PUC

Series: Monographs in Computer Science

and Computer Applications

Nº 12/70

ON THE USE OF PUSHDOWN AUTOMATA FOR THE PARSING OF SENTENCES OF CONTEXT-FREE LANGUAGES

by

Roberto Lins de Carvalho

Computer Science Department - Rio Datacenter

TÉCNICO CIENTÍFICO

dade Católica do Rio de Janeiro

de São Vicente, 209 — ZC-20

Rio de Janeiro — Brasil

ON THE USE OF PUSHDOWN AUTOMATA FOR THE PARSING OF SENTENCES OF CONTEXT-FREE LANGUAGES

by

Roberto Lins de Carvalho

Computer Science Department - Rio Datacenter

ABSTRACT

This work is a different presentation of an existing algorithm [3] for parsing sentences of context-free languages.

A formal presentation of the algorithm is given, and improve - ment are made in the processing time.

1. INTRODUCTION:

In this introduction we present some definitions in order to achieve completeness.

We denote the set of all words (sequences of symbols) formed by an alphabet A, including the empty word ε, by A*, and by A+ the set A* - {ε}. A context-free grammar G is a 4-tuple $G = (V_N, V_T, P, \sigma)$, where V_N and V_T are finite non-empty sets, called the set of nonterminals and the set of terminals, respectively, P is a subset of $V_N \times (V_N \cup V_T)^+$ called the set of production rules of G, and σ is an element of V_N, called the start symbol. The interpretation of a context-free grammar (cfg, for short) is the following.

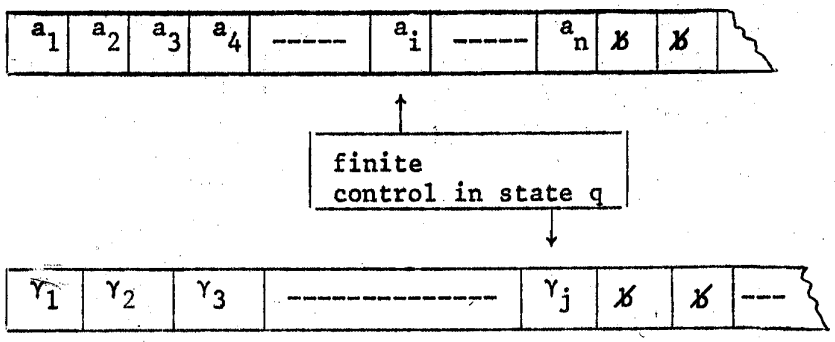
If α is an element of $(V_N \cup V_T)^+$, and a particular element of V_N, say η, occurs in α and there exists in P the pair (η, u), then we can substitute u for that occurence of η in α.

We sometimes use $\eta \to u$ for (η,u), we define a relation $\underset{G}{\Longrightarrow} \subseteq (V_N \cup V_T)^+ \times (V_N \cup V_T)^+$ such that $\alpha \underset{G}{\Longrightarrow} \beta$ iff $\alpha = \beta$ or $\alpha = u\eta v$, $\beta = uwv$ and $(\eta, w) \in P$. Another relation, the transitive closure of $\underset{G}{\Longrightarrow}$, is $\underset{G}{\overset{*}{\Longrightarrow}}$ which is defined by $\alpha \underset{G}{\overset{*}{\Longrightarrow}} \beta$ iff there exist words $\alpha_1, \alpha_2, \ldots, \alpha_n$ such that $\alpha_1 = \alpha$, $\alpha_n = \beta$, and for all $1 \leq i < n$ $\alpha_i \underset{G}{\Longrightarrow} \alpha_i + 1$. Now we define the set $L(G) = \{x \in V_T^+ \mid \sigma \underset{G}{\overset{*}{\Longrightarrow}} x\}$, this set is called the language (cfg) generated by G.

We now present a very useful automaton, called pushdown automata (pda, for short).

