REGULATED PUSHDOWN AUTOMATA REVISITED

Lukáš Rychnovský
Doctoral Degree Programme (3), FIT BUT
E-mail: rychnov@fit.vutbr.cz

Supervised by: Alexander Meduna
E-mail: meduna@fit.vutbr.cz

ABSTRACT

This paper demonstrates the alternative proof for the theorem of equivalence between regulated pushdown automata and recursive enumerable languages as shown in [Med–00].

1 INTRODUCTION

When developing some applications of formal languages it is necessary to understand proofs in their construct way. For this purpose, we demonstrate constructive alternative of the proof that a regulated pushdown automaton is equivalent to a Turing machine from [Med–00].

2 DEFINITIONS

Definition 2.1. *A* extended pushdown automaton *(PDA for short) is a rewriting system, usually noted as a 7-tuple $T = (Q, \Sigma, \Omega, \delta, s, \nabla, F)$, where Q is a finite set of states, Σ is a finite set of the input alphabet, Ω is a finite set of the stack alphabet, δ is a finite transition relation $((\Sigma \cup \{\varepsilon\}) \times Q \times \Omega) \to Q \times \Omega^*$, $s \in Q$ is the start state, $\nabla \in \Omega$ is the initial stack symbol and $F \subseteq Q$ is a set of final states.*

A configuration *of the pushdown automaton is a triple (q, w, γ), where $q \in Q$ is the current state, $w \in \Sigma^*$ are non read characters and $\gamma \in \Omega^*$ are symbols on the stack.*

A computational step *of pushdown automaton is a binary relation $\vdash_T$ (or simply $\vdash$ if no confusion can arise) defined as*

$$(q_1, aw, Z\gamma) \vdash_T (q_2, w, Y\gamma) \Leftrightarrow \delta(q_1, a, Z) = (q_2, Y).$$

In the previously defined manner, we extend $\vdash$ to $\vdash^n$, where $n \geq 0$, $\vdash^+$ and $\vdash^*$.

Let $T = (Q, \Sigma, \Omega, \delta, s, \nabla, F)$.
The language accepted by pushdown automaton T by final state *is*

$$\mathcal{L}(T) = \{w \mid w \in \Sigma^*, (s, w, \nabla) \vdash_T^* (q_F, \varepsilon, \gamma), q_F \in F, \gamma \in \Omega^*\}$$

The language accepted by pushdown automaton T by empty pushdown *is*

$$\mathcal{L}(T) = \{w \mid w \in \Sigma^*, (s, w, \nabla) \vdash_T^* (q, \varepsilon, \varepsilon), q \in Q\}$$

Definition 2.2. *Let $M = (Q, \Sigma, \Omega, \delta, s, \nabla, F)$ be a PDA and let $x, x', x'' \in \Omega^*, y, y', y'' \in \Sigma^*$, q, q', $q'' \in Q$, and $\nabla xqy \vdash \nabla x'q'y' \vdash \nabla x''q''y''$. If $|x| \leq |x'|$ and $|x'| > |x''|$, then $\nabla x'q'y' \vdash \nabla x''q''y''$ is a* turn. *If M makes no more than one turn during any sequence of moves starting from an initial configuration, then M is said to be* one-turn *(OTSA).*

Definition 2.3. *Let $G = (V, P)$ be a rewriting system. Let Ψ be an alphabet of* rule labels *such that card(Ψ) = card(P), and ψ be a bijection from P to Ψ. For simplicity, to express that ψ maps a rule, $u \to v \in P$, to ρ, where $\rho \in \Psi$, we write $\rho.u \to v \in P$; in other words, $\rho.u \to v$ means $\psi(u \to v) = \rho$.*

If $u \to v \in P$ and $x, y \in V^*$, then $xuy \Rightarrow xvy$ $[u \to v]$ or simply $xuy \Rightarrow xvy$ $[\rho]$. Let there exists a sequence $x_0, x_1, \ldots, x_n \in V^*$ for some $n \geq 1$ such that $x_{i-1} \Rightarrow x_i$ $[\rho_i]$, where $\rho_i \in \Psi$, for $i = 1, \ldots, n$. Then G rewrites x_0 to x_n in n steps according to $\rho_1, \ldots, \rho_n$, symbolically written as $x_0 \Rightarrow^n x_n$ $[\rho_1 \ldots \rho_n]$.

