Limbaje Formale, Automate şi Compilatoare

Curs 7

2014-15

Curs 7

1. Automate pushdown deterministe
2. Lema Bar-Hillel pentru $\mathcal{L}_2$
3. Maşini Turing

Curs 7

1. Automate pushdown deterministe
2. Lema Bar-Hillel pentru $\mathcal{L}_2$
3. Maşini Turing

Definiţie 1

Automatul pushdown $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ *este determinist dacă funcţia de tranziţie* $\delta : Q \times (\Gamma \cup \{\epsilon\}) \times \Gamma \longrightarrow 2^{Q \times \Gamma^*}$ *îndeplineşte condiţiile:*

1. $|\delta(q, a, z)| = 1, \forall a \in \Sigma \cup \{\epsilon\},\ \forall q \in Q,\ \forall z \in \Gamma$
2. *Dacă* $\delta(q, \epsilon, z) \neq \emptyset$ *atunci* $\delta(q, a, z) = \emptyset, \forall a \in \Sigma$

Un automat pushdown determinist poate avea ϵ-tranziţii

Definiţie 1

Automatul pushdown $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ *este determinist dacă funcţia de tranziţie* $\delta : Q \times (\Gamma \cup \{\epsilon\}) \times \Gamma \longrightarrow 2^{Q \times \Gamma^*}$ *îndeplineşte condiţiile:*

1. $|\delta(q, a, z)| = 1, \forall a \in \Sigma \cup \{\epsilon\}, \ \forall q \in Q, \ \forall z \in \Gamma$
2. *Dacă* $\delta(q, \epsilon, z) \neq \emptyset$ *atunci* $\delta(q, a, z) = \emptyset, \forall a \in \Sigma$

Un automat pushdown determinist poate avea ϵ-tranziţii

$M = (\{q_0, q_1, q_2\}, \{0, 1, a\}, \{0, 1, z\}, \delta, q_0, z, \{q_2\})$

1. $\delta(q_0, i, z) = \{(q_0, iz)\}, (i \in \{0, 1\})$
2. $\delta(q_0, i, j) = \{(q_0, ij)\}, (i, j \in \{0, 1\})$
3. $\delta(q_0, a, i) = \{(q_1, i)\}$
4. $\delta(q_1, i, i) = \{(q_1, \epsilon)\}$
5. $\delta(q_1, \epsilon, z) = \{(q_2, \epsilon)\}$

$L(M) = \{waw^R | w \in \{0, 1\}^*\}$

$\mathcal{L}_{2DET}$ - Limbaje de tip 2 deterministe

$\mathcal{L}_{2DET} = \{L | \exists M$ automat pushdown determinist astfel ca $L = L(M)\}$.

- Clasa $\mathcal{L}_{2DET}$ este o clasă proprie a clasei de limbaje $\mathcal{L}_2$ ($\mathcal{L}_{2DET} \subset \mathcal{L}_2$).
- $\{ww^R | w \in \{0,1\}^*\} \in \mathcal{L}_2 \setminus \mathcal{L}_{2DET}$

$\mathcal{L}_{2DET}$ - Limbaje de tip 2 deterministe

$\mathcal{L}_{2DET} = \{L | \exists M$ automat pushdown determinist astfel ca $L = L(M)\}$.

- Clasa $\mathcal{L}_{2DET}$ este o clasă proprie a clasei de limbaje $\mathcal{L}_2$ ($\mathcal{L}_{2DET} \subset \mathcal{L}_2$).
- $\{ww^R | w \in \{0,1\}^*\} \in \mathcal{L}_2 \setminus \mathcal{L}_{2DET}$
  $M = (\{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, z\}, \delta, q_0, z, \{q_2\})$
  1. $\delta(q_0, i, z) = \{(q_0, iz)\}$, $(i \in \{0, 1\})$
  2. $\delta(q_0, i, j) = \{(q_0, ij)\}$, $(i, j \in \{0, 1\},\ i \neq j)$
  3. $\delta(q_0, i, i) = \{(q_0, ii),\ (q_1, \epsilon)\}$
  4. $\delta(q_1, i, i) = \{(q_1, \epsilon)\}$
  5. $\delta(q_1, \epsilon, z) = \{(q_2, \epsilon)\}$

$\mathcal{L}_{2DET}$ - Limbaje de tip 2 deterministe

Definiţie 2

O gramatică G este deterministă dacă:

