

Transducers and Translation Grammars

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

Finite Transducers

Finite Transducer

A **finite transducer** is a quintuple

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

where

Q is a finite set of **states**

Σ is an **alphabet**, $\Sigma \cap Q = \emptyset$, $\Sigma = I \cup O$,
 I and O are **input** and **output alphabets**

R is a finite set of **rules** of the form

$$pa \rightarrow qz$$

$$p, q \in Q, a \in I \cup \{\varepsilon\}, z \in O^*$$

$s \in Q$ is the **start state**

$F \subseteq Q$ is a set of **final states**

Input Finite Automaton

Input Finite Automaton

Let $M = (Q, \Sigma, R, s, F)$ be a finite transducer, then

$$M_I = (Q, I, R_I, s, F)$$

where

- I is the input alphabet of M and
- $R_I = \{qa \rightarrow p : qa \rightarrow px \in R, x \in O^*\}$

is the **input finite automaton**

Finite Transducers – Computational Step

Configuration

$$\chi \in QI^*\{\mid\}O^*$$

Move

If

$$r : qa \rightarrow pz \in R,$$

$$\chi = qa^{\color{red}w}|y,$$

$$\chi' = p^{\color{red}w}|yz,$$

then

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

Finite Transducers – Translation

Translation of a Word

M translates x into y if

$$sx| \Rightarrow^* f|y \text{ where } f \in F$$

Translation Defined by M

$$T(M) = \{(x, y) \in I^* \times O^* : sx| \Rightarrow^* f|y, f \in F\}$$

■ \Rightarrow^* denotes the reflexive and transitive closure of \Rightarrow

Finite Transducers – Input and Output Language

Input Language

$$L_I(M) = \{x \in I^* : (x, y) \in T(M) \text{ for some } y \in O^*\}$$

Output Language

$$L_O(M) = \{y \in O^* : (x, y) \in T(M) \text{ for some } x \in I^*\}$$

Theorem

Both input and output languages are regular.

Example

$$M = (\{s, q, f\}, \{!, a\}, R, s, \{f\})$$

where

$$R = \begin{array}{ll} 1 : s! \rightarrow q, & 3 : q! \rightarrow s, \\ 2 : sa \rightarrow fa, & 4 : qa \rightarrow f!a \end{array}$$

$$s!!!a \Rightarrow q!!a \mid [1] \Rightarrow s!a \mid [3] \Rightarrow qa \mid [1] \Rightarrow f!a \mid [4]$$

$$T(M) = \{(!^i a, a) : i \geq 0, i = 2k, k \geq 0\} \\ \cup \{(!^i a, !a) : i \geq 1, i = 2k + 1, k \geq 0\}$$

$$L_I(M) = \{!^i a : i \geq 0\} \quad L_O(M) = \{a, !a\}$$

Finite Transducers – Determinism

Deterministic Finite Transducer

M is **deterministic** if each rule $r \in R$ with $\text{lhs}(r) = pa$ satisfies

$$\{r\} = \{r' \in R : pa = \text{lhs}(r') \text{ or } p = \text{lhs}(r')\}$$

Example

Deterministic finite transducer

$$M = (\{f\}, \{0\}, \{f \rightarrow f0\}, f, \{f\}),$$

then

$$T(M) = \{(\varepsilon, 0^i) : i \geq 0\}$$

Pushdown Transducers

Pushdown Transducer

A **pushdown transducer** is a quintuple

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

where

Q, s, F have the same meaning as in the case of finite transducers

Σ is an **alphabet**, $\Sigma \cap Q = \emptyset$, $\Sigma = I \cup O \cup P_D$,

I, O, P_D are **input, output and pushdown alphabets**,

$S \in P_D$ is the **start pushdown symbol**

R is a finite set of **rules** of the form

$$A p a \rightarrow u q v$$

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

Input Pushdown Automaton

Input Pushdown Automaton

Let $M = (Q, \Sigma, R, s, F)$ be a pushdown transducer, then

$$M_I = (Q, I \cup P_D, R_I, s, F)$$

where

- I and P_D are the input and the pushdown alphabets of M
- $R_I = \{Aqa \rightarrow up : Aqa \rightarrow upv \in R, v \in O^*\}$

is the **input pushdown automaton**

Pushdown Transducers – Computational Step

Configuration

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

Move

If

$$r : Aqa \rightarrow upv \in R,$$

$$\chi = zAqaw|y,$$

$$\chi' = zupw|yv,$$

then

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

Pushdown Transducers – Translation

Translation of a Word

M translates x into y if

$$Ssx| \Rightarrow^* z\textcolor{red}{f}|y \text{ where } \textcolor{red}{f} \in \textcolor{red}{F}$$