Let Ξ be a control language *over Ψ; that is $\Xi \in \Psi^*$.*

Definition 2.4. *Let $T = (Q, \Sigma, \Omega, \delta, s, \nabla, F)$ be a PDA and let Ψ be an alphabet of rule labels and let Ξ be a control language. A* language generated by pushdown automaton *T* regulated by control language Ξ *is*

$$\mathcal{L}(T, \Xi) = \{w \mid w \in \Sigma^*, (s, w, \nabla) \vdash_T^n (q_F, \varepsilon, \gamma)\ [\rho_1 \ldots \rho_n], \rho_1, \ldots, \rho_n \in \Xi, q_F \in F, \gamma \in \Omega^*\}.$$

If it is useful to distinguish, T defines the following types of accepted languages:

1. *$\mathcal{L}(T, \Xi, 1) = \mathcal{L}(T, \Xi)$ – the language accepted by the final state.*
2. *$\mathcal{L}(T, \Xi, 2)$ – the language accepted by an empty pushdown.*
3. *$\mathcal{L}(T, \Xi, 3)$ – the language accepted by the final state and an empty pushdown.*

Definition 2.5. *A type-0 grammar $G = (N, T, P, S)$ is in* Penttonen normal form *if every production $p \in P$ has one of these forms*

1. *$CB \to CD$*
2. *$D \to BC$*
3. *$C \to c$*
4. *$C \to \varepsilon$*

3 RESULT

Theorem 3.1. *Any recursive enumerable language L can be generated as $L = \mathcal{L}(M, L_1, 3)$ where M is an OTSA and L_1 is a linear language.*

Proof. Let L be any recursive enumerable language so that $L = \mathcal{L}(G)$ where $G = (N, T, P, S)$ is type-0 grammar in Penttonen normal form. Let $M = (Q, \Sigma, \Omega, \delta, s, \nabla, F)$ be an OTSA, where

1. $Q = \{q, q_{in}, q_{out}\}$,
2. $\Sigma = T$,
3. $\Omega = T \cup N \cup \{\#\} \cup \{\nabla\}$, where $\# \notin \{N \cup T\}$,
4. $s = q$,

5. $\nabla \in \Omega$ is the initial stack symbol

6. $F = \{q_{out}\}$.

7. $\delta = \delta' \cup \delta_{in} \cup \delta_{out}$, where
$\delta' = \{\langle a\rangle.aq \to qa \mid \text{for every } a \in T\} \cup \{\langle\#\rangle.q \to q_{in}\#\}$,
$\delta_{in} = \{\langle A\rangle.q_{in} \to q_{in}A \mid \text{for every } A \in T \cup N \cup \{\#\}\} \cup \{\langle 2\rangle.q_{in} \to q_{out}\}$,
$\delta_{out} = \{\langle\overline{A}\rangle.q_{out}A \to q_{out} \mid \text{for every } A \in T \cup N \cup \{\#\}\}$.

A control language L_1, which is linear, is defined by the following grammar $G_1 = (N_1, T_1, P_1, S_1)$:

1. $N_1 = \{S_1, K, M, M', O\}$,

2. $T_1 = \{\langle A\rangle, \langle\overline{A}\rangle \mid A \in T \cup N \cup \{\#\}$ and $\langle A\rangle$ is label from $\Psi\} \cup \{\langle 2\rangle\}$,

