Fundamenta Informaticae XXI (2001) 1001–1007 1001
IOS Press

One-Turn Regulated Pushdown Automata and Their Reduction

Alexander Meduna
Brno University of Technology
Department of Computer Science and Engineering
Božetěchova 2, Brno 612 66, Czech Republic
meduna@dcse.fee.vutbr.cz

Dušan Kolář
Brno University of Technology
Department of Computer Science and Engineering
Božetěchova 2, Brno 612 66, Czech Republic
kolar@dcse.fee.vutbr.cz

Abstract. This paper discusses some simple and natural restrictions of regulated pushdown automata. Most importantly, it studies one-turn regulated pushdown automata and proves that they characterize the family of recursively enumerable languages. In fact, this characterization holds even for atomic one-turn regulated pushdown automata of a reduced size. This result is established in terms of acceptance by final state and empty pushdown, acceptance by final state, and acceptance by empty pushdown.

Keywords: one-turn regulated pushdown automata, reduced and atomic variants, recursively enumerable languages

1. Introduction

Regulated grammars play an important role in the language theory. Recently, this theory has introduced their machine-based counterpart—pushdown automata regulated by linear languages or, more simply, regulated pushdown automata, which characterize the family of recursively enumerable languages (see [5]). The present paper continues with the discussion of these automata. More specifically, it studies *one-turn regulated pushdown automata*.

To recall the concept of one-turn pushdown automata (see [2]), consider two consecutive moves made by a pushdown automaton, M. If during the first move M does not shorten its pushdown and during the second move it does, then M makes a turn during the second move. A pushdown automaton is *one-turn* if it makes no more than one turn with either of its pushdowns during any computation starting from an initial configuration. Recall that the one-turn pushdown automata characterize the family of linear languages (see [2]) while their unrestricted versions characterize the family of context-free languages. As a result, the one-turn pushdown automata are less powerfull than the pushdown automata.

The present paper demonstrates that one-turn regulated pushdown automata characterize the family of recursively enumerable languages. Thus, as opposed to the ordinary one-turn pushdown automata, the one-turn regulated pushdown automata are as powerfull as the regulated pushdown automata that can make any number of turns. In fact, this equivalence holds even for some restricted versions of one-turn regulated pushdown automata, including their atomic and reduced versions, which are sketched next.

I. During a move, an *atomic* one-turn regulated pushdown automaton changes a state and, in addition, performs exactly one of the following actions:

1. it pushes a symbol onto the pushdown
2. it pops a symbol from the pushdown
3. it reads an input symbol

II. A *reduced* one-turn regulated pushdown automaton has a limited number of some components, such as the number of states, pushdown symbols or transition rules.

The present paper proves that every recursively enumerable language is accepted by an atomic reduced one-turn regulated pushdown automaton in terms of (A) acceptance by final state and empty pushdown, (B) acceptance by final state, and (C) acceptance by empty pushdown.

2. Preliminaries

We assume that the reader is familiar with the language theory (see [4]).

For a set, X, $card(X)$ denotes its cardinality.

Let V be an alphabet. V^* represents the free monoid generated by V under the operation of concatenation. The unit of V^* is denoted by ε. Set $V^+ = V^* - \{\varepsilon\}$; algebraically, V^+ is thus the free semigroup generated by V under the operation of concatenation.

For $w \in V^*$, $|w|$ and $rev(w)$ denote the length of w and the reversal of w, respectively. Set $prefix(w) = \{x \mid x \text{ is a } prefix\ of\ w\}$, $suffix(w) = \{x \mid x \text{ is a } suffix\ of\ w\}$, and $alph(w) = \{a \mid a \in V, \text{and } a \text{ appears in } w\}$.

For $w \in V^+$ and $i \in \{1, \dots, |w|\}$, $sym(w, i)$ denotes the ith symbol of w; for instance, $sym(abcd, 3) = c$.

A *linear grammar* is a quadruple, $G = (N, T, P, S)$, where N and T are alphabets such that $N \cap T = \emptyset$, $S \in N$, and P is a finite set of productions of the form $A \to x$, where $A \in N$ and $x \in T^*(N \cup \{\varepsilon\})T^*$. If $A \to x \in P$ and $u, v \in T^*$, then $uAv \Rightarrow uxv\ [A \to x]$ or, simply, $uAv \Rightarrow uxv$. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$; then, based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of G, $L(G)$, is defined as $L(G) = \{w \in T^* \mid S \Rightarrow^* w\}$. A language, L, is *linear* if and only if $L = L(G)$, where G is a linear grammar.