Definition 1.1 - A pda is a 7-tuple $A=(K, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ where K is finite non-empty set of states, Σ is a finite non-empty set os input symbols, Γ is a finite non-empty set of pushdown symbols, δ is a function from $K \times (\Sigma \cup \{\varepsilon\}) \times \Gamma$ into finite subsets of $K \times \Gamma^*$, q_0 is the start state, Z_0 is the start pushdown symbol and F is a subset of K and is called the final set of states.

An illustration of such automata is given by the following picture.



the picture shows the automaton in state q, reading input a_i where $a \in \Sigma \cup \{\varepsilon\}$, and scanning the right most symbol (the top) γ_j in the pushdown tape.

An atomic move of the automaton A, if $(q', w) \in \delta(q, a_i, \gamma_j)$, may be the following:

(i) go into state q'

(ii)substitute the word w for γ_j in the pushdown tape.

If $w = \varepsilon$, A erases or "pops-up" the pushdown store.

Formally we have:

Definition 1.2 - An instantaneous description (id, for short) of a pda $(K, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ is an element of $K \times \Sigma^* \times \Gamma^*$, which we denote by (q, x, w).

Definition 1.3 - Let $A = (K, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a pda and define a relation $\vdash$ on id's of A, as follows.

for each $q \in K$, $a \in \Sigma \cup \{\varepsilon\}$, $x \in \Sigma^*$, $w \in \Gamma^*$ and $Z \in \Gamma$, $(q, ax, wZ) \vdash (q', x, wy)$ iff $(q', y) \in \delta(q, a, Z)$ thus $\vdash$ denotes an elementary (atomic) move of the automaton also we define the relation $\vdash^*$ by

$\alpha \vdash^* \beta$ iff there is a sequence of id's $\alpha_1 = \alpha, \alpha_2, \alpha_3, \ldots, \alpha_n = \beta$, such that for all i, $1 \leq i < n$ $\alpha_i \vdash \alpha_{i+1}$

Definition 1.4 - Let $A = (K, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a pda, define the sets.

$T(A) = \{x \in \Sigma^* \mid (q_0, x, Z_0) \vdash^* (q, \varepsilon, w)$, for some $(q, w) \in F \times \Gamma^*\}$.

and $N(A) = \{x \in \Sigma^* \mid (q_0, x, Z_0) \vdash^* (q, \varepsilon, \varepsilon)$, for some $q \in K\}$

Intuitively, T (A) and N (A) are sets of words accepted in two senses:

(i) by the final state

(ii) by the empty pushdown tape

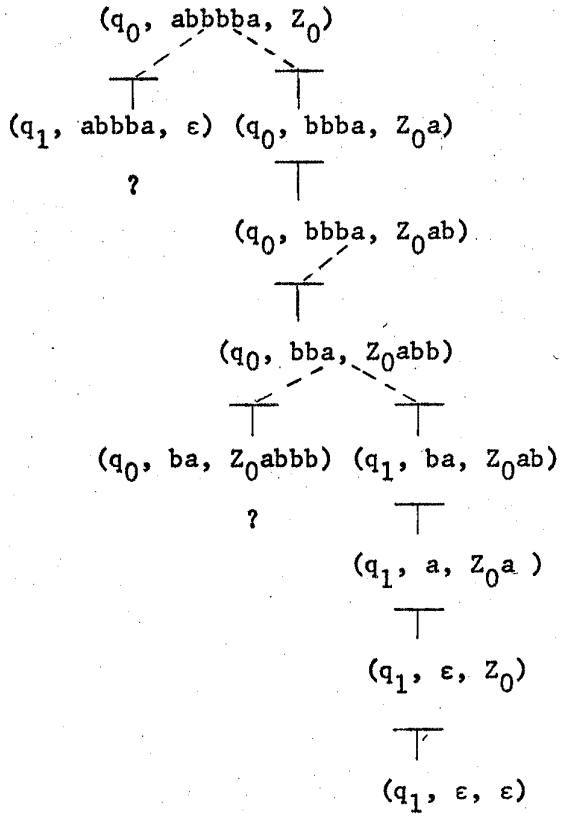
Example 1: Let $A=(K, \Sigma, \Gamma, \delta, q_0, Z_0, \phi)$ be a pda where $K=\{q_0, q_1\}$, $\Sigma = \{a, b\}$, $\Gamma=\{Z_0, a, b\}$ and δ is defined for each $c, A \in \Sigma$, by

