Turing Machines and Two-Pushdown Automata

Jiří Techet Tomáš Masopust Alexander Meduna

Department of Information Systems
Faculty of Information Technology
Brno University of Technology
Božetěchova 2, Brno 61266, Czech Republic

Modern Formal Language Theory, 2007

Turing Machines

Turing Machine

A Turing machine is a quintuple

$$M = (Q, \Sigma, R, s, F)$$

where

$$\begin{array}{rl}
Q & \text{is a finite set of states} \\
\Sigma & \text{is a tape alphabet, } \Sigma \cap Q = \emptyset, \\
 & I \subset \Sigma \text{ is an input alphabet,} \\
 & \sqcup \in \Sigma - I \text{ is the blank symbol} \\
R \subseteq Q\Sigma \times Q\Sigma & \text{is a finite set of rules,} \\
 & R = R_s \cup R_r \cup R_l \text{ (stationary, right, and left moves)} \\
s \in Q & \text{is the start state} \\
F \subseteq Q & \text{is a set of final states}
\end{array}$$

Turing Machines – Notation

Stationary move

$(qX, pY) \in R_s$ is symbolically written as

$$qX \rightarrow_s pY$$

Right move

$(qX, pY) \in R_r$ is symbolically written as

$$qX \rightarrow_r pY$$

Left move

$(qX, pY) \in R_l$ is symbolically written as

$$qX \rightarrow_l pY$$

Turing Machines – Computational Step

Configuration

$$\chi \in \Sigma^* Q \Sigma^* \{\sqcup\}$$

Move

If at least one of the following holds,

Stationary move $\chi = xpUy$, $\chi' = xqVy$, and $r : pU \rightarrow_s qV \in R$,

Right move $\chi = xpUy$, $\chi' = xVqy'$, and $r : pU \rightarrow_r qV \in R$, $y' = y$ if $y \neq \varepsilon$, and $y' = \sqcup$ if $y = \varepsilon$

Left move $\chi = xXpUy$, $\chi' = xqXVy$, and $r : pU \rightarrow_l qV \in R$, for some $X \in \Sigma$

then

$$\chi \Rightarrow \chi' \; [r]$$

Turing Machines – Accepted Language

Accepted Word

Turing machine M accepts $w \in I^*$ if

$$sw\sqcup \Rightarrow^* ufv$$

for some configuration ufv with $f \in F$

- $\Rightarrow^*$ denotes the reflexive and transitive closure of $\Rightarrow$

Accepted Language

The set of all words M accepts is the language of M, denoted by $L(M)$, thus

$$L(M) = \{w \in I^* : sw\sqcup \Rightarrow^* ufv, f \in F\}$$

Turing Machines – Example

Example

$$M = (\{q_0, q_1, q_2, q_3, q_4\}, \{a, b, \sqcup\}, R, q_0, \{q_4\})$$

where

$$\begin{array}{lll} R = \{1 : q_0 a \rightarrow_r q_1\sqcup, & 5 : q_1 \sqcup \rightarrow_l q_2\sqcup, & \\ \quad 2 : q_1 a \rightarrow_r q_1 a, & 6 : q_2 b \rightarrow_l q_3\sqcup, & \\ \quad 3 : q_1 b \rightarrow_r q_1 b, & 7 : q_3 a \rightarrow_l q_3 a, & \\ \quad 4 : q_3 \sqcup \rightarrow_r q_0\sqcup, & 8 : q_3 b \rightarrow_l q_3 b, & 9 : q_0 \sqcup \rightarrow_s q_4\sqcup\} \end{array}$$

$$\begin{array}{l} q_0 aabb\sqcup \Rightarrow \sqcup q_1 abb\sqcup\ [1] \Rightarrow \sqcup a q_1 bb\sqcup\ [2] \Rightarrow \sqcup ab q_1 b\sqcup\ [3] \\ \Rightarrow \sqcup abb q_1\sqcup\ [3] \Rightarrow \sqcup ab q_2 b\sqcup\ [5] \Rightarrow \sqcup a q_3 b\sqcup\sqcup\ [6] \Rightarrow \sqcup q_3 ab\sqcup\ [8] \\ \Rightarrow q_3\sqcup ab\sqcup\ [7] \Rightarrow \sqcup q_0 ab\sqcup\ [4] \Rightarrow \sqcup\sqcup q_1 b\sqcup\ [1] \Rightarrow \sqcup b q_1\sqcup\ [3] \\ \Rightarrow \sqcup q_2 b\sqcup\ [5] \Rightarrow q_3\sqcup\sqcup\sqcup\ [6] \Rightarrow \sqcup q_0\sqcup\ [4] \Rightarrow \sqcup q_4\sqcup\ [9] \end{array}$$

$$L(M) = \{a^n b^n : n \geq 0\}$$

Church's Thesis

Church's Thesis

For every algorithm that exits there is an equivalent Turing Machine.