A *queue grammar* (see [3]) is a sixtuple, $Q = (V, T, W, F, S, P)$, where V and W are alphabets satisfying $V \cap W = \emptyset, T \subseteq V, F \subseteq W, S \in (V - T)(W - F)$, and $P \subseteq (V \times (W - F)) \times (V^* \times W)$ is a finite relation such that for every $a \in V$, there exists an element $(a, b, x, c) \in P$. If $u, v \in V^*W$ such that $u = arb$, $v = rzc$, $a \in V$, $r, z \in V^*$, $b, c \in W$ and $(a, b, z, c) \in P$, then $u \Rightarrow v$ $[(a, b, z, c)]$ in G or, simply, $u \Rightarrow v$. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$. Based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of Q, $L(Q)$, is defined as $L(Q) = \{w \in T^* \mid S \Rightarrow^* wf \text{ where } f \in F\}$.

A *left-extended queue grammar* (see [5]) is a sixtuple, $Q = (V, T, W, F, S, P)$, where $V, T, W, F, S,$ and P have the same meaning as in a queue grammar; in addition, assume that $\# \notin V \cup W$. If $u, v \in V^*\{\#\}V^*W$ so $u = w\#arb$, $v = wa\#rzc$, $a \in V$, $r, z, w \in V^*$, $b, c \in W$, and $(a, b, z, c) \in P$, then $u \Rightarrow v$ $[(a, b, z, c)]$ in G or, simply, $u \Rightarrow v$. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$. Based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$ as usual. The language of $Q, L(Q)$, is defined as $L(Q) = \{v \in T^* \mid \#S \Rightarrow^* w\#vf \text{ for some } w \in V^* \text{ and } f \in F\}$.

3. Definitions

This section defines the notion of a one-turn atomic pushdown automaton regulated by a linear language.

Informally, an atomic pushdown automaton changes a state and, in addition, makes only one of these three actions:

(a) pushing a symbol onto the pushdown;

(b) popping a symbol from the pushdown;

(c) reading a symbol on the input tape.

Formally, an *atomic pushdown automaton* is a 7-tuple, $M = (Q, \Sigma, \Omega, R, s, \$, F)$, where Q is a finite set of states, Σ is an input alphabet, Ω is a pushdown alphabet (Q, Σ, and Ω are pairwise disjoint), $s \in Q$ is the start state, \$ is the pushdown-bottom marker, $\$ \notin Q \cup \Sigma \cup \Omega$, $F \subseteq Q$ is a set of final states, R is a finite set of rules of the form $Apa \rightarrow wq$, where $p, q \in Q$, $A, w \in \Omega \cup \{\varepsilon\}$, $a \in \Sigma \cup \{\varepsilon\}$, such that $|Aaw| = 1$. That is, R is a finite set of rules such that each of them has one of these forms

(1) $Ap \rightarrow q$ (*popping rule*)

(2) $p \rightarrow wq$ (*pushing rule*)

(3) $pa \rightarrow q$ (*reading rule*)

Let Ψ be an alphabet of *rule labels* such that $card(\Psi) = card(R)$, and ψ be a bijection from R to Ψ. For simplicity, to express that ψ maps a rule, $Apa \rightarrow wq \in R$, to ρ, where $\rho \in \Psi$, this paper writes $\rho.Apa \rightarrow wq \in R$; in other words, $\rho.Apa \rightarrow wq$ means $\psi(Apa \rightarrow wq) = \rho$. A *configuration* of M, χ, is any word from $\{\$\}\Omega^*Q\Sigma^*$; χ is an initial configuration if $\chi = \$sw$, where $w \in \Sigma^*$. For every $x \in \Omega^*$, $y \in \Sigma^*$, and $\rho.Apa \rightarrow wq \in R$, M makes a move from configuration $\$xApay$ to configuration $\$xwqy$ according to ρ, written as $\$xApay \Rightarrow \$xwqy$ $[\rho]$ or, more simply, $\$xApay \Rightarrow \$xwqy$. Let χ be any configuration of M. M makes *zero moves from* χ *to* χ *according to* ε, symbolically written as $\chi \Rightarrow^0 \chi$ $[\varepsilon]$. Let there exist a sequence of configurations $\chi_0, \chi_1, \ldots, \chi_n$ for some $n \geq 1$ such that $\chi_{i-1} \Rightarrow \chi_i$ $[\rho_i]$, where $\rho_i \in \Psi$, for $i = 1, \ldots, n$, then M makes n *moves from* χ_0 *to* χ_n *according to*

$\rho_1 \dots \rho_n$, symbolically written as $\chi_0 \Rightarrow^n \chi_n \, [\rho_1 \dots \rho_n]$ or, more simply, $\chi_0 \Rightarrow^n \chi_n$. Define $\Rightarrow^*$ and $\Rightarrow^+$ in the standard manner.

