REDUCING DEEP PUSHDOWN AUTOMATA

Ing. Zbyněk KŘIVKA, Doctoral Degree Programme (2)
Dept. of Information Systems, FIT, BUT
E-mail: krivka@fit.vutbr.cz

Ing. Rudolf SCHÖNECKER, Doctoral Degree Programme (1)
Dept. of Information Systems, FIT, BUT
E-mail: schonec@fit.vutbr.cz

Supervised by: Prof. Alexander Meduna

ABSTRACT

This contribution presents reducing variant of the deep pushdown automata. Deep pushdown automata is a new generalization of the classical pushdown automata. Basic idea of the modification consists of allowing these automata to access more deeper parts of pushdown and reducing strings to non-input symbols in the pushdown. It works similarly to bottom-up analysis simulation of context-free grammars in the classical pushdown automata. Further, this paper presents results of equivalence of reducing deep pushdown automata with *n*-limited state grammars and infinite hierarchy of language families based on that.

1 INTRODUCTION

Consider the standard simulation of a context-free grammar by a classical pushdown automaton acting as a general bottom-up parser (see [4]). During every move, the parser either shifts or reduces its pushdown depending on the top pushdown symbol, current input symbol, and state. Shift operation takes one input symbol and moves it to the top of the pushdown. If a reversal string on the top of the pushdown equals to any right-handed side of a context-free production, this string is reduced to one non-input symbol.

In this paper, we discuss one variant of a slight generalization of this automaton. Hereafter, the generalized bottom-up parser represented by pushdown automaton works exactly the same as the above automaton except that it can make *reductions of depth m* so it replaces the pushdown's substring with *m*th topmost non-input symbol in the pushdown, for some $m \geq 1$. We call it *reducing deep pushdown automaton* (abbrev. *RDPDA*) and it is a modification of the recently published generalizations of pushdown automata (see [3, 5]).

RDPDA has no input tape because the input string is immediately part of the pushdown in the start configuration of *RDPDA*. The pushdown bottom represented by *bottom* symbol corresponds to endmarker of the input string (used in LL(k) translation, see [1]). This minor property can be also simulated by reading the input tape from the right to the left by shift operations. *RDPDA* also do not need start pushdown symbol.

2 PRELIMINARIES

This paper assumes that the reader is familiar with the theory of automata, formal languages, and parsing (see [1, 4]). For a set, Q, $card(Q)$ denotes the cardinality of Q. I denotes the set of all positive integers. For an alphabet, V, V^* represents the free monoid generated by V under the operation of concatenation. The identity 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 $w \in V^*$, $|w|$ denotes the length of w and $alph(w)$ denotes the set of symbols occurring in w. For $W \subseteq V$, $occur(w,W)$ denotes the number of occurrences of symbols from W in w. For every $i \geq 0$, $\mathbf{prefix}(w,i)$ is w's prefix of length i if $|w| \geq i$, and $\mathbf{prefix}(w,i) = w$ if $i \geq |w|+1$.