Recursively Enumerable Language

A language L is recursively enumerable if there is a Turing machine M such that $L(M) = L$.

Recursive Language

A language L is recursive if there is a Turing machine M that always halts such that $L(M) = L$.

Deterministic Turing Machine

Deterministic Turing Machine

Turing machine M is deterministic if every rule $r \in R$ satisfies

$$\mathrm{lhs}(r) \notin \{\mathrm{lhs}(r') \,:\, r' \in R - \{r\}\}$$

Theorem

A language L is recursively enumerable if there is a deterministic Turing machine M such that $L(M) = L$.

Theorem

A language L is recursive if there is a deterministic Turing machine M that always halts such that $L(M) = L$.

Linear Bounded Automata

Linear Bounded Automaton

A linear bounded automaton is a Turing machine M that never extends its tape.

Consequence

With an input word w, M uses no more than the first $|w|$ tape squares.

Theorem

A language L is context-sensitive if and only if there is a linear bounded automaton M such that $L(M) = L$.

Open Problem

Are deterministic linear bounded automata as powerful as linear bounded automata?

Two-Pushdown Automata

Two-Pushdown Automaton

A two-pushdown automaton is a quintuple

$$M = (Q, \Sigma, R, s, F)$$

where

Q**,** s**,** F have the same meaning as in the definition of Turing machine

Σ is an alphabet, $\Sigma \cap Q = \emptyset$, $\Sigma = \{|\} \cup I \cup P_D$, where
$|$ is a special symbol, $| \notin I \cup P_D$,
I is an input alphabet, P_D is a pushdown alphabet, $S \in P_D$ is a start pushdown symbol

R is a finite set of rules of the form

$$A|Bpa \rightarrow u|vq$$

where $A, B \in P_D$, $p, q \in Q$, $a \in I \cup \{\varepsilon\}$, $u, v \in P_D^*$

Two-Pushdown Automata – Computational Step

Configuration

$$\chi \in P_D^* \{|\} P_D^* Q I^*$$

Move

If

$$r : A|Bpa \to u|vq \in R,$$

$$\chi = yA|xBpaz,$$

$$\chi' = yu|xvqz,$$

then

$$\chi \Rightarrow \chi' \ [r]$$

Two-Pushdown Automata – Accepted Language

Accepted Language by Final State

$$L_f(M) = \{w \in I^* : S|Ssw \Rightarrow^* x|yf, f \in F\}$$

Accepted Language by Empty Pushdown

$$L_e(M) = \{w \in I^* : S|Ssw \Rightarrow^* |q, q \in Q\}$$

Accepted Language by Final State and Empty Pushdown

$$L_{fe}(M) = \{w \in I^* : S|Ssw \Rightarrow^* |f, f \in F\}$$

- $\Rightarrow^*$ denotes the reflexive and transitive closure of $\Rightarrow$

Two-Pushdown Automata – Example

Example

$$M = (\{s, p, q, f\}, \{S, a, b, c, |\}, R, s, \{f\}),$$

where

$$\begin{aligned} R = \{1 : S|Ssa \to S|Sas, \quad & 4 : b|aqb \to bb|q, \\ 2 : S|asa \to S|aas, \quad & 5 : b|Sqc \to |Sp, \\ 3 : S|asb \to Sb|q, \quad & 6 : b|Spc \to |Sp, \quad 7 : S|Sp \to |f\} \end{aligned}$$

Then,

$$\begin{aligned} S|Ssaabbcc &\Rightarrow S|Sasabbcc\ [1] \Rightarrow S|Saasbbcc\ [2] \Rightarrow Sb|Saqbcc\ [3] \\ &\Rightarrow Sbb|Sqcc\ [4] \Rightarrow Sb|Spc\ [5] \Rightarrow S|Sp\ [6] \Rightarrow |f\ [7] \end{aligned}$$

$$L_f(M) = L_e(M) = L_{fe}(M) = \{a^n b^n c^n : n \geq 1\}$$

Two-Pushdown Automata – Results

Determinism

M is deterministic if each $r \in R$ with $\text{lhs}(r) = A|Bpq$ satisfies

$$\{r\} = \{r' \in R : A|Bpa = \text{lhs}(r') \text{ or } A|Bp = \text{lhs}(r')\}$$

Theorem

All acceptance modes (f, e, fe) are equivalent.

Theorem

The following models are equivalent:

- *Turing machines*
- *deterministic Turing machines*
- *two-pushdown automata*
- *deterministic two-pushdown automata*

Bibliography

S. Y. Kuroda.
Classes of languages and linear-bounded automata.
Information and Control, 7(2):207–223, 1964.

A. Meduna.
Automata and Languages: Theory and Applications.
Springer, London, 2000.

A. Turing.
On computable numbers with an application to the entscheidungs problem.
In *Proceedings of the London Mathematical Society*, volume 2, pages 230–265, 1936.
A correction, ibid, 544-546.