- *Orice regulă este de forma $A \to a\alpha$, unde $a \in T$ iar $\alpha \in (N \cup T)^*$*
- *Pentru orice $A \in N$, dacă $A \to a\alpha$, $A \to b\alpha'$ sunt reguli, atunci $a \neq b$*

Pentru orice gramatică deterministă G există un automat pushdown determinist M astfel ca $L(G) = L(M)$

Curs 7

1. Automate pushdown deterministe
2. Lema Bar-Hillel pentru $\mathcal{L}_2$
3. Maşini Turing

Lema Bar-Hillel (de pompare) pentru $\mathcal{L}_2$

Lema 2.1

Pentru orice limbaj L de tip 2 există o constantă n astfel încât pentru orice cuvânt $w \in L$, $|w| \geq n$, există o descompunere $w = xyzuv$ cu proprietăţile:

1. $|yu| \geq 1$
2. $|yzu| \leq n$
3. $xy^izu^iv \in L, \forall i \geq 0$

Lema Bar-Hillel (de pompare) pentru $\mathcal{L}_2$

Lema 2.1

Pentru orice limbaj L de tip 2 există o constantă n astfel încât pentru orice cuvânt $w \in L$, $|w| \geq n$, există o descompunere $w = xyzuv$ cu proprietăţile:

1. $|yu| \geq 1$
2. $|yzu| \leq n$
3. $xy^izu^iv \in L, \forall i \geq 0$

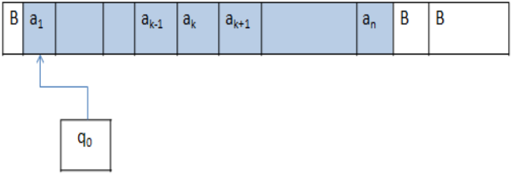
Următoarele limbaje nu sunt de tip 2:

- $L = \{a^nb^nc^n | n \geq 1\}$
- $L = \{a^nb^ma^nb^m | n \geq 1\}$
- $L = \{a^ib^jc^k | i \neq j, k \text{ şi } j \neq k\}$
- $L = \{ww | w \in \{a, b\}^*\}$

Curs 7

Maşini Turing

- Alan Turing 1936
- Număr finit de stari
- Banda de intrare este infinită în ambele părţi
- Capul de citire se poate mişca în ambele direcţii
- Pe banda de intrare se pot citi şi scrie simboluri



Maşini Turing - definiţie

Definiţie 3

O maşină Turing deterministă este un 6-uplu $M = (Q, \Sigma, \delta, \Gamma, q_0, F)$

- Q *este mulţimea finită a stărilor*
- q_0 *este starea iniţială*
- $F \subseteq Q$ *este mulţimea stărilor finale*
- Σ *este alfabetul de intrare*
- $\Gamma \supseteq \Sigma \cup \{B\}$ *este alfabetul benzii ce include alfabetul de intrare şi un simbol special, blanc*
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ *este funcţia de tranziţie care poate fi parţial definită*

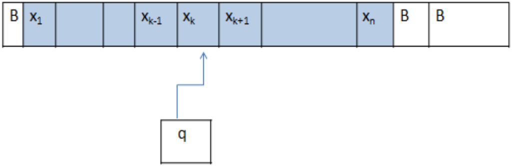
Maşini Turing - tranziţii

Fie $M = (Q, \Sigma, \delta, \Gamma, q_0, F)$ o maşină Turing.

- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$
  - $\delta(q, x) = (q', y, L)$: din starea q, dacă M vizează o celulă cu conţinutul x, trece în starea q', schimbă conţinutul celulei în y şi trece la celula din stânga
  - $\delta(q, x) = (q', y, R)$: din starea q, dacă M vizeaza o celulă cu conţinutul x, trece în starea q', schimbă conţinutul celulei în y şi trece la celula din dreapta
  - $\delta(q, x)$ nedefinit: maşina se opreşte

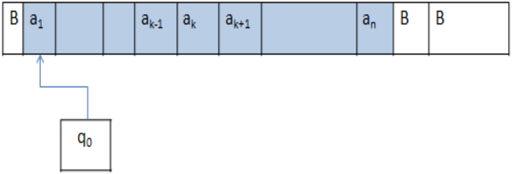
Maşini Turing - configuraţii

- Configuraţie: $x_1x_2\dots x_{k-1}q\uparrow x_kx_{k+1}\dots x_n$
  $x_1, x_2, \dots, x_n \in \Gamma, q \in Q$