A *state grammar* (see [2]) is a quintuple, $G = (V,W,T,P,S)$, where V is a *total alphabet*, W is a finite set of *states*, $T \subseteq V$ is an *alphabet of terminals*, $S \in (V-T)$ is the *start symbol*, and $P \subseteq (W \times (V-T)) \times (W \times V^+)$ is a finite relation. Instead of $(q,A,p,v) \in P$, we write $(q,A) \to (p,v) \in P$ throughout. For every $z \in V^*$, set ${}_G states(z) = \{q | (q,B) \to (p,v) \in P$, where $B \in (V-T) \cap alph(z), v \in V^+, q,p \in W\}$. If $(q,A) \to (p,v) \in P, x,y \in V^*, {}_G states(x) = \emptyset$, then G makes a *derivation step* from (q,xAy) to (p,xvy), symbolically written as $(q,xAy) \Rightarrow (p,xvy)$ $[(q,A) \to (p,v)]$ in G; in addition, if n is a positive integer satisfying $occur(xA,V-T) \leq n$, we say that $(q,xAy) \Rightarrow (p,xvy)$ $[(q,A) \to (p,v)]$ is *n-limited*, symbolically written as $(q,xAy)\ {}_n\!\Rightarrow (p,xvy)$ $[(q,A) \to (p,v)]$. Whenever there is no danger of confusion, we simplify $(q,xAy) \Rightarrow (p,xvy)$ $[(q,A) \to (p,v)]$ and $(q,xAy)\ {}_n\!\Rightarrow (p,xvy)$ $[(q,A) \to (p,v)]$ to $(q,xAy) \Rightarrow (p,xvy)$ and $(q,xAy)\ {}_n\!\Rightarrow (p,xvy)$, respectively. In the standard manner, we extend $\Rightarrow$ *to* $\Rightarrow^m$, where $m \geq 0$; then, based on $\Rightarrow^m$, we define $\Rightarrow^+$ *and* $\Rightarrow^*$. Let $n \in I$ and $\upsilon, \varpi \in (W \times V^+)$. To express that every derivation step in $\upsilon \Rightarrow^m \varpi, \upsilon \Rightarrow^+ \varpi$, and $\upsilon \Rightarrow^* \varpi$ is n-limited, we write $\upsilon\ {}_n\!\Rightarrow^m \varpi, \upsilon\ {}_n\!\Rightarrow^+ \varpi$, and $\upsilon\ {}_n\!\Rightarrow^* \varpi$ instead of $\upsilon \Rightarrow^m \varpi, \upsilon \Rightarrow^+ \varpi$, and $\upsilon \Rightarrow^* \varpi$, respectively. The *language of* G, $L(G)$, is defined as $L(G) = \{w \in T^* | (q,S) \Rightarrow^* (p,w), q,p \in W\}$. Furthermore, we define for every $n \geq 1, L(G,n) = \{w \in T^* | (q,S)\ {}_n\!\Rightarrow^* (p,w), q,p \in W\}$, and $L(G,n)$ is called *n-limited language of* G. A derivation of the form $(q,S)\ {}_n\!\Rightarrow^* (p,w)$, where $q,p \in W$ and $w \in T^*$, represents a *successful n-limited generation* of w in G. A state grammar G is of *degree n* for a positive integer n if and only if $L(G,n) = L(G)$. $\boldsymbol{ST}_n$ denotes the family of languages containing (n or less)-limited languages of arbitrary state grammar. More formally, for every $n \geq 1$, set $\boldsymbol{ST}_n = \{L(G,i) |\ G$ is an arbitrary state grammar, $1 \leq i \leq n\}$. If $L(G,n) \neq L(G)$ for every positive integer n, then G is state grammar of *infinite degree*. Let $\boldsymbol{ST}_\infty = \bigcup_{n=1}^{\infty} \boldsymbol{ST}_n$. Let $\boldsymbol{ST}_\omega$ be the entire family of state languages.

CF and ***CS*** denote the families of context-free and context-sensitive languages, respectively.

Kasai proved in his paper (see [2]) these crucial theorems concerning state grammars (reformulated in the terms of this paper):

Theorem Kasai.2. $\boldsymbol{ST}_\omega = \boldsymbol{CS}$.

Corollary Kasai.1. $\boldsymbol{ST}_\infty \subset \boldsymbol{ST}_\omega$.

Theorem Kasai.5. For every $n \geq 1$, $\boldsymbol{ST}_n \subset \boldsymbol{ST}_{n+1}$.

Observe that for each $n \geq 1$, $\boldsymbol{ST}_n \subseteq \boldsymbol{ST}_{n+1}$ follows from the definition of state languages.

3 DEFINITIONS

A *reducing deep pushdown automaton*, a *RDPDA* for short, is a 6-tuple, $M = (Q, \Sigma, \Gamma, R, s, F)$, where Q is a finite set of *states*, Σ is an *input alphabet*, and Γ is a *pushdown alphabet*, I, Q, Γ are pairwise disjoint (see Section 2 for I), $\Sigma \subseteq \Gamma$, $\Gamma - \Sigma$ contains a special *bottom* symbol denoted by #, $R \subseteq (I \times Q \times (\Gamma - \{\#\})^+ \times Q \times (\Gamma - (\Sigma \cup \{\#\}))) \cup (I \times Q \times (\Gamma - \{\#\})^* \{\#\} \times Q \times \{\#\})$ is a *finite relation*, $s \in Q$ is the *start state*, $F \subseteq Q$ is a set of *final states*. Instead of $(m, q, v, p, A) \in R$, we write $qv \vdash mpA \in R$ and call $qv \vdash mpA$ a *rule*; accordingly, R is referred to as the *set of M's rules*. A *configuration of M* is a pair in $Q \times (\Gamma - \{\#\})^* \{\#\}$. Let χ denote the set of all configurations of M. Let $x, y \in \chi$ be two configurations. *M reduces* its pushdown (or makes a *move*) from x to y, symbolically written as $x \vDash y$, if $x = (q, uvz), y = (p, uAz)$, $qv \vdash mpA \in R$, where $A \in \Gamma - \Sigma, u, v, z \in \Gamma^*, q, p \in Q$, and $occur(u, \Gamma - \Sigma) = m - 1$. To express that M makes $x \vDash y$ according to $qv \vdash mpA$, we write $x \vDash y$ $[qv \vdash mpA]$. We say that $qv \vdash mpA$ is a *rule of depth m*; accordingly, $x \vDash y$ $[qv \vdash mpA]$ is a *reduction of depth m*. If $n \in I$ is the minimal positive integer such that each of M's rules is of depth n or less, we say that *M is of depth n*, symbolically written as ${}_nM$. In the standard manner, extend $\vDash$ to $\vDash^m$, respectively, for $m \geq 0$; then, based on $\vDash^m$ define $\vDash^+$, and $\vDash^*$.