$\delta(q_0, c, Z_0) = \{(q_0, Z_0, c)\}$ $\quad$ $\delta(q_0, \varepsilon, Z_0)=\{(q_1, \varepsilon)\}$

$\delta(q_0, c, A) = \{(q_0, Ac)\}$ $\quad$ $\delta(q_1, c, c) = \{(q_1, \varepsilon)\}$

$\delta(q_0, c, c) = \{(q_0, cc),(q_1,\varepsilon)\}$ $\quad$ $\delta(q_1, \varepsilon, Z_0)=\{(q_1, \varepsilon)\}$

If we have, at the beginning the word "abbbba" on the input we shall have the following "derivation tree"



We see that we finally arrive at $(q_1, \varepsilon, \varepsilon)$ and that abbbba ε N(A) by definition. The signal "?" means that by that ramification of the tree we shall fail.

2. RECOGNIZING CONTEXT-FREE LANGUAGES:

We present in this paragraph an algorithm based on pda to recognize cfl.

Theorem 2.1 - For each cfg $G = (V_N, V_T, P, \sigma)$, there exists a pda A such that $N(A) = L(G)$.

Let us define $A = (\{q\}, V_T, V_N \cup V_T, \delta, q, \sigma, \phi)$, where

$\delta(q, \varepsilon, \eta) = \{(q, x^T) \mid \eta \to x \in P\}$

$\delta(q, a, a) = \{(q, \varepsilon)\}$ for each $a \in V_T$, formally, one shows that

$\sigma \overset{*}{\underset{G}{\Longrightarrow}} x, x \in V_T^+$ iff $(q, x, \sigma) \overset{*}{\vdash} (q, \varepsilon, \varepsilon)$

See Ginsburg [1]

Example 2 - Let $G = (\{\sigma\}, \{a, b\}, \{(\sigma, aa), (\sigma, bb), (\sigma, a\sigma a), (\sigma, b\sigma b)\}, \sigma)$ be a cfg.

Then by the theorem 2.1, the pda A is the following:

$A = (\{q\}, \{a, b\}, \{\sigma, a, b\}, \delta, \sigma, q, \phi)$ where δ is defined by:

$\delta(q, \varepsilon, \sigma) = \{(q, aa), (q, bb), (q, a\sigma a), (q, b\sigma b)\}$

$\delta(q, a, a) = \{(q, \varepsilon)\}$

$\delta(q, b, b) = \{(q, \varepsilon)\}$

By example 3, we see that if we want to simulate the pda for a cfg, we must keep track of the derivations and automatically emit the "?" in the tree.

It is easy to prove the following:

Lemma 2.2- Let $A = (K, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a pda, where for all $Z \in \Gamma$ if $\delta(q, \varepsilon, Z) \neq \phi$ then $(p, \varepsilon) \notin \delta(q, \varepsilon, Z)$ for all $p \in K$, suppose that $xy \in N(A)$ and that a possible derivation is $(q_0, xy, Z_0) \vdash^* (q, y, w) \vdash^* (p, \varepsilon, \varepsilon)$, then $|y| \geq |w|$..

The importance of this Lemma is just decide when we can or not to continue the tree, the "?" as in example 3.