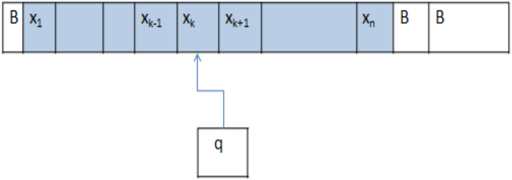
Maşini Turing - configuraţii

- Configuraţia iniţială: $q_0 \uparrow a_1 a_2 \ldots a_k a_{k+1} \ldots a_n$
  $a_1, a_2, \ldots, a_n \in \Sigma$



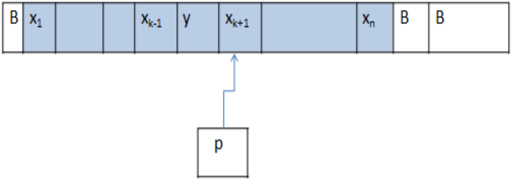
Maşini Turing - tranziţii

- Fie configuraţia $x_1x_2 \ldots x_{k-1}q \uparrow x_kx_{k+1} \ldots x_n$ şi $\delta(q, x_k) = (p, y, R)$



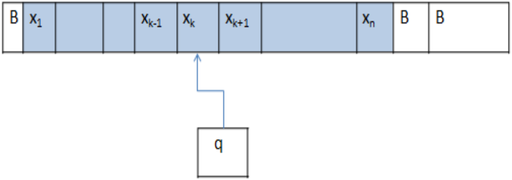
Maşini Turing - tranziţii

- $x_1 x_2 \ldots x_{k-1} q \uparrow x_k x_{k+1} \ldots x_n \vdash x_1 x_2 \ldots x_{k-1} y p \uparrow x_{k+1} \ldots x_n$
  $(\delta(q, x_k) = (p, y, R))$



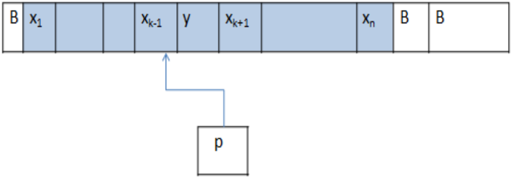
Maşini Turing - configuraţii

- Fie configuraţia $x_1x_2 \ldots x_{k-1}q \uparrow x_kx_{k+1} \ldots x_n$ şi $\delta(q, x_k) = (p, y, L)$



Maşini Turing - tranziţii

- $x_1x_2\ldots x_{k-1}q\uparrow x_kx_{k+1}\ldots x_n \vdash x_1x_2\ldots p\uparrow x_{k-1}yx_{k+1}\ldots x_n$
  $(\delta(q,x_k)=(p,y,L))$



Maşini Turing - limbaj acceptat

- Configuraţie finală: $\alpha_1 q \uparrow \alpha_2$, $q \in F$, $\alpha_1, \alpha_2 \in \Gamma^*$
- Configuraţie de blocare:
  $C = \alpha q \uparrow a\beta$, astfel încât nu există C' cu $C \vdash C'$ ($\delta(q,a)$ nedefinit)
- Calcul: închiderea reflexivă şi tranzitivă a relaţiei de mai sus: dacă $C_1, \ldots, C_n$ configuraţii astfel astfel încât :

$$C_1 \vdash C_2 \vdash \ldots \vdash C_n,$$

atunci:

$$C_1 \vdash^+ C_n \text{ dacă } n > 1,\ C_1 \vdash^* C_n \text{ dacă } n > 0$$

Maşini Turing - limbaj acceptat

Fie $M = (Q, \Sigma, \delta, \Gamma, q_0, F)$ o maşină Turing.

Maşini Turing - limbaj acceptat

Fie $M = (Q, \Sigma, \delta, \Gamma, q_0, F)$ o maşină Turing.

- Se ”încarcă” simbolurile lui w în celule consecutive pe banda de intrare, restul celulelor conţin simbolul special blanc

Maşini Turing - limbaj acceptat

Fie $M = (Q, \Sigma, \delta, \Gamma, q_0, F)$ o maşină Turing.

- Se ”încarcă” simbolurile lui w în celule consecutive pe banda de intrare, restul celulelor conţin simbolul special blanc
- Se poziţionează controlul la primul simbol din w şi la starea iniţială

Maşini Turing - limbaj acceptat

Fie $M = (Q, \Sigma, \delta, \Gamma, q_0, F)$ o maşină Turing.