Let M be of depth n, for some $n \in I$. We define the *language reduced by ${}_nM$, $L({}_nM)$*, as $L({}_nM) = \{w \in \Sigma^* | \ (s, w\#) \vDash^* (f, \#)$ in ${}_nM$ with $f \in F\}$.

For every every $k \geq 1$, set $\boldsymbol{RDPD}_k = \{L({}_iM) \mid {}_iM$ is a *RDPDA*, $1 \leq i \leq k\}$.

Example 1 *Consider a RDPDA*, ${}_2M = (\{s, t, q, p, f\}, \{a, b, c\}, \{A, B, \#\}, R, s, \{f\})$ *with*

$$
\begin{array}{rrcl}
R = \{ & sab & \vdash & 1tA, \\
 & tc & \vdash & 2pB, \\
 & paAb & \vdash & 1qA, \\
 & qBc & \vdash & 2pB, \\
 & pAB\# & \vdash & 1f\# \quad \}.
\end{array}
$$

With aabbcc, M makes

$$
\begin{array}{rcll}
(s, aabbcc\#) & \vDash & (t, aAbcc\#) & [sab \vdash 1tA] \\
 & \vDash & (p, aAbBc\#) & [tc \vdash 2pB] \\
 & \vDash & (q, ABc\#) & [paAb \vdash 1qA] \\
 & \vDash & (p, AB\#) & [qBc \vdash 2pB] \\
 & \vDash & (f, \#) & [pAB\# \vdash 1f\#]
\end{array}
$$

We write $(s, aabbcc\#) \vDash^* (f, \#)$, *and we say that the string aabbcc is successfully reduced by RDPDA M. Observe that* $L(M) = \{a^n b^n c^n | n \geq 1\} \in \boldsymbol{RDPD}_2$, *and* $L(M) \in \boldsymbol{CS} - \boldsymbol{CF}$.

4 RESULTS

Lemma 1 *For every $n \geq 1$ and every state grammar, G, there exists RDPDA of depth n, ${}_nM$, such that $L(G,n) = L({}_nM)$.*

Construction. Let $G = (V,W,T,P,S)$ be a state grammar and $n \geq 1$. Set $N = V - T$. Define the homomorphism f over $(\{\#\} \cup V)^*$ as $f(A) = A$ for every $A \in \{\#\} \cup N$, and $f(a) = \varepsilon$ for every $a \in T$. Introduce the *RDPDA* of depth n,

$${}_nM = (Q, T, V \cup \{\#\}, R, s, \{\$\}),$$

where $Q = \{s,\$\} \cup \{\langle p,u \rangle \mid p \in W, u \in \textbf{prefix}(v,n), v \in N^*\{\#\}^n\}$ and R is constructed by performing the following steps:

1. if $(p,A) \rightarrow (q,x) \in P$, and $(t,S)\ {}_n\Rightarrow^+ (q,w)$ with $w \in T^*$ for some $p,q,t \in W$, $A \in N$, $x \in V^+$, then add
   $s\# \vdash 1\langle q, \#^n \rangle \#$ to R;
2. if $(p,A) \rightarrow (q,x) \in P$, for $\langle q, \textbf{prefix}(f(uxv)\#^n, n) \rangle \in Q$, $p,q \in W$, $A \in N$, $x \in V^+$, $u,v \in V^*$, $|f(u)| = m-1$, $m \in I$, $1 \leq m \leq n$, then add
   $\langle q, \textbf{prefix}(f(uxv)\#^n, n) \rangle x \vdash m \langle p, \textbf{prefix}(f(u)Af(v)\#^n, n) \rangle A$ to R
3. for every $(p,S) \rightarrow (q,x) \in P$, $p,q \in W$, $x \in V^+$, $\langle q, \textbf{prefix}(f(x)\#^n, n) \rangle \in Q$, add
   $\langle q, \textbf{prefix}(f(x)\#^n, n) \rangle x\# \vdash 1\$\#$ to R

Basic Idea. Every n-limited derivation step in G is simulated by reversal reduction step in ${}_nM$. So, if some nonterminal (ith from left) is rewritten by string in G, then exactly the same string on ${}_nM$'s pushdown is replaced by the same non-input symbol in the depth of i, $1 \geq i \geq n$. ${}_nM$'s states are composed of two components: (a) original G's state and (b) string of length n which remembers first n nonterminals in current sentential form (completed by # symbols from behind if needed). When G successfully completes the generation of a string of terminals, ${}_nM$ completes by entering the final state \$ and with empty pushdown.