Translation Defined by M

$$T(M) = \{(x, y) \in I^* \times O^* : Ssx| \Rightarrow^* zf|y, f \in F\}$$

■ \Rightarrow^* denotes the reflexive and transitive closure of \Rightarrow

Pushdown Transducers – Input and Output Language

Input Language

$$L_I(M) = \{x \in I^* : (x, y) \in T(M) \text{ for some } y \in O^*\}$$

Output Language

$$L_O(M) = \{y \in O^* : (x, y) \in T(M) \text{ for some } x \in I^*\}$$

Pushdown Transducers – Example

Example

$$M = (\{s, q, f\}, \{S, A, +, *, a\}, R, s, \{f\})$$

where

$$R = \{ \begin{array}{ll} 1 : Ss \rightarrow SAq, & 4 : Aq* \rightarrow *AAq, \\ 2 : Aqa \rightarrow qa, & 5 : +q \rightarrow q+, \\ 3 : Aq+ \rightarrow +AAq, & 6 : *q \rightarrow q*, \\ & 7 : Sq \rightarrow f \end{array} \}$$

$$\begin{aligned} Ss + *aaa &\Rightarrow SAq + *aaa \mid [1] \Rightarrow S + AAq * aaa \mid [3] \\ &\Rightarrow S + A * AAqaaa \mid [4] \Rightarrow S + A * Aqaa \mid a [2] \Rightarrow S + A * qa \mid aa [2] \\ &\Rightarrow S + Aqa \mid aa * [6] \Rightarrow S + q \mid aa * a [2] \Rightarrow Sq \mid aa * a + [5] \\ &\Rightarrow f \mid aa * a + [7] \end{aligned}$$

$$T(M) = \{(pre, post) : pre = \text{prefix expression}, post = \text{postfix expression}\}$$

Pushdown Transducers – Determinism

Deterministic Pushdown Transducer

M is **deterministic** if each rule $r \in R$ with $\text{lhs}(r) = Apa$ satisfies

$$\{r\} = \{r' \in R : Apa = \text{lhs}(r') \text{ or } Ap = \text{lhs}(r')\}$$

Extended Pushdown Transducer

$M = (Q, \Sigma, R, s, F)$ is **extended** if R is a finite set of productions of the form

$$zpa \rightarrow uqv$$

$$z \in P_D^*, p, q \in Q, a \in I \cup \{\varepsilon\}, u \in P_D^*, v \in O^*$$

Translation Grammar

A **translation grammar** is a quadruple

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

where

N, S are defined as usual, $S \in N$

T is a terminal alphabet, $T \cap N = \emptyset$, $T = I \cup O$

I and O are **input** and **output** alphabets

P is a finite set of productions of the form

$$A \rightarrow u_0 B_1 u_1 \dots B_n u_n \mid v_0 B_1 v_1 \dots B_n v_n$$

\mid is a **special symbol**, $B_i \in N$, $u_j \in I^*$, $v_j \in O^*$,

$i = 1, \dots, n$, $j = 0, \dots, n$

Notation

For

$$p = A \rightarrow u_0 B_1 u_1 \dots B_n u_n \mid v_0 B_1 v_1 \dots B_n v_n,$$

- $\text{lhs}(p) = A$
- $\text{irhs}(p) = u_0 B_1 u_1 \dots B_n u_n$
- $\text{orhs}(p) = v_0 B_1 v_1 \dots B_n v_n$

Direct Derivation

For $G = (N, T, P, S)$, $p \in P, x, y, u, v \in (N \cup T)^*$, then

$$x \text{lhs}(p) y \mid u \text{lhs}(p) v \Rightarrow x \text{irhs}(p) y \mid u \text{orhs}(p) v$$

Translation Grammars – Translation

Translation Defined by G

$$T(G) = \{u|v : S|S \Rightarrow^* u|v, u \in I^*, v \in O^*\}$$

Input Grammar

$$G_I = (N, I, P_I, S)$$

where $P_I = \{A \rightarrow x : A \twoheadrightarrow x|y \in P\}$

Output Grammar

$$G_O = (N, O, P_O, S)$$

where $P_O = \{A \rightarrow y : A \twoheadrightarrow x|y \in P\}$

Translation Grammars – Example

Example

Let the translation grammar G be defined by the following productions:

$$\begin{aligned}\langle expr \rangle &\rightarrow \langle expr \rangle + \langle term \rangle \mid \langle expr \rangle \langle term \rangle + \\ \langle expr \rangle &\rightarrow \langle term \rangle \mid \langle term \rangle \\ \langle term \rangle &\rightarrow \langle term \rangle * \langle factor \rangle \mid \langle term \rangle \langle factor \rangle * \\ \langle term \rangle &\rightarrow \langle factor \rangle \mid \langle factor \rangle \\ \langle factor \rangle &\rightarrow (\langle expr \rangle) \mid \langle expr \rangle \\ \langle factor \rangle &\rightarrow a \mid a\end{aligned}$$

$$\langle expr \rangle \mid \langle expr \rangle \Rightarrow^* (a + a) * a \mid aa + a *$$

G translates an infix expression with $+$ and $*$ to the corresponding postfix expression.



J. Evey.

Application of pushdown store machines.

In *Proceedings 1963 Fall Joint Computer Conference*, pages 215–227, Montvale, NJ: AFIPS Press, 1963.



A. Meduna.

Automata and Languages: Theory and Applications.

Springer, London, 2000.



J. Sheperdson.

The reduction of two-way automata to one-way automata.

IBM Journal of Research and Development, 3:198–200, 1959.