3. $P_1 = P_a \cup P_{\langle\#\rangle} \cup P_b \cup P_c \cup P_d \cup P_{\overline{c}} \cup P_{\overline{d}} \cup P_e \cup P_f \cup P_g \cup P_h \cup P_{\langle 2\rangle}$, where
$P_a = \{S_1 \to \langle a\rangle S_1 \mid \text{for every } a \in T\}$,
$P_{\langle\#\rangle} = \{S_1 \to \langle\#\rangle K\}$,
$P_b = \{K \to \langle A\rangle K\langle\overline{A}\rangle \mid \text{for every } A \in T \cup N\}$,
$P_c = \{K \to \langle C\rangle M\langle\overline{C}\rangle \mid \text{for every rule in the form } CB \to CD \in P\}$,
$P_d = \{M \to \langle B\rangle O\langle\overline{D}\rangle \mid \text{for every rule in the form } CB \to CD \in P\}$,
$P_{\overline{c}} = \{K \to \langle D\rangle M'\langle\overline{C}\rangle \mid \text{for every rule in the form } D \to BC \in P\}$,
$P_{\overline{d}} = \{M' \to O\langle\overline{B}\rangle \mid \text{for every rule in the form } D \to BC \in P\}$,
$P_e = \{K \to \langle C\rangle O\langle\overline{c}\rangle \mid \text{for every rule in the form } C \to c \in P\}$,
$P_f = \{K \to \langle C\rangle O \mid \text{for every rule in the form } C \to \varepsilon \in P\}$,
$P_g = \{O \to \langle A\rangle O\langle\overline{A}\rangle \mid \text{for every } A \in T \cup N\}$,
$P_h = \{O \to \langle\#\rangle K\langle\overline{\#}\rangle\}$,
$P_{\langle 2\rangle} = \{K \to \langle 2\rangle\langle\overline{\#}\rangle\langle\overline{S}\rangle\}$.

Now, we prove two standard inclusions. First, $L \subseteq \mathcal{L}(M, L_1)$. For every $w \in L$ there exists some successful derivation $S = w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_n = w$ in L. We will construct the control string R as follows (for the sake of simplicity we omit $\langle$ and $\rangle$ if no confusion can arise)

$$R = w\#w_{n-1}\#\ldots\#w_1\#S\#\langle 2\rangle\overline{\#S\#w_1^R\#\ldots\#w_{n-1}^R\#w^R}.$$

It is easy to verify, that OTSA M under regulation of R reaches the final state and empties its pushdown (because $R = R'\langle 2\rangle\overline{rev(R')}$).

We need to prove that $R \in \mathcal{L}(G_1)$. For every R_i:

$$R_0 = w\#\,K$$

$$R_1 = w\#w_{n-1}\#\,K\,\overline{\#w^R}.$$

$$\vdots$$

$$R_m = w\#w_{n-1}\#\ldots\#w_{n-m}\,K\,\overline{\#w_{n-(m-1)}^R\#\ldots\#w_{n-1}^R\#w^R}.$$

holds $S_1 \Rightarrow^* R_i$ by induction on i.
$i = 0$: $S_1 \Rightarrow^{|w|} wS_1 \Rightarrow w\#\,K$, hence $w\#K \in \mathcal{L}(G_1)$.
$i = k$:

$$R_k = w\#w_{n-1}\#\ldots\#w_{n-k}\,K\,\overline{\#w_{n-(k-1)}^R\#\ldots\#w_{n-1}^R\#w^R}.$$

That is, $K \Rightarrow^* w_{n-(k+1)}\ O\ \overline{w_{n-k}^R} \Rightarrow w_{n-(k+1)}\#\ K\ \overline{\#w_{n-k}^R}$ by using rules from P_b to elements not affected in the rewriting of w_{n-k} to w_{n-k+1}. Then one or two rules from sets $P_c, P_d, P_{\overline{c}}, P_{\overline{d}}, P_e$ and P_f are used according to used rule from P. The rest rules are taken from P_g and finally one rule from P_h rewrites nonterminal O to K.

$$R_k \Rightarrow^* w\#w_{n-1}\#\ldots\#w_{n-k}\#w_{n-(k+1)}\#\ K\ \overline{\#w_{n-k}^R\#w_{n-(k-1)}^R\#\ldots\#w_{n-1}^R\#w^R} = R_{k+1}.$$

Let us see a short example. For the sake of simplicity we again omit $\langle$ and $\rangle$ if no confusion can arise. The derivation $S \Rightarrow AX \Rightarrow ABC \Rightarrow aBC \Rightarrow aDC \Rightarrow aDc \Rightarrow abc$ in grammar $G = (\{S,A,B,C,X\}, \{a,b,c\}, S, \{S \to AX, X \to BC, BC \to DC, A \to a, D \to b, C \to c\})$ results in $abc\#aDc\#aDC\#aBC\#ABC\#AX\#S\#\langle 2\rangle\overline{\#S\#XA\#CBA\#CBa\#CDa\#cDa\#cba}$ as the control string.

