Theoretical Computer Science 18 (1982) 33–40
North-Holland Publishing Company

A PUSHDOWN AUTOMATON OR A CONTEXT-FREE GRAMMAR—WHICH IS MORE ECONOMICAL?*,**

Jonathan GOLDSTINE, John K. PRICE*** and Detlef WOTSCHKE
Computer Science Department, The Pennsylvania State University, University Park, PA 16802, U.S.A.

Communicated by R. Book
Received January 1980
Revised September 1980

Abstract. For every pair of positive integers n and p, there is a language accepted by a real-time deterministic pushdown automaton with n states and p stack symbols and size $O(np)$, for which every context-free grammar needs at least n^2p+1 nonterminals if $n>1$ (or p non-terminals if $n=1$). It follows that there are context-free languages which can be recognized by pushdown automata of size $O(np)$, but which cannot be generated by context-free grammars of size smaller than $O(n^2p)$; and that the standard construction for converting a pushdown automaton to a context-free grammar is optimal in the sense that it infinitely often produces grammars with the fewest number of nonterminals possible.

1. Introduction

When trying to describe a context-free language as concisely as possible, one often hesitates for a second or two in order to decide whether to use a pushdown automaton (PDA) or a context-free grammar (CFG). Most people would agree that for some languages, a PDA is preferable, and for others a CFG is. But many arguments have been presented, particularly by designers of programming languages, as to why a grammar is usually the preferable tool for specifying a context-free language. So the question remains: why, then, does one sometimes prefer a PDA for the specification of a particular language?

In this paper, we offer a partial answer to this question. We show that, for every pair of positive integers n and p, there is a language that can be accepted by a PDA with n states and p stack symbols and size proportional to np, but for which every CFG must have at least $n^2p \oplus 1$ nonterminals, where

$$n^2p \oplus 1 = \textbf{if } n>1 \textbf{ then } n^2p+1 \textbf{ else } n^2p \textbf{ fi},$$

* A preliminary version of this paper was presented at the Fifth International Colloquium on Automata, Languages and Programming in Udine, Italy, July 1978.

** This research was supported in part by the National Science Foundation under Grants MCS76-10076 and MCS76-10076A01.

*** Current address: Bell Laboratories, Holmdel, NJ 07733, U.S.A.

0304-3975/82/0000–0000/$02.75

and hence must have size at least proportional to n^2p. It follows that some context-free languages can be defined much more concisely by PDAs than by CFGs. It also follows that the usual algorithm for converting a PDA to a CFG, which uses the seemingly large number of $n^2p \oplus 1$ non-terminals for the CFG, cannot be improved, in the sense that no algorithm is possible that always produces fewer nonterminals. This is shown in Section 2.

In Section 3, similar techniques are employed to show that the usual algorithm for intersecting a context-free language (specified by a CFG with p nonterminals) and a regular set (specified by a deterministic finite automaton with n states) is optimal in the sense that the resulting CFG will infinitely often have the minimal number of nonterminals possible. These techniques are also used to show that, for each $n \geq 1$ and $p \geq 2$, there is a linear context-free language accepted by a one-turn deterministic real-time PDA with n states and p stack symbols for which every CFG (whether linear or not) needs at least $n^2(p-1) \oplus 1$ nonterminals.

Section 4 has a somewhat different perspective. It deals with the question of whether the smallest CFGs which generate the languages in Section 2 need to exploit the full power of arbitrary CFGs by using ambiguity, e-productions, and the like. In view of other results on succinctness and economy of description, it would not be surprising if these grammars had to be ambiguous and had to use many e-productions. However, because the PDAs in Section 2 are deterministic, accept by empty stack, and make no e-moves, they can be used to obtain minimal grammars which are very simple in the sense that they are LR(0), hence unambiguous, and are in Greibach Normal Form with no e-productions.

Section 5 contains some concluding remarks.

2. PDA's which are smaller than CFG's

The standard method for converting a PDA which accepts by empty stack into an equivalent CFG is the following 'triple' construction (see, e.g. [3]). Let $M = (Q, \Sigma, \Gamma, \delta, \bar{q}, \bar{Z}, \emptyset)$ be a PDA with $n = \#Q$ states and $p = \#\Gamma$ stack symbols. Then for each move

$$(q_1, A_1A_2 \cdots A_r) \in \delta(q_0, x, A_0) \tag{1}$$

sending M from state q_0 to q_1 while consuming input $x \in \Sigma \cup \{e\}$ (e the empty string) and replacing A_0 on top of the stack by $A_1A_2 \cdots A_r$ with A_1 on top, the triple construction introduces context-free productions

$$[q_0, A_0, q_{r+1}] \to x[q_1, A_1, q_2][q_2, A_2, q_3] \cdots [q_r, A_r, q_{r+1}],$$
$$(q_2, \ldots, q_{r+1}) \in Q^r. \tag{2}$$