- Se ”încarcă” simbolurile lui w în celule consecutive pe banda de intrare, restul celulelor conţin simbolul special blanc
- Se poziţionează controlul la primul simbol din w şi la starea iniţială
- Se produc tranziţiile conform cu δ

Maşini Turing - limbaj acceptat

Fie $M = (Q, \Sigma, \delta, \Gamma, q_0, F)$ o maşină Turing.

- Se ”încarcă” simbolurile lui w în celule consecutive pe banda de intrare, restul celulelor conţin simbolul special blanc
- Se poziţionează controlul la primul simbol din w şi la starea iniţială
- Se produc tranziţiile conform cu δ
- Maşina M acceptă w, dacă se ajunge într-o configuraţie finală

Maşini Turing - limbaj acceptat

Fie $M = (Q, \Sigma, \delta, \Gamma, q_0, F)$ o maşină Turing.

- Se "încarcă" simbolurile lui w în celule consecutive pe banda de intrare, restul celulelor conţin simbolul special blanc
- Se poziţionează controlul la primul simbol din w şi la starea iniţială
- Se produc tranziţiile conform cu δ
- Maşina M acceptă w, dacă se ajunge într-o configuraţie finală
- Maşina M respinge w dacă se ajunge într-o configuraţie de blocare

Maşini Turing - limbaj acceptat

Fie $M = (Q, \Sigma, \delta, \Gamma, q_0, F)$ o maşină Turing.

- Se "încarcă" simbolurile lui w în celule consecutive pe banda de intrare, restul celulelor conţin simbolul special blanc
- Se poziţionează controlul la primul simbol din w şi la starea iniţială
- Se produc tranziţiile conform cu δ
- Maşina M acceptă w, dacă se ajunge într-o configuraţie finală
- Maşina M respinge w dacă se ajunge într-o configuraţie de blocare
- M nu se opreşte pentru w, dacă nu există o configuraţie C, cu $q_0 \uparrow w \vdash^* C$

Maşini Turing - limbaj acceptat

Notaţii:

- $accept(M)$: mulţimea cuvintelor acceptate de M
- $reject(M)$: mulţimea cuvintelor respinse de M
- $loop(M)$: mulţimea cuvintelor pentru care M nu se opreşte
- $accept(M) \cup reject(M) \cup loop(M) = \Sigma^*$

Maşini Turing - limbaj acceptat

Limbaj acceptat

$$L(M) = \{w \in \Sigma^* | q0 \uparrow w \vdash^* \alpha_1 q \uparrow \alpha_2,\ q \in F,\ \alpha_1, \alpha_2 \in \Gamma^*\}$$

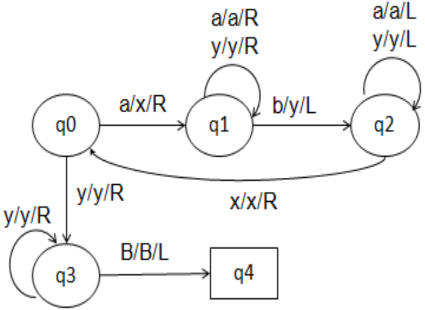
- $accept(M) = L(M)$
- Clasa de limbaje acceptate de maşini Turing: $\mathcal{L}_0$

Exemplu

$M = (\{q_0, q_1, q_2, q_3, q_4\}, \{a, b\}, \{a, b, x, y, B\}, \delta, q_0, \{q_4\})$
$L(M) = \{a^n b^n | n \geq 1\}$

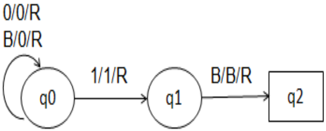
	a	b	x	y	B
q0	(q1,x,R)	_	_	(q3,y,R)	_
q1	(q1,a,R)	(q2,y,L)	_	(q1,y,R)	_
q2	(q2,a,L)	_	(q0,x,R)	(q2,y,L)	_
q3	_	_	_	(q3,y,R)	(q4,B,R)
q4	_	_	_	_	_

Exemplu



Exemplu

- MT pentru $L = \{a^n b^n c^n | n \geq 1\}$?
- Ce limbaj acceptă următoarea MT?



Maşini Turing - limbaj acceptat

Definiţie 4

Un limbaj L este recursiv enumerabil dacă există o maşină Turing M astfel încât $L(M) = L$.