The underlying OTSA under such derivation string operates as follows:
$a.aq \to qa : (abc, q, \nabla) \vdash (bc, q, a)$
$b.bq \to qb : (bc, q, a) \vdash (c, q, ab)$
$c.cq \to qc : (c, q, ab) \vdash (\varepsilon, q, abc)$
$\#.q \to q_{in}\# : (\varepsilon, q, abc) \vdash (\varepsilon, q_{in}, abc\#)$
$a.q_{in} \to q_{in}a : (\varepsilon, q_{in}, abc\#) \vdash (\varepsilon, q_{in}, abc\#a)$
$D.q_{in} \to q_{in}D : (\varepsilon, q_{in}, abc\#a) \vdash (\varepsilon, q_{in}, abc\#aD)$
$c.q_{in} \to q_{in}c : (\varepsilon, q_{in}, abc\#aD) \vdash (\varepsilon, q_{in}, abc\#aDc)$
$\#.q_{in} \to q_{in}\# : (\varepsilon, q_{in}, abc\#aDc) \vdash (\varepsilon, q_{in}, abc\#aDc\#)$
$\vdots$
$S.q_{in} \to q_{in}S : (\varepsilon, q_{in}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#) \vdash$
$\vdash (\varepsilon, q_{in}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#S)$
$\#.q_{in} \to q_{in}\# : (\varepsilon, q_{in}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#S) \vdash$
$\vdash (\varepsilon, q_{in}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#S\#)$
$\langle 2\rangle.q_{in} \to q_{out}\# : (\varepsilon, q_{in}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#S\#) \vdash$
$\vdash (\varepsilon, q_{out}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#S\#)$
$\overline{\#}.q_{out}\# \to q_{out} : (\varepsilon, q_{out}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#S\#) \vdash$
$\vdash (\varepsilon, q_{out}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#S)$
$\overline{S}.q_{out}S \to q_{out} : (\varepsilon, q_{out}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#S) \vdash$
$\vdash (\varepsilon, q_{out}, abc\#aDc\#aDC\#aBC\#ABC\#AX\#)$
$\vdots$
$\overline{\#}.q_{out}\# \to q_{out} : (\varepsilon, q_{out}, abc\#) \vdash (\varepsilon, q_{out}, abc)$
$\overline{c}.q_{out}c \to q_{out} : (\varepsilon, q_{out}, abc) \vdash (\varepsilon, q_{out}, ab)$
$\overline{b}.q_{out}b \to q_{out} : (\varepsilon, q_{out}, ab) \vdash (\varepsilon, q_{out}, a)$
$\overline{a}.q_{out}a \to q_{out} : (\varepsilon, q_{out}, a) \vdash (\varepsilon, q_{out}, \nabla)$
so OTSA is in final state and has empty stack.

The derivation of control string in control language is

$$S_1 \Rightarrow aS_1 \Rightarrow abS_1 \Rightarrow abcS_1 \Rightarrow abc\#K \Rightarrow abc\#aK\overline{a} \stackrel{D\to b}{\Longrightarrow} abc\#aD\ O\ \overline{ba} \Rightarrow$$

$$\Rightarrow abc\#aDc\ O\ \overline{cba} \Rightarrow abc\#aDc\#\ K\ \overline{\#cba} \Rightarrow \ldots \Rightarrow$$

$$\Rightarrow abc\#aDc\#aDC\#aBC\#ABC\#AX\#S\#\ K\ \overline{\#XA\#CBA\#CBa\#CDa\#cDa\#cba} \Rightarrow$$

$$\Rightarrow abc\#aDc\#aDC\#aBC\#ABC\#AX\#S\#\langle 2\rangle\overline{\#S\#XA\#CBA\#CBa\#CDa\#cDa\#cba}.$$

