stichting
mathematisch
centrum

MC

AFDELING INFORMATICA
(DEPARTMENT OF COMPUTER SCIENCE)

IW 70/76 DECEMBER

P.M.B. VITÁNYI

ACHIEVABLE HIGH SCORES OF ε-MOVES AND RUNNING TIMES IN DPDA COMPUTATIONS

Prepublication

2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O).

AMS(MOS) subject classification scheme (1970): 68A25, 68A20, 94A30

ACM-Computing Reviews-categories: 5.23, 5.25, 5.26, 4.1.

Achievable high scores of ε-moves and running times in DPDA computations *)

by

P.M.B. Vitányi

ABSTRACT

Large scores in the number of consecutive ε-moves a DPDA can make without entering a loop or decreasing its stack below the original stack height are investigated. The achieved scores are very near to an upper bound in the general case and are the upper bound for one-state DPDA's. Upper and lower bounds are derived for the worst case running times of accepting DPDA computations.

KEY WORDS & PHRASES: *Deterministic pushdown automata computations, Maximal number of ε-moves, Largest running times, Highest inefficiency.*

---

*) This report will be submitted for publication elsewhere

1. INTRODUCTION

Deterministic pushdown automata (*DPDA's*) accept the so-called deterministic context free languages and constitute an important device in the theory of parsing and compiling [1]. Given a DPDA acceptor for some language (the device tells us whether an input word is in the language) we can convert it to a recognizer (the device tells us whether or not the input word is in the language) by eliminating *loops*, i.e., infinite sequences of consecutive ε-*moves* (nonreading machine steps). SCHÜTZENBERGER [5] showed how one can do so. Later proofs analyzed the amount of work involved in bringing a DPDA in loop-free form, which involved giving an upper bound on the number of consecutive ε-moves a DPDA can make without entering a loop or decreasing its stack below its original height.
In [3, Lemma 12.1] it is shown that for a DPDA with n_1 states, n_2 stack symbols and ℓ the maximal lenght of a string with which the topmost stack symbol can be replaced in a single move, $n_1(n_2+1)^{n_1 n_2 \ell}$ is such an upper bound. In [1, Algorithm 2.16] the slightly better upper bound of $n_1(n_2^{n_1 n_2 \ell}-n_2)/(n_2-1)$ (or n_1 if $n_2 = 1$) is given. Using a different approach, in [4] the upper bound of $(\ell^{n_1 n_2}-1)/(\ell-1)$ (or $n_1 n_2$ if $\ell = 1$) is given. This latter bound is achieved by using techniques already appearing in [6], where it is proven that we can test for looping configurations in DPDA's in time linear in the parameters. Hence the problem of determining the maximal number of consecutive ε-moves a DPDA can make without looping or decreasing the stack below the original stack height merits interest primarily as a combinatorial problem. In the present note we investigate how high a score a DPDA can actually achieve. It is shown that for DPDA's which read input

$$((\ell-1)^{n_1}\ell^{(n_1+1)(n_2-2)}-\ell^{n_2-2})/((\ell-1)\ell^{n_2-2}-1)$$

is an achievable lower bound on this maximal number of ε-moves for $n_1 \geq 1$, $n_2 \geq 3$ and $\ell \geq 2$. For $n_1 = 1$ (one-state DPDA's) this is also an upper bound, and the above score is very near to an upper bound in the general case. Finally, we give upper and lower bounds on the worst case running times of DPDA computations in which all input is read.

2. RESULTS.

Definitions and terminology closely follow [1]. We assume familiarity with the way of looking at DPDA computations of [4] and [6].
Let M be a DPDA with n_1, n_2 and ℓ as in the introduction. Denote the maximal number of consecutive ε-moves a DPDA M with these parameters can make, without entering a loop or decreasing its stack below the original stack height, by $f(n_1,n_2,\ell)$ where we assume that there is at least one (state, stack symbol) pair for which M reads input. When we do not impose the latter requirement we denote the corresponding function by $f'(n_1,n_2,\ell)$ and observe that DPDA's with parameters n_1,n_2,ℓ which score between $f(n_1,n_2,\ell)$ and $f'(n_1,n_2,\ell)$ accept the language $\emptyset$ or $\{\varepsilon\}$.