Here, the $[q_i, A_i, q_j]$ are nonterminals and $r \geq 0$. It is then a simple matter to show by induction on the length of computations and derivations that each nonterminal

$[q, A, q']$ generates the terminal string w iff the input w can send M through some sequence of moves whose net effect is to change state from q to q' while removing A from the stack. Hence, adding the productions

$$S \to [\bar{q}, \bar{Z}, q], \quad q \in Q,$$

if $n = \# Q > 1$, or setting

$$S = [\bar{q}, \bar{Z}, \bar{q}]$$

if $Q = \{\bar{q}\}$, results in a CFG G with start variable S generating the same language that M accepts by empty stack.

The CFG G has $n^2p \oplus 1$ nonterminals, where

$$n^2p \oplus 1 = \textbf{if } n > 1 \textbf{ then } n^2p + 1 \textbf{ else } n^2p \textbf{ fi},$$

and it has n^r productions of form (2) for each move of M of form (1). Because of the profligate number of productions introduced by this construction, the grammar G is generally much larger than necessary. For example, if the productions of form (2) are replaced by the following productions;

$$\begin{aligned}
&[q_0, A_0, q_{r+1}] \to x[q_1, A_1A_2 \cdots A_r, q_{r+1}], && q_{r+1} \in Q,\\
&[q_1, A_1A_2 \cdots A_r, q_{r+1}] \to [q_1, A_1, q_2][q_2, A_2 \cdots A_r, q_{r+1}], && (q_2, q_{r+1}) \in Q^2,\\
&\qquad\vdots\\
&[q_{r-1}, A_{r-1}A_r, q_{r+1}] \to [q_{r-1}, A_{r-1}, q_r][q_r, A_r, q_{r+1}], && (q_{r-1}, q_r, q_{r+1}) \in Q^3,
\end{aligned}$$

then an equivalent grammar is obtained which uses more nonterminals but which has $O(rn^3)$ rather than n^r productions for each move of form (1). Thus, the CFG produced by the standard triple construction will in general be far from minimal in size, whether size is measured by the number of productions or by the sum of the lengths of all productions.

Since the standard triple construction also introduces a large number $n^2p \oplus 1$ of nonterminals, it is somewhat surprising that the construction is not at all profligate in this regard, as we now prove.

First, we introduce some notation. The letters s, r, u and d are mnemonic for set, reset, up and down, respectively.

Notation. For positive integers n and p, let M_{np} be the PDA

$$M_{np} = (Q_n, \Sigma_{np}, \Gamma_p, \delta_{np}, q_1, Z_1, \emptyset),$$

where

$$\begin{aligned}
&Q_n = \{q_1, \ldots, q_n\}, \qquad \Gamma_p = \{Z_1, \ldots, Z_p\},\\
&\Sigma_{np} = \{s_{ij}, r_{ij}, u, d \mid 1 \leq i \leq n, 1 \leq j \leq p\},
\end{aligned}$$

and where δ_{np} is the following partial function from $Q_n \times \Sigma_{np} \times \Gamma_p$ to $Q_n \times \Gamma_p^*$:

$$\delta_{np}(q_1, s_{ij}, Z_1) = (q_i, Z_j), \quad 1 \leq i \leq n, 1 \leq j \leq p,$$

$$\delta_{np}(q_i, r_{ij}, Z_j) = (q_1, Z_1), \quad 1 \leq i \leq n, 1 \leq j \leq p,$$

$$\delta_{np}(q_i, u, Z_j) = (q_i, Z_jZ_p), \quad 1 \leq i \leq n, 1 \leq j \leq p,$$

$$\delta_{np}(q_i, d, Z_j) = (q_i, e), \quad 1 \leq i \leq n, 1 \leq j \leq p.$$

[*Note*: The move $\delta_{np}(q_i, u, Z_j) = (q_i, Z_jZ_p)$ slips the symbol Z_p under the top stack symbol Z_j, so the stack contents are always in $\Gamma_p Z_p^* \cup \{e\}$.]

Theorem 1. *For every pair of positive integers n and p, M_{np} is a realtime deterministic PDA with n states, p stack symbols, and 4np moves, which accepts by empty stack a language L_{np} for which every CFG needs at least $n^2p \oplus 1$ nonterminals.*

[*Note*: The number of moves of a deterministic PDA is just the number of triples on which the next-move function δ is defined.]

