem $k \geq 1$, deci numărul de ”a”-uri este strict mai mare decât numărul de ”b-uri” din cuvântul β). Am obținut o contradicție pentru (3), deci presupunerea făcută este falsă și L_0 nu este limbaj regulat. ■

- ***Exemplu:*** $\boldsymbol{L_1 = \{a^m b^n \mid m > n \geq 0\} \notin REG}$ *(discutat la seminar 3)*

Idee: Vrem să demonstrăm că L_1 **nu este limbaj regulat**. Observăm că L_1 conține cuvinte formate din litere de ”a” urmate de strict mai puține litere de ”b”. În demonstrație trebuie să obținem un cuvânt β care să **nu** respecte această proprietate (adică să fie de forma a^*b^*, dar să aibă $|\beta|_a \leq |\beta|_b$).

Obs: Ca să putem ajunge la contradicție, trebuie să alegem cuvântul α care respectă la limită inegalitatea dintre a-uri și b-uri (adică m exact cu o unitate mai mare decât n). De asemenea, b-urile trebuie să existe (deci alegem $n \neq 0$).

Demonstrație: Presupunem prin reducere la absurd că L_1 este limbaj regulat. Atunci $\exists p \in \mathbb{N}$ și putem aplica lema de pompare. *(În continuare negăm afirmația lemei.)*

Alegem cuvântul $\boldsymbol{\alpha = a^{p+1}b^p \in L_1}$, cu $|\alpha| = 2p + 1 \geq p, \forall p \in \mathbb{N}$ *(deci lungimea cuvântului respectă ipoteza lemei)*. Conform lemei, cuvântul poate fi scris sub forma $\alpha = u \cdot v \cdot w$.

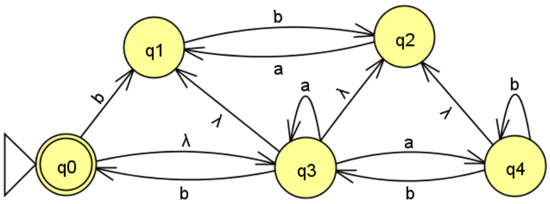
Din condiția (1) avem $|u \cdot v| \leq p$. Rezultă că cuvântul $u \cdot v$ conține doar litere de ”a” (pentru că $u \cdot v$ este un prefix al primelor p caractere din α).

Atunci notăm $\boldsymbol{v = a^k}$. Din condițiile (1) și (2) avem $1 \leq |v| \leq p$ *(pentru că se poate ca $u = \lambda$, adică $|u| = 0$)*. Deci $1 \leq |a^k| \leq p$, adică $1 \leq k \leq p$ (*).

De asemenea, condiția (3) spune că $u \cdot v^i \cdot w \in L_1, \ \forall i \geq 0$. Alegem $\boldsymbol{i = 0}$ și avem cuvântul $\boldsymbol{\beta} = u \cdot v^0 \cdot w = u \cdot w = \boldsymbol{a^{p+1-k}b^p}$. Conform (*) avem $1 \leq k \leq p$ (înmulțim cu -1) <=> $-p \leq -k \leq -1$ (adunăm p+1) <=> $1 \leq p + 1 - k \leq p$. Avem $|\beta|_a \leq |\beta|_b$ => $\boldsymbol{\beta \notin L_1}$**.** Am obținut o contradicție pentru (3), deci presupunerea făcută este falsă și L_1 nu este limbaj regulat. ■

~ Temă ~

EX_1: Se dă următorul AFN–λ *(același din tema seminar 2)*. Să se construiască un AFN și un AFD echivalente cu el.



(a) Transformați AFN–λ în AFN (folosind algoritmul de la curs: pas1 “λ–completion”, apoi pas2 “λ–transition removal”). Desenați AFN-ul.
(b) Calculați λ-închiderile tuturor stărilor. Transformați AFN–λ în AFD (calculând tabelul cu drumurile de forma $\lambda^* x \lambda^*, \forall x \in \Sigma$, apoi tabelul pentru AFD). Desenați AFD-ul.
(c) Explicați care este limbajul recunoscut de AFD-ul obținut.

EX_2: Demonstrați că următoarele limbaje nu sunt regulate, folosind lema de pompare.
$L2 = \{a^{2n}b^{n+3}c^n \mid n \geq 1\} \notin REG$
$L3 = \{a^m b^{3n} c^{n+3} \mid m \geq 5, n \geq 1\} \notin REG$