Let $x, x', x'' \in \Omega^*$, $y, y', y'' \in \Sigma^*$, $q, q', q'' \in Q$, and $\$xqy \Rightarrow \$x'q'y' \Rightarrow \$x''q''y''$. If $|x| \leq |x'|$ and $|x'| > |x''|$, then $\$x'q'y' \Rightarrow \$x''q''y''$ 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*.

Let Ξ be a *control language* over Ψ; that is, $\Xi \subseteq \Psi^*$. With Ξ, M defines the following three types of accepted languages:

$L(M, \Xi, 1)$—*the language accepted by final state*

$L(M, \Xi, 2)$—*the language accepted by empty pushdown*

$L(M, \Xi, 3)$—*the language accepted by final state and empty pushdown*

defined as follows. Let $\chi \in \{\$\}\Omega^* Q \Sigma^*$. If $\chi \in \{\$\}\Omega^* F$, $\chi \in \{\$\}Q$, $\chi \in \{\$\}F$, then χ is a *1-final configuration*, *2-final configuration*, *3-final configuration*, respectively. For $i = 1, 2, 3$, define $L(M, \Xi, i)$ as $L(M, \Xi, i) = \{w \mid w \in \Sigma^*, \text{ and } \$sw \Rightarrow^* \chi \, [\sigma] \text{ in } M \text{ for an } i\text{−final configuration}, \ \chi, \text{ and } \sigma \in \Xi\}$.

For any family of languages, X, and $i \in \{1, 2, 3\}$, set $\mathcal{L}(X, i) = \{L \mid L = L(M, \Xi, i)$, where M is a pushdown automaton and $\Xi \in X\}$. RE and LIN denote the families of recursively enumerable and linear languages, respectively.

4. Results

This section proves that the one-turn atomic pushdown automata regulated by linear languages characterize RE. In fact, these automata need no more than one state and two pushdown symbols to achieve this characterization.

Theorem 4.1. For every left-extended queue grammar, K, there exists a left-extended queue grammar $Q = (V, \tau, W, F, s, P)$ satisfying $L(K) = L(Q)$, ! is a distinguished member of $(W - F)$, $V = U \cup Z \cup \tau$ such that U, Z, τ are pairwise disjoint, and Q derives every $z \in L(Q)$ in this way

$$
\begin{array}{lll}
\#S & \Rightarrow^+ & x\#b_1 b_2 \dots b_n! \\
 & \Rightarrow & x b_1 \# b_2 \dots b_n y_1 p_2 \\
 & \Rightarrow & x b_1 b_2 \# b_3 \dots b_n y_1 y_2 p_3 \\
 & \vdots & \\
 & \Rightarrow & x b_1 b_2 \dots b_{n-1} \# b_n y_1 y_2 \cdots y_{n-1} p_n \\
 & \Rightarrow & x b_1 b_2 \dots b_{n-1} b_n \# y_1 y_2 \cdots y_n p_{n+1}
\end{array}
$$

where $n \in N$, $x \in U^*$, $b_i \in Z$ for $i = 1, \dots, n$, $y_i \in \tau^*$ for $i = 1, \dots, n$, $z = y_1 y_2 \dots y_n$, $p_i \in W - \{!\}$ for $i = 1, \dots, n-1$, $p_n \in F$, and in this derivation $x\#b_1 b_2 \dots b_n!$ is the only word containing !.

Proof:
see [5]. □

Theorem 4.2. Let Q be a left-extended queue grammar satisfying the properties of Theorem 4.1. Then, there is a linear grammar, G, and a one-turn atomic pushdown automaton $M = (\{\lfloor\}, \tau, \{0,1\}, H, \lfloor, \$, \{\lfloor\})$ such that $card(H) = card(\tau) + 4$ and $L(Q) = L(M, L(G), 3)$.

Proof:
Let $Q = (V, \tau, W, F, s, R)$ be a queue grammar satisfying the properties of Theorem 4.1. For some $n \geq 1$, introduce a homomorphism f from R to X, where $X = (\{1\}^*\{0\}\{1\}^*\{1\}^n \cap \{0,1\}^{2n})$. Extend f so it is defined from R^* to X^*. Define the substitution h from V^* to X^* as $h(a) = \{f(r) : r = (a, p, x, q) \in R$ for some $p, q \in W, x \in V^*\}$. Define the coding d from $\{0,1\}^*$ to $\{2,3\}^*$ as $d(0) = 2$, $d(1) = 3$. Construct the linear grammar $G = (N, T, P, S)$ as follows. Set

$$T = \{0, 1, 2, 3\} \cup \tau$$

$$N = \{S\} \cup \{\tilde{q} : q \in W\} \cup \{\hat{q} : q \in W\}$$

$$P = \{S \rightarrow \tilde{f} : f \in F\} \cup \{\tilde{!} \rightarrow \hat{!}\}$$

Extend P by performing 1 through 3 given next.