Proof. Obviously, M_{np} is a realtime deterministic PDA with the required number of states, stack symbols, and moves. Let L_{np} be the language accepted by M_{np}, and let $G = (N, \Sigma_{np}, P, S)$ be a CFG for L_{np}. By Ogden's Lemma [9], there is an integer m such that if w is in L_{np} and if m or more distinct positions in w are designated as distinguished, then $w = w_1w_2w_3w_4w_5$, where

(i) either w_1, w_2, w_3 each contain a distinguished position or w_3, w_4, w_5 each contain a distinguished position,

(ii) $w_2w_3w_4$ contains at most m distinguished positions, and

(iii) there is a nonterminal A in G such that

$$S \Rightarrow^* w_1Aw_5 \Rightarrow^* w_1w_2Aw_4w_5 \Rightarrow^* w_1w_2w_3w_4w_5.$$

Let

$$w = w_{ijk} = s_{ij}u^m r_{ij}s_{kp}d^m r_{kp}d, \quad 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n.$$

Then w is in L_{np}, so $w = w_1w_2w_3w_4w_5$ as in Ogden's Lemma, where every occurrence of u and d is distinguished except for the last d. Thus,

$$S \Rightarrow^* w_1A_{ijk}w_5 \Rightarrow^* w_1w_2A_{ijk}w_4w_5 \Rightarrow^* w_1w_2w_3w_4w_5 = w_{ijk}$$

for some nonterminal A_{ijk}. To complete the proof, it suffices to show that all of the nonterminals A_{ijk} are distinct, and that each is distinct from S if $n > 1$, for then G has at least $n^2p \oplus 1$ nonterminals.

By (iii), both $w_1w_3w_5$ and $w_1w_2w_3w_4w_5$ are generated by S and hence are in L_{np}, so w_2w_4 has the same number of occurrences of u as of d, and so by (i) it has at least one occurrence of each. But by (ii), $w_2w_3w_4$ has at most m distinguished occurrences of u and d in toto, so w_1 and w_5 must contain a distinguished occurrence of u and d

respectively. Thus,

$$(w_1, w_2w_3w_4, w_5) = (s_{ij}u^{t_1}, u^{m-t_1}r_{ij}s_{kp}d^{m-t_2}, d^{t_2}r_{kp}d) \tag{3}$$

for some positive integers t_1 and t_2.

Now suppose $A_{ijk} = A_{i'j'k'}$. Then $w_{i'j'k'}$ has a primed formula analogous to (3), and

$$S \Rightarrow^* w_1A_{ijk}w_5 = w_1A_{i'j'k'}w_5 \Rightarrow^* w_1w_2'w_3'w_4'w_5 = s_{ij}u^{t_1}u^{m-t_1'}r_{i'j'}s_{k'p}d^{m-t_2'}d^{t_2}r_{kp}d,$$

so the latter string is in L_{np}. But this is only possible if $i = i'$, $j = j'$, and $k = k'$. Hence, all of the variables A_{ijk} are distinct.

Finally, suppose $n > 1$. If $S = A_{ijk}$ for some $1 \leq i \leq n$, $1 \leq j \leq p$, $1 \leq k \leq n$, then choose $k' \neq k$ in the range $1 \leq k' \leq n$, and observe that $A_{ijk} = S \Rightarrow^* s_{k'p}d$, since the latter string is in L_{np}. Hence, by (3),

$$S \Rightarrow^* s_{ij}u^{t_1}A_{ijk}d^{t_2}r_{kp}d \Rightarrow^* s_{ij}u^{t_1}s_{k'p}dd^{t_2}r_{kp}d.$$

But since $k \neq k'$, the latter string is not in L_{np} even though it is generated by S. This is a contradiction. Hence, S is distinct from all of the A_{ijk} when $n > 1$.

The number of moves of M_{np} is $O(np)$, as is the length of a description of M_{np} in any reasonable coding. [This assumes that each coding alphabet contains the input alphabet Σ_{np}. If the $O(np)$ input symbols must be coded into a single alphabet of fixed size, then the length of a description of M_{np} would be $O(np \log(np))$.] On the other hand, any CFG for the corresponding language L_{np} must have at least $O(n^2p)$ nonterminals by Theorem 1, and hence must have size at least $O(n^2p)$, whether size is measured by the number of productions or by the length of a description in any of the usual coding schemes. Hence, any CFG equivalent to M_{np} is at least $O(n)$ times larger than M_{np}. Thus, for large n, L_{np} is an example of a context-free language which can be described much more concisely by a PDA than by a CFG.