THEOREM 1. $f(n_1,n_2,\ell) \geq g(n_1,n_2,\ell)$

where

$$g(n_1,n_2,\ell) = ((\ell-1)^{n_1}\ell^{(n_1+1)(n_2-2)} - \ell^{n_2-2})/((\ell-1)\ell^{n_2-2}-1)$$

for $n_1 \geq 1$, $n_2 \geq 3$ *and* $\ell \geq 1$.

PROOF. Let the state set of M be $\varphi = \{1,2,\ldots,n_1\}$ and let the set of stack symbols be $\Gamma = \{1,2,\ldots,n_2\}$. The following *canonical scheme* (see [4]) for (1,1) with respect to M will achieve the claimed lower bound. The canonical scheme is the context free grammar

$$G = (\varphi \times \Gamma \cup \varphi,\ \varphi \cup \{(1,n_2)\},\ (1,1),\ P)$$

where P is defined by:

(i) $(1,1) \xrightarrow{\varepsilon} (1,2)(1,2)\ldots(1,2)(1,n_2)$

(ii) $(i,n_2-1) \xrightarrow{\varepsilon} (i+1,1)(i+1,1)\ldots(i+1,1)(i+1,n_2)$
for $1 \leq i < n_1$,

(iii) $(i,j) \xrightarrow{\varepsilon} (i,j+1)(i,j+1)\ldots(i,j+1)$
for $1 \leq i \leq n_1$, $1 \leq j < n_2-1$ and $(i,j) \neq (1,1)$,

(iv) $(i,n_2) \xrightarrow{\varepsilon} i-1$ for $1 < i \leq n_1$,

(v) $(n_1,n_2-1) \xrightarrow{\varepsilon} n_1$,

Only reading sequences can increase the height of the stack and then by not more than $n_1n_2(\ell-1)$. Hence if M accepts a word $a_1a_2\ldots a_n$ the total number of symbols pushed on the stack (by sequences) is less than $n\, n_1n_2(\ell-1)$ and therefore the total running time is less than $n(n_1n_2(\ell-1)+1)\, f(n_1,n_2,\ell)$, i.e., the combined length of popping and reading sequences. □

It is clear that there is a trade-off between the fact that anything is stacked in a read sequence and wether a large sequence in the order of $f(n_1,n_2,\ell)$ is reached.

Let T(n) be the longest running time of a computation by a DPDA M with parameters n_1,n_2,ℓ up to reading the n th letter of an input $a_1a_2\ldots a_n$.

THEOREM 5.

$$(2n-1)g(n_1,n_2,\ell) \le T(n) \le (n-1)(\ell^{n_1n_2}-1)/(\ell-1).$$

PROOF. $(2n-1)g(n_1,n_2,\ell) \le T(n)$. The lower bound on T(n) is achieved by adding, in the proof of theorem 1, the read move $(1,n_2) \xrightarrow{a} (1,2)(1,2)\ldots(1,2)(1,1)$ for each input letter a.

$T(n) \le (n-1)(\ell^{n_1n_2}-1)/(\ell-1)$. In the proof of Lemma 4 we introduced sequences of ε-moves. If, starting from some starting (state, stack symbol) pair the sequence of ε-moves leads to a read move and the stack height has been increased by $x(\ell-1)$ then a popping or reading sequence has a length of less than $(\ell^{n_1n_2-x}-1)/(\ell-1)$ since there are at least x (state, stack symbol) pairs which lead to a premature read move. Hence the total number of ε-moves up to reading the n-th letter of input is less than $(n-1)((\ell-1)x+1)(\ell^{n_1n_2-x}-1)/(\ell-1)$ which is largest for x = 0. □

Another, easier, subject is how large a stack a DPDA can accumulate up to reading the n-th letter of input. It is easy to show that

$$(n-1)n_1n_2(\ell-1) + (n_1n_2-2)(\ell-1) + 1$$

can be reached, which seems to be the maximum. Notice, that the machine cannot achieve both a large score in stack height and running time.

where the lengths of the righthand sides of rules (i)-(iii) is ℓ.

The unique leftmost derivation of the unique terminal word $i_1 i_2 \ldots i_k(1,n_2)$ produced by G represents the sequence of ε-moves of the corresponding DPDA M starting in state 1 with stack symbol 1 as its stack contents and ending in state 1 with stack symbol n_2 as its stack contents, i.e., the only (state, stack symbol) pair which reads input. Every direct production of the leftmost derivation corresponds to an ε-move of M and vice-versa. For an intermediate sentential form

$$i_1 i_2 \ldots i_m(i_{m+1},j_{m+1})(i_{m+2},j_{m+2})\ldots(i_s,j_s)(1,n_2)$$

$i_1, i_2 \ldots, i_m$ are the return states (states resulting from) of all popmoves executed up to the present stage (and in historical order from left to right); i_{m+1} is the present state of the finite control and $j_{m+1} j_{m+2} \ldots j_s\ n_2$ is the present stack contents. i_{m+p}, $2 \leq p \leq s-m$, represents the state of the finite control when it accesses for the first time stack symbol j_{m+p}. (i)-(iii) correspond to pushmoves and (iv)-(v) to popmoves. The constraints on such a context-free grammar representing a nonlooping ε-computation are therefore:

(a) There are no circular nonterminals.

(b) There is a unique production for all nonterminals.

(c) If $(i,j) \overset{\varepsilon}{\rightarrow} i' \in P$ (a popmove) then (i,j) can only occur in a righthand side followed by (i'',j') for some $j' \in \Gamma$ if $i'' = i'$.

(a) and (b) garanty determinacy and absence of loops, while (c) garanties that the nonterminal right of a nonterminal which is rewritten according to (iv) or (v) will indeed represent by its first coordinate the return state of the executed pop. We display the derivation tree of the unique derivation in G in fig. 1 where it is clear that identically labelled nodes are the roots of identical subtrees in the derivation tree. The internal nodes in the tree correspond to ε-moves of M and counting their number yields $g(n_1,n_2,\ell)$. □

COROLLARY 2. *If we do not insist on* M *having a (state, stack symbol) pair for a read move we achieve a score of consecutive* ε*-moves of*

$$g'(n_1,n_2,\ell) = (\ell^2/\ell-1)(g(n_1,n_2,\ell)-1) + \ell + 1,$$

in the obvious way.

COROLLARY 3. *For one-state* DPDA's *it is easily verified that* $g(1,n_2,\ell)$ *(and* $g'(1,n_2,\ell)$*) are also upper bounds, and indeed* $g'(1,n_2,\ell)$ *is equal to the bound in* [4] *for* $n_1 = 1$. *Therefore,* $f(1,n_2,\ell) = g(1,n_2,\ell)$ *for* $n_2 \geq 3$ *and* $\ell \geq 2$.

For lower values of the parameters n_1,n_2,ℓ we can similarly to theorem 1 derive $f(n_1,n_2,\ell) \geq g(n_1,n_2,\ell)$ where for $n_2 < 3$ or $\ell < 2$ $g(n_1,n_2,\ell)$ is given by:

(i) $g(1,2,2) = 1$,

(ii) $g(2,2,\ell) = 2\ell$ for $\ell \geq 2$,

(iii) $g(n_1,1,\ell) = n_1-1$,

(iv) $g(n_1,n_2,1) = n_1n_2-1$,

(v) $g(n_1,2,2) = 4n_1-4$ for $n_1 \geq 2$,

(vi) $g(n_1,2,\ell) = 4((\ell-1)^{n_1}-1)/(\ell-2) - 2(\ell-1)^{n_1-1} - 2$

for $\ell \geq 3$ and $n_1 \geq 2$;

as we leave for the reader to verify, from fig. 2.

That $g(n_1,n_2,\ell)$ is very near an uper bound on f is argued as follows. Since M needs at least one read move and n_1 popmoves to access all elements of $\varphi \times \Gamma$ (necessary for a balanced derivation tree), the number of push-moves is less than $n_1(n_2-1)$ and $\ell^{n_1(n_2-1)}$ is surely an upper bound on the number of ε-moves. More detailed reasoning gets f closer to g, and it seems very likely that $f = g$ (and $f' = g'$).

We now take a look at the running time of DPDA computations. The following fact presumably belongs to the folklore in the field and is implicit in [2].

LEMMA 4. DPDA's *accept in linear time.*

PROOF. We can distinguish sequences of consecutive ε-moves, which from start to finish do not decrement the stack height below its starting height except possibly at the last move, in:

(i) *popping* sequences, i.e., the last move decrements the stack height to 1 below its starting height.

(ii) *reading* sequences, i.e., those which end with a read move.

(iii) *looping* sequences.

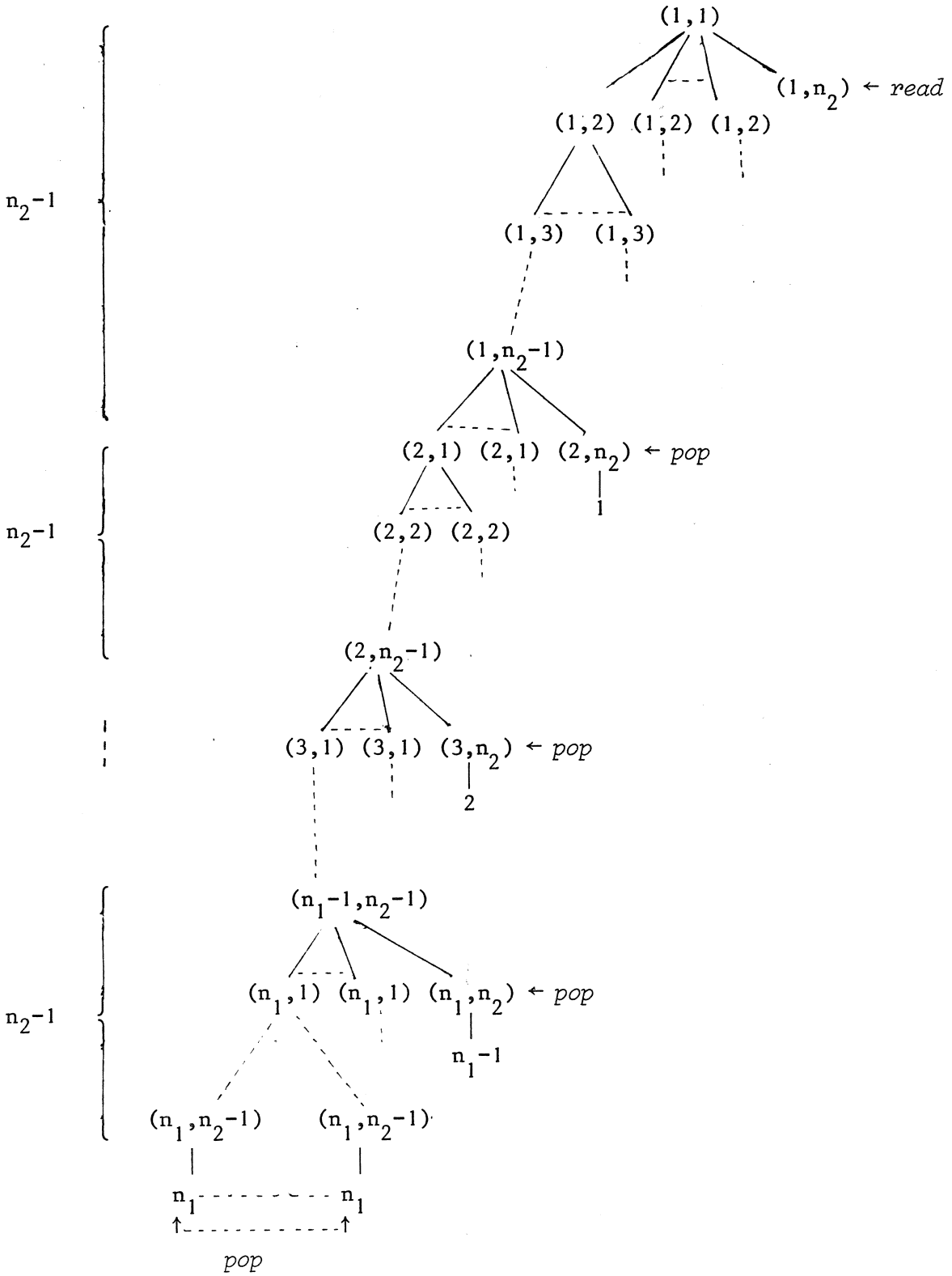