The second inclusion is $\mathcal{L}(M, L_1) \subseteq L$. Let us suppose that the word $w_m = x_1 x_2 \ldots x_p \in \mathcal{L}(M, L_1)$. We will prove the following theorem by induction on n:
For any integer n, the word $w_{m-n}, 1 \leq n \leq m$ in the control string R_n

$$R_0 = w_m\# \; K$$

$$R_1 = w_m\# w_{m-1}\# \; K \; \overline{\# w_m^R}$$

$$\vdots$$

$$R_n = w_m\# w_{m-1}\# \ldots \# w_{m-n}\# \; K \; \overline{\# w_{m-(n-1)}^R \# \ldots \# w_{m-1}^R \# w_m^R}$$

can be derived from w_{m-n} to w_m in n steps in G, hence $w_{m-n} \Rightarrow^n w_m$ in G.
$n = 0$: $w_m \Rightarrow^0 w_m$.
$n = k$:

$$R_k = w_m \ldots \# w_{m-k}\# \; K \; \overline{\# w_{m-(k-1)}^R \# \ldots \# w_m^R}$$

$w_{m-k} = y_1 y_2 \ldots y_q$. As M is OTSA, the sequence of pushed symbols onto the stack will be popped in reverse order. Hence,

$$K \Rightarrow^* z_1 z_2 \ldots z_r \# \; K \; \overline{\# y_q \ldots y_2 y_1}$$

where $z_i \in (N \cup T)$ and there exists index i such as $y_1 = z_1, \ldots, y_i = z_i$ and

$$K \Rightarrow^i y_1 y_2 \ldots y_i \; K \; \overline{y_i \ldots y_2 y_1},$$

according to i applications of rules from P_b. Now there are 4 possible rules to apply $P_c, P_{\overline{c}}, P_e$, and P_f. The next step has to generate $\overline{y_{i+1}}$ on the right side of K.

1. P_c: $K \Rightarrow C \; M \; \overline{C} \Rightarrow CB \; O \; \overline{DC} \Leftrightarrow CB \rightarrow CD \in P$ and $y_{i+2} = D$ and $y_{i+1} = C$.
2. $P_{\overline{c}}$: $K \Rightarrow D \; M' \; \overline{C} \Rightarrow D \; O \; \overline{BC} \Leftrightarrow D \rightarrow BC \in P$ and $y_{i+2} = B$ and $y_{i+1} = C$.
3. P_e: $K \Rightarrow C \; O \; \overline{c} \Leftrightarrow C \rightarrow c \in P$ and $y_{i+1} = c$.
4. P_f: $K \Rightarrow C \; O \; \Leftrightarrow C \rightarrow \varepsilon \in P$.

Now there are two possible rules to apply. From P_g and P_h. As there are still some elements of y_k on the right side of O, we have to use rules from P_g until there is complete $\overline{y_q \ldots y_2 y_1}$ generated on the right side of O. Consequently, there exists index j such that $y_j = z_k, \ldots, y_q = z_r$. Then, the last rule from P_h generates # and $\overline{\#}$ on both sides of O and O rewrites to K. Then,

$$R_k \Rightarrow^* w_m\# \ldots \# w_{m-k}\# w_{m-(k+1)} \; K \; \overline{\# w_{m-k}^R \# w_{m-(k-1)}^R \# \ldots \# w_m^R} = R_{k+1}$$

and $w_{m-k} \Rightarrow w_{m-(k+1)}$ in G.

So, the complete control string will be

$$R = w_n\# w_{n-1}\# \ldots \# w_1 \# S \# \langle 2 \rangle \overline{\# S \# w_1^R \# \ldots \# w_{n-1}^R \# w_n^R}$$

and there exists the derivation $S \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_{n-1} \Rightarrow w_n$ in G and $w_n \in L = \mathcal{L}(G)$. □

REFERENCES

[Med–00] Meduna, A., Kolář, D.: Regulated Pushdown Automata, *Acta Cybernetica*, Vol. 14, 2000. 653–664.