Suppose we have the id (q, ax, yZ) where $a \in \Sigma$ and $Z \in \Gamma$, we should like that

$$(q, ax, yZ) \vdash_A^* (p, ax, yva)$$

for in this case, we can continue the computation. If our automaton is the one described in Theorem 2.1, this case corresponds to having

$$Z \underset{G}{\Longrightarrow}^* aw \ , \ w \in (V_N \cup V_T)^*$$

Suppose the following matrices

$$A = [a_{ij}]_{mxn} \ , \text{ where } \quad m = \# (V_N), \ n = \#(V_T)$$

$$\text{and } a_{ij} = \begin{cases} 1 \text{ if } \eta_i \to a_j \, \omega \in P \\ 0 \text{ otherwise} \end{cases}$$

$B = \left[b_{ij}\right]_{m \times m}$, where $b_{ij} = \begin{cases} 1 \text{ if } \eta_1 \rightarrow \eta_1 u \in P \\ 0 \text{ otherwise} \end{cases}$

Then $C = \left[\left(\bigcup_{k=1}^{m} B^k\right) . A\right] \cup A$

where

$B^1 = B$

$B^i = B^{i-1}.B$

and if $A.B = \left[d_{ij}\right]$ then

$d_{ij} = \sum_{k=1}^{m} a_{ik} \cdot b_{kj}$, where $A = \left[a_{ij}\right]_{m \times n}$

$B = \left[b_{ij}\right]_{n \times p}$

Is easy to prove that $c_{ij} = 1$ iff $\eta_i \overset{*}{\underset{G}{\Longrightarrow}} a_j w$, for some $w \in (V_N \cup V_T)^*$

Hence if a derivation yields an id $(q, a_j x, y\eta)$, then we use the C matrix as a test: if $c_{ij} = 1$ we continue, otherwise we fail.

We will present an algorithm in which we use the following notations:

nr - denotes the number of rules of the ordered set P

nv - denotes the number of nonterminals of the ordered set V_N

nt - denotes the number of terminals of the ordered set V_T

v_i - denotes the i^{th} nonterminal, and $v_1 = \sigma$

d_i - denotes the i^{th} terminal

$p_i = p_{i_1} p_{i_2} \ldots p_{i_{k_i}}$ - denotes the i^{th} rule of P; p_{i_1} is the left side and $p_{i_2} \ldots p_{i_{k_i}}$ is the right side

$|p_i|$ - length of the right side of p_i

$A = [a_{ij}]$ - denotes a matrix nv x nr, in which a_{i1} is the number of alternative rules of the monternimal v_i, and a_{ij}, $2 < j \leq a_{i1}+1$ are the ordinal numbers of the alternation rules of the ordered set P.

$C = [c_{ij}]$ - denotes a matrix nv x nt defined by

$$c_{ij} = \begin{cases} 1 & \text{iff } v_i \Longrightarrow^{*} a_j\omega, \text{ for some } \omega \in V^* \\ 0 & \text{otherwise} \end{cases}$$

The pushdown store is denoted by s, its top is denoted by s_t. Suppose we want to replace the top symbol by a string x. We will represent this by $s \leftarrow sx$ and if $s = \epsilon$ by $s \leftarrow x$.

The length of s is denoted by t. If we want to take away a single character of the top of s we make t equal to t-1 ($t \leftarrow t-1$).

The input string is denoted by $x = x_1 x_2 x_3 \ldots x_n$, $n > 1$ where $x_i \in V_T$, $1 \leq i \leq n$

The variable m denotes the scanning position of the string x.