Lemma 2 *For every $n \geq 1$ and RDPDA of depth n, ${}_nM$, there exists state grammar, G, such that $L({}_nM) = L(G,n)$.*

Construction. Let $n \geq 1$ and ${}_nM = (Q,T,V,R,s,F)$ be a *RDPDA*. Let Z and \$ be two new symbols that occur in no component of ${}_nM$. Set $N = V - T$. Introduce sets $C = \{\langle q,i,\triangleright \rangle | q \in Q, 1 \leq i \leq n\}, D = \{\langle q,i,\triangleleft \rangle | q \in Q, 0 \leq i \leq n-1\}$, an alphabet W such that $card(V) = card(W)$, and for all $1 \leq i \leq n$, an alphabet U_i such that $card(U_i) = card(N)$. Without any loss of generality, assume that V, Q, and all these newly introduced sets and alphabets are pairwise disjoint. Set $U = \cup_{i=1}^n U_i$. Introduce a bijection h from V to W. For each $1 \leq i \leq n$, introduce a bijection ${}_ig$ from N to U_i. Define the state grammar

$$G = (V \cup W \cup U \cup \{Z\}, Q \cup C \cup D \cup \{\$\}, T, P, S),$$

where P is constructed by performing the following steps:

1. for every $pxY\# \vdash 1f\#$, $f \in F$, $x \in V^*$, $Y \in V$, $p \in Q$, add
   $(f, S) \to (\langle p, 1, \triangleright\rangle, xh(Y))$ to P;
2. for every $q \in Q, A \in N, 1 \leq i \leq n-1, x \in V^+$, add
   $(\langle q, i, \triangleright\rangle, A) \to (\langle q, i+1, \triangleright\rangle, {}_ig(A))$ and
   $(\langle q, i, \triangleleft\rangle, {}_ig(A)) \to (\langle q, i-1, \triangleleft\rangle, A)$ to P;
3. if $pxY \vdash iqA \in R$, for some $p, q \in Q, A \in N, x \in V^*, Y \in V, i = 1, \ldots, n$, then add
   $(\langle q, i, \triangleright\rangle, A) \to (\langle p, i-1, \triangleleft\rangle, xY)$ and
   $(\langle q, i, \triangleright\rangle, h(A)) \to (\langle p, i-1, \triangleleft\rangle, xh(Y))$ to P;
4. for every $q \in Q$, $A \in N$, $Y \in V$, add
   $(\langle q, 0, \triangleleft\rangle, A) \to (\langle q, 1, \triangleright\rangle, A)$ and
   $(\langle q, 0, \triangleleft\rangle, h(Y)) \to (\langle q, 1, \triangleright\rangle, h(Y))$ to P;
5. for every $a \in T$, add
   $(\langle s, 0, \triangleleft\rangle, h(a)) \to (\$, a)$ to P.

Basic Idea. G simulates reversal effect of the application of the rule $px \vdash iqA \in R$. G scans (left-to-right) the sentential form, counts the occurrences of nonterminals until it reaches the ith occurrence of a nonterminal. If it is A, G replaces it with x which corresponds to reducing x to A by ${}_nM$. G completes the simulation of the reduction of a string x by ${}_nM$ so it marks every last symbol by bijection h and in the last step rewrites it to the terminal, to generate x. Bijection h compensates non-existence of the final state in G.

Due to the insufficient space in this contribution, rigorous proofs are omitted.

Theorem 3 *For every $k \geq 1$, $\boldsymbol{RDPD}_k \subset \boldsymbol{RDPD}_{k+1}$.*

Proof. Clearly, $\boldsymbol{RDPD}_k = \boldsymbol{ST}_k$ is proved by Lemma 1 and 2. So, this theorem follows from Lemma 1, 2, and Theorem Kasai.5 from [2].

ACKNOWLEDGEMENTS

The paper has been prepared with the support of the FRVŠ MŠMT grant FR1909/2006/G1.

REFERENCES

[1] Aho, A. V., Ullman, J. D.: *The Theory of Parsing, Translation and Compiling, Volume I: Parsing* (Prentice Hall, Englewood Cliffs, New Jersey, 1972)

[2] Kasai, T.: An Hierarchy Between Context-Free and Context-Sensitive Languages. In: *Journal of Computer and System Sciences* vol 4 (1970) pp 492-508

[3] Křivka, Z., Meduna, A.: General Top-Down Parsers Based On Deep Pushdown Expansions. In: *Workshop on the Formal Models* (2006, accepted)

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

[5] Meduna, A.: Deep Pushdown Automata. In: *Acta Informatica* (2006, in press)