In addition, it follows from Theorem 1 that the triple construction produces a minimal CFG for each PDA M_{np}, where a CFG is *minimal* if no equivalent CFG has fewer nonterminals. Therefore, while the grammars produced by the triple construction are in general extravagantly large, the numbers of nonterminals in them are not, for there is no construction that always produces fewer nonterminals.

3. Behavior of related conversion techniques

In this section, we investigate the behavior of one commonly used construction for intersecting a context-free language with a regular set. We also investigate the performance of the triple construction on one-turn PDAs.

There are two frequently used methods for intersecting a context-free language L with a regular set R. The first one takes the state set Q_1 of a PDA accepting L and the state set Q_2 of a finite automaton accepting R. It then forms a new PDA accepting $L \cap R$ which has $Q_1 \times Q_2$ as its state set. The second construction uses a CFG G

generating L and a finite automaton M accepting R. It then produces a CFG G' generating $L \cap R$. The grammar G' has nonterminals of the form $[q, A, r]$, where q and r are states in M and A is a nonterminal of G, as well as a separate start symbol if M has more than one state. We now show that it follows from Theorem 1 that the second construction, like the triple construction, yields a minimal grammar infinitely often.

Theorem 2. *For every pair of positive integers n and p, there is a finite automaton M with n states and a CFG G with p nonterminals such that $L(G) \cap L(M)$ can be generated only by a CFG having at least $n^2p \oplus 1$ nonterminals.*

Proof. Let $M = (Q_n, \Sigma_{np}, \delta, q_1, Q_n)$, where Q_n and Σ_{np} were defined previously, and

$$\delta(q_1, s_{ij}) = q_i, \qquad \delta(q_i, r_{ij}) = q_1,$$

$$\delta(q_i, u) = q_i, \qquad \delta(q_i, d) = q_i.$$

Let $G = (\Gamma_p, \Sigma_{np}, P, Z_1)$, where P is the set of productions

$$Z_1 \to s_{ij}Z_j, \qquad Z_j \to r_{ij}Z_1,$$

$$Z_j \to uZ_jZ_p, \qquad Z_j \to d.$$

Then $L(G) \cap L(M) = L_{np}$, which has the required property by Theorem 1.

Suppose that a context-free language is considered grammatically complex if it can be defined by a PDA with n states and p stack symbols and size $O(np)$, but can only be generated by a CFG with $O(n^2p)$ nonterminals. Then the languages used in Theorem 1 are grammatically complex. But the following theorem shows that a slight modification of the PDAs in Theorem 1 produces one-turn PDAs, and hence that even languages as simple as the linear context-free languages can be grammatically complex. The lower bound on the number of nonterminals is slightly less than $n^2p \oplus 1$, but it applies to all CFGs for the linear language, and not just to linear CFGs.

Theorem 3. *For every pair of integers $n \geq 1$ and $p \geq 2$, there is a linear context-free language accepted by empty stack by a realtime deterministic one-turn PDA with n states and p stack symbols for which every CFG needs at least $n^2(p-1) \oplus 1 = (n^2p \oplus 1) - n^2$ nonterminals.*

Proof. Let M'_{np} be the PDA M_{np} with the moves $\delta_{np}(q_i, u, Z_j) = (q_i, Z_jZ_p)$ deleted for $j = p$, with $\delta_{np}(q_k, d, Z_j) = (q_k, e)$ deleted for $j \neq p$, and with $\delta_{np}(q_k, r_{kp}, Z_p)$ redefined to be (q_k, Z_p). Then during the first phase of any computation, the stack contents are in $\Gamma_{p-1}Z_p^*$ and the stack height cannot decrease. Once an input symbol s_{kp} is encountered, the stack contents remain in Z_p^* and the stack height can never again increase. Thus, M'_{np} is a one-turn PDA.

The remainder of the proof is the same as that of Theorem 1, but with j constrained to the range $1 \leq j \leq p-1$.