1. for every $r = (a, p, x, q) \in R, p, q \in w, x \in T^*$: $P = P \cup \{\tilde{q} \rightarrow \tilde{p}d(f(r))x\}$
2. for every $(a, p, x, q) \in R$: $P = P \cup \{\hat{q} \rightarrow y\hat{p}b : y \in rev(h(x)), b \in h(a)\}$
3. for every $(a, p, x, q) \in R$, $ap = S, p, q \in W, x \in V^*$: $P = P \cup \{\hat{q} \rightarrow y : y \in rev(h(x))\}$

Define the pushdown automaton $M = (\{\lfloor\}, \tau, \{0,1\}, H, \lfloor, \$, \{\lfloor\})$, where H contains the next transition rules:

0. $\lfloor \rightarrow 0\lfloor$
1. $\lfloor \rightarrow 1\lfloor$
2. $0\lfloor \rightarrow \lfloor$
3. $1\lfloor \rightarrow \lfloor$

a. $\lfloor a \rightarrow \lfloor$ for every $a \in \tau$

We next demonstrate that $L(M, L(G, 3), 3) = L(Q)$.

To demonstrate $L(M, L(G, 3), 3) = L(Q)$, observe that M accepts every word w as

$$
\begin{array}{lll}
\$w_1 \dots w_{m-1}w_m & \Rightarrow^{+} & \$\bar{b}_m \dots \bar{b}_1 \bar{a}_n \dots \bar{a}_1 \lfloor w_1 \dots w_{m-1}w_m \\
& \Rightarrow & \$\bar{b}_m \dots \bar{b}_1 \bar{a}_n \dots \bar{a}_1 \lfloor w_1 \dots w_{m-1}w_m \\
& \Rightarrow^{n} & \$\bar{b}_m \dots \bar{b}_1 \lfloor w_1 \dots w_{m-1}w_m \\
& \Rightarrow & \$\bar{b}_m \dots \bar{b}_1 \lfloor w_1 \dots w_{m-1}w_m \\
& \Rightarrow^{|w_1|} & \$\bar{b}_m \dots \bar{b}_1 \lfloor w_2 \dots w_{m-1}w_m \\
& \Rightarrow & \$\bar{b}_m \dots \bar{b}_2 \lfloor w_2 \dots w_{m-1}w_m \\
& \Rightarrow^{|w_2|} & \$\bar{b}_m \dots \bar{b}_2 \lfloor w_3 \dots w_{m-1}w_m \\
& \Rightarrow & \$\bar{b}_m \dots \bar{b}_3 \lfloor w_3 \dots w_{m-1}w_m \\
& \vdots & \\
& \Rightarrow & \$\bar{b}_m \lfloor w_m \\
& \Rightarrow^{|w_m|} & \$\bar{b}_m \lfloor \\
& \Rightarrow & \$\lfloor
\end{array}
$$

according to a word of the form $\beta\alpha\alpha'\beta' \in L(G)$ where

$$
\begin{array}{lll}
\beta & = & \mathrm{r}ev(f(r_m))\mathrm{r}ev(f(r_{m-1}))\dots \mathrm{r}ev(f(r_1)), \\
\alpha & = & \mathrm{r}ev(f(t_n))\mathrm{r}ev(f(t_{n-1}))\dots \mathrm{r}ev(f(t_1)), \\
\alpha' & = & f(t_0)f(t_1)\dots f(t_n), \\
\beta' & = & d(f(r_1))w_1 d(f(r_2))w_2 \dots d(f(r_m))w_m,
\end{array}
$$

for some $m, n \geq 1$ so that

for $i = 1, \dots, m$,

$t_i \quad = \quad (b_i, q_i, w_i, q_{i+1}) \in R$, $b_i \in V \setminus \tau$, $q_i, q_{i+1} \in Q$, $\bar{b}_i = f(t_i)$

for $j = 1, \dots, n+1$,

$r_j \quad = \quad (a_{j-1}, p_{j-1}, x_j, p_j)$, $a_{j-1} \in V \setminus \tau$, $p_{j-1}, p_j \in Q \setminus F$, $x_j \in (V \setminus \tau)^*$, $\bar{a}_j = f(r_j)$, $q_{m+1} \in F$, $\bar{a}_0 p_0 = s$