We also use a matrix $B = [b_{ij}]_{Kx4}$ which is constructued by the algorithm and its meaning is: If in a derivation we have $(q_0, x_m \cdots x_n, s_1 s_2 \cdots s_{t-1} v_i) \vdash (q_0, x_m \cdots x_n, s_1 s_2 \cdots s_{t-1} p_{\ell_1 k_{\ell_1}} \cdots p_{\ell_1 2})$ and $\ell-1$ was the last row of B that was constructed then $b_{\ell 1} = i$, $b_{\ell 2} = t$, $b_{\ell 3} = K$ and $b_{\ell 4} = m$ the algorithm follows:

1. $s \leftarrow \sigma$; $t \leftarrow 1$; $\ell \leftarrow 0$; $m \leftarrow 1$
2. If $s_t = v_i$ for some i, $1 < i < nv$, go to 9
3. If $s_t \neq x_m$ go to 7
4. $m \leftarrow m + 1$; $t \leftarrow t - 1$; If $\neq 0$ go to 3
5. If $m < n$ go to 12
6. End (the string x is accepted)
7. If $s_t \neq v_i$ for all i, $1 < i < nv$ go to 12
8. Let x_m be d_j for some j, $1 < j < nt$ then if $c_{ij} = 0$ go to 15
9. $k \leftarrow 2$; $\ell \leftarrow \ell + 1$
10. $\ell_1 \leftarrow a_{ik}$; If $|x_m x_{m+1} \cdots x_n| < t + |p_{\ell 1}|$ go to 12
11. $b_{\ell 1} \leftarrow i$; $b_{\ell 2} \leftarrow t$; $b_{\ell 3} \leftarrow k$; $b_{\ell 4} \leftarrow m$; $s \leftarrow s\, p_{\ell 1}\, k_{\ell 2} \cdots p_{\ell 1} 2$; $t \leftarrow t + |p_{\ell 1}| - 1$; go to 2
12. $K \leftarrow b_{\ell 3} + 1$; $i \leftarrow b_{\ell 1}$; IF $K > a_{ij} + 1$ go to 14
13. $t \leftarrow b_{\ell 2}$; $m \leftarrow b_{\ell 4}$; go to 10

14. $t \leftarrow b_{\ell 2}$; $s_t \leftarrow v_i$

15. $\ell \leftarrow \ell - 1$; If $\ell = 0$ then End (the string is accepted)

16. $r_1 \leftarrow b_{\ell+1,4}$; $r_2 \leftarrow b_{\ell 4}$; If $r_1 = r_2$ go to 12

17. $s \leftarrow s s_t x_{r_1-1} x_{r_1-2} \ldots x_{r_2}$; go to 12

We verify from (15, 16, 17) that the pushdown store has been generated once we back up ($\ell \leftarrow \ell - 1$) in the construction of B.

3 - GRAMMARS WITH LEFT-CYCLIC VARIABLES:

Let $G = (V_N, V_T, P, \sigma)$ be a cfg, we say that a monterminal symbol $\eta \in V_N$ is a left-cyclic variable iff $\eta \overset{*}{\underset{G}{\Longrightarrow}} \eta u$ for some $u \in (V_N \cup V_T)^*$.

Let us suppose that in G we have such a left-cyclic variable with $u \neq \varepsilon$. We show that our algorithm also works in this case.

Let η be such avariable, if in some point of the derivation we have the id $(q_0, ax, \gamma\eta)$ where $|ax| > |\gamma\eta|$ and $\eta \overset{*}{\underset{G}{\Longrightarrow}} a\omega$, we continue the derivation and suppose that $(q_0, ax, \gamma\eta) \overset{*}{\underset{A}{\vdash}} (q_0, ax, \gamma(u^T)^i \eta)$ then since $|u| > 1$ for some $i > 1$ we will have $|\gamma(u^T)^i \eta| > |ax|$, and then by the algorithm we return to some point and change the alternative rule. It is easy to see that the algorithm works, even for extremely long string which incur long processing times.

BIBLIOGRAPHY

1 - Ginsburg, S. - "The Mathematical Theory of Context-Free Languages"
McGraw-Hill - 1966.

2 - Hopcroft & Ullman - "Formal Languages and Their Relations to Automata"

3 - Kuno, S. - "Computer Analysis of Natural Languages"
Proceeding of Symposia in Applied Mathematics, Vol - XIX