4. Minimal grammars with additional properties

Recently, several studies have been made on succinctness and economy of description [2, 4, 6, 8, 10, 11, 12, 13]. Many of them are of the flavor: if you take a device A that is more powerful than a device B, then the description of a set in terms of A can be more economical or succinct than the description in terms of B. In this section, we present a result that lies at the other end of the spectrum. The following theorem shows that there are infinitely many context-free languages for which an LR(0) grammar in Greibach Normal form requires no more nonterminals than does a more general CFG. In other words, there are nontrivial cases where we do not have to pay with extra nonterminals for such nice features as Greibach Normal Form, unambiguity, and the like. This contrasts with the fact that the various algorithms for converting a CFG to Greibach Normal Form [1] introduce a large number of additional nonterminals; and the fact that the increase in size needed to replace an ambiguous CFG by an equivalent unambiguous one, when this can be done at all, is recursively unbounded [12].

Theorem 4. *There are infinitely many context-free languages having minimal grammars which are* LR(0) *(hence unambiguous) and in Greibach Normal Form with no e-productions.*

Proof. Let G_{np} be the grammar obtained by applying the triple construction to the PDA M_{np}, and let G'_{np} be the grammar obtained from G_{np} by using Algorithm 2.11 in Aho and Ullman [1] to remove productions of the form $S \to A$. Since G_{np} is minimal by Theorem 1, and since G_{np} and G'_{np} have the same number of nonterminals, G'_{np} is also minimal. The other properties of G'_{np} are derived from those of M_{np}. Specifically, G'_{np} is LR(0) because M_{np} is deterministic, and G'_{np} is in Greibach Normal Form with no e-productions because M_{np} is real-time. (For a detailed explanation of why G'_{np} is LR(0), see the proof of Theorem 12.9 in Hopcroft and Ullman [7].)

5. Conclusion

The following simple argument shows that for the PDAs M_{np}, a reduction in the number of states by a factor of k will force the number of stack symbols to increase by a factor of at least k^2. Suppose that the number of states in M_{np} is reduced by a factor k to obtain an equivalent PDA M'_{np}. Let G_{np} and G'_{np} be the grammars obtained by applying the triple construction to M_{np} and M'_{np} respectively. If M'_{np} had fewer than

k^2 times as many stack symbols as M_{np}, then G'_{np} would have fewer nonterminals than G_{np}. But by Theorem 1, this is impossible since G_{np} is minimal.

Thus, a reduction in the number of states of a PDA can necessitate a larger proportionate increase in the number of stack symbols. This observation leads to the general question: is it more economical to keep the number of states or the number of stack symbols small when designing a PDA? This question deserves further investigation.

References

[1] A.V. Aho and J.D. Ullman, *The Theory of Parsing, Translation, and Compiling, Vol. 1* (Prentice-Hall, Englewood Cliffs, NJ, 1972).

[2] M.M. Geller, H.B. Hunt, III, T.G. Szymanski and J.D. Ullman, Economy of description by parsers, DPDAs and PDAs, *Theoret. Comput. Sci.* **4** (1977) 143–153.

[3] S. Ginsburg, *The Mathematical Theory of Context-Free Languages* (McGraw-Hill, New York, 1966).

[4] S. Ginsburg and N. Lynch, Size complexity in context-free grammar forms, *J. ACM* **23** (1976) 582–598.

[5] S. Ginsburg and E. H. Spanier, Finite-turn pushdown automata, *SIAM J. Control* **4** (1966) 423–434.

[6] J. Gruska, Descriptional complexity (of languages), a short survey, *Proc. Conference on Mathematical Foundations of Computer Science*, Gdansk, September 1976, Lecture Notes in Computer Science **45** (Springer, Berlin, 1976) 65–80.

[7] J.E. Hopcroft and J.D. Ullman, *Formal Languages and their Relation to Automata* (Addison-Wesley, Reading, MA, 1969).

[8] A.R. Meyer and M.J. Fischer, Economy of description by automata, grammars, and formal systems, *Proc. 12th Symposium on Switching and Automata Theory* (1971) 188–191.

[9] W. Ogden, A helpful result for proving inherent ambiguity, *Math. Systems Theory* **2** (1968) 191–194.

[10] A. Pirická, Complexity and normal forms of context-free languages, *Proc. Conference on Mathematical Foundations of Computer Science*, June 1974, Lecture Notes in Computer Science **28** (Springer, Berlin, 1974) 292–297.

[11] A. Pirická-Kelemenová, Greibach normal form complexity, *Proc. Conference on Mathematical Foundations of Computer Science*, September 1975, Lecture Notes in Computer Science **32** (Springer, Berlin, 1975) 344–350.

[12] E. Schmidt and T. Szymanski, Succinctness of descriptions of unambiguous context-free languages, *SIAM J. Comput.* **6** (1977) 547–553.

[13] L.G. Valiant, A note on the succinctness of descriptions of deterministic languages, *Information and Control* **32** (1976) 139–145.