Thus, in Q,

$$
\begin{array}{lll}
\#a_0p_0 & \Rightarrow a_0\#y_0x_1p_1 & [(a_0, p_0, x_1, p_1)] \\
& \Rightarrow a_0a_1\#y_1x_2p_2 & [(a_1, p_1, x_2, p_2)] \\
& \Rightarrow a_0a_1a_2\#y_2x_3p_3 & [(a_2, p_2, x_3, p_3)] \\
& \vdots & \\
& \Rightarrow a_0a_1a_2\dots a_{n-1}\#y_{n-1}x_np_n & [(a_{n-1}, p_{n-1}, x_n, p_n)] \\
& \Rightarrow a_0a_1a_2\dots a_n\#y_nx_{n+1}q_1 & [(a_n, p_n, x_{n+1}, q_1)] \\
& \Rightarrow a_0\dots a_nb_1\#b_2\dots b_mw_1q_2 & [(b_1, q_1, w_1, q_2)] \\
& \Rightarrow a_0\dots a_nb_1b_2\#b_3\dots b_mw_1w_2q_3 & [(b_2, q_2, w_2, q_3)]
\end{array}
$$

$$\vdots$$
$$\Rightarrow a_0 \ldots a_n b_1 \ldots b_{m-1} \# b_m w_1 w_2 \ldots w_{m-1} q_m \qquad [(b_{m-1}, q_{m-1}, w_{m-1}, q_m)]$$
$$\Rightarrow a_0 \ldots a_n b_1 \ldots b_m \# w_1 w_2 \ldots w_m q_{m+1} \qquad [(b_m, q_m, w_m, q_{m+1})]$$

Therefore, $w_1 w_2 \ldots w_m \in L(Q)$. Consequently, $L(M, L(G), 3) \subseteq L(Q)$.

A proof that $L(Q) \subseteq L(M, L(G), 3)$ is left to the reader.

As $L(Q) \subseteq L(M, L(G), 3)$ and $L(M, L(G), 3) \subseteq L(Q)$, $L(Q) = L(M, L(G), 3)$. Observe that M is atomic and one-turn. Furthermore, $card(H) = card(\tau) + 4$. Thus, Theorem 4.2 holds. □

Theorem 4.3. For every $L \in RE$, there is a linear language Ξ, and a one-turn atomic pushdown automaton, $M = (Q, \Sigma, \Omega, R, s, \$, F)$ such that $card(Q) \leq 1$, $card(\Omega) \leq 2$, $card(R) \leq card(\Sigma) + 4$, and $L(M, \Xi, 3) = L$.

Proof:
By Theorem 2.1 in [3], for every $L \in RE$, there is a queue grammar Q such that $L = L(Q)$. Clearly, there is a left-extended queue grammar, Q', such that $L(Q) = L(Q')$. Thus, this theorem follows from Theorems 4.1 and 4.2. □

Theorem 4.4. For every $L \in RE$, there is a linear language Ξ, and a one-turn atomic pushdown automaton, $M = (Q, \Sigma, \Omega, R, s, \$, F)$ such that $card(Q) \leq 1$, $card(\Omega) \leq 2$, $card(R) \leq card(\Sigma) + 4$, and $L(M, \Xi, 1) = L$.

Proof:
Prove this theorem by analogy with the demonstration of Theorem 4.3. □

Theorem 4.5. For every $L \in RE$, there is a linear language Ξ, and a one-turn atomic pushdown automaton, $M = (Q, \Sigma, \Omega, R, s, \$, F)$ such that $card(Q) \leq 1$, $card(\Omega) \leq 2$, $card(R) \leq card(\Sigma) + 4$, and $L(M, \Xi, 2) = L$.

Proof:
Prove this theorem by analogy with the demonstration of Theorem 4.3. □

Theorem 4.6. For $i \in \{1, 2, 3\}$, $RE = \mathcal{L}(LIN, i)$.

Proof:
This theorem follows from the previous three theorems. □

References

[1] Dassow, J. and Paun, G.: *Regulated Rewriting in Formal Language Theory*. Springer, New York, 1989.

[2] Harrison, M.: *Introduction to Formal Language Theory*. Addison Wesley, Reading, 1978

[3] Kleijn, H. C. M. and Rozenberg, G.: On the Generative Power of Regular Pattern Grammars, *Acta Informatica*, Vol. 20, pp. 391–411, 1983.

[4] Meduna, A.: *Automata and Languages: Theory and Applications*. Springer, London, 2000.

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