- este posibil ca M să nu se oprească pe cuvinte de intrarea $w \notin L$
- Clasa limbajelor recursiv enumerabile: $\mathcal{L}_{re}$
- $\mathcal{L}_{re} = \mathcal{L}_0$

Maşini Turing - limbaj acceptat

Definiţie 5

Un limbaj L este recursiv (decidabil) dacă există o maşină Turing M astfel încât $L(M) = L$ *şi M se opreşte pentru orice cuvânt* $w \in \Sigma^*$

- $accept(M) = L$, $reject(M) = \Sigma^* \setminus L$, $loop(M) = \emptyset$
- Clasa limbajelor recursive: $\mathcal{L}_r$
- $\mathcal{L}_r \subset \mathcal{L}_{re}$

Maşina Turing - model de calcul

- M poate fi utilizată pentru recunoaşterea limbajelor sau pentru calculul functiilor
- Model pentru calculul unei funcţii $f : \mathbb{N}^k \to \mathbb{N}$
  - în configuraţia iniţială, pe banda de intrare se memorează codificat argumentele $(n_1, n_2, \ldots, n_k)$: (codificare unară) $(0^{n_1}10^{n_2}1\ldots 10^{n^k})$
  - se aplică tranziţiile din configuraţia iniţială; daca maşina se opreşte şi banda conţine codificarea valorii $f(n_1, n_2, \ldots, n_k)$ $(0^{f(n_1,\ldots,n_k)})$, spunem că M calculează f
  - în cazul funcţiilor parţial definite, dacă f nu este definită în valorile $n_1, n_2, \ldots n_k$, M nu se opreşte
- Dacă există o maşină Turing care calculează f, atunci f se numeşte Turing-calculabilă.

Probleme de decizie

- Problemă algoritmică: o funcţie $P : \mathcal{I} \rightarrow \mathcal{F}$, $\mathcal{I}$ - mulţimea datelor de intrare (instanţelor problemei), $\mathcal{F}$ - mulţimea datelor finale, $\mathcal{I}, \mathcal{F}$ cel mult numărabile.
- $\mathcal{I}, \mathcal{F}$ cel mult numărabile, elementele lor pot fi reprezentate drept cuvinte peste un alfabet Σ
- Dacă $|\mathcal{F}| = 2$, $\mathcal{F} = \{true, false\}$, atunci P se numeşte problemă de decizie, altfel P se numeşte problemă computaţională
- Limbaj asociat unei probleme de decizie:
  $L_P = \{w \in \mathcal{I} | P(w) = true\}$

Probleme de decizie

- problemă computaţională:
  - ”Dat un cuvânt $w \in \Sigma^*$, să se obţină w^R”:
    $P : \Sigma^* \to \Sigma^*, P(w) = w^R$
- problemă de decizie
  - ”Să se decidă dacă numărul x este par”: $P : \mathbb{N} \to \{true, false\}$

    $$P(x) = true, \text{ dacă x este par}, P(x) = false, \text{ dacă x este impar}$$

  - $L_P = \{x | x \text{ este par }\}$

Probleme decidabile

Limbaj asociat unei probleme de decizie: $L_P = \{w \in \mathcal{I} | P(w) = \mathit{true}\}$

- Problema P este decidabilă dacă L_P este recursiv (există M cu $L(M) = L_P$, şi M se opreşte pentru toate cuvintele)
  - $\mathit{accept}(M) = L_P = \{w \in \mathcal{I} | P(w) = \mathit{true}\}$
  - $\mathit{reject}(M) = \{w \in \mathcal{I} | P(w) = \mathit{false}\}$
- Problema P este nedecidabilă dacă L_P nu este limbaj recursiv

Modele echivalente

- Maşini Turing nedeterministe
- Maşini Turing cu mai multe benzi
- Maşini Turing cu o bandă şi mai multe capete de citire a benzii
- Maşini Turing cu banda cu 2 dimensiuni, infinită la dreapta şi ”în jos”

- Puterea de calcul este aceeaşi: toate modelele pot fi simulate de o maşină deterministă cu o bandă

- Automate liniar mărginite - LBA (Linear Bounded Automata)
  - Maşini Turing ce au banda de intrare limitată la lungimea cuvântului de intrare
  - Clasa de limbaje acceptate: $\mathcal{L}_1$
- Teza lui Church $-$ Turing:
  - Orice funcţie efectiv calculabilă (calculabilă algoritmic) poate fi calculată cu o maşină Turing

sau

  - Nu există un model de calcul mai puternic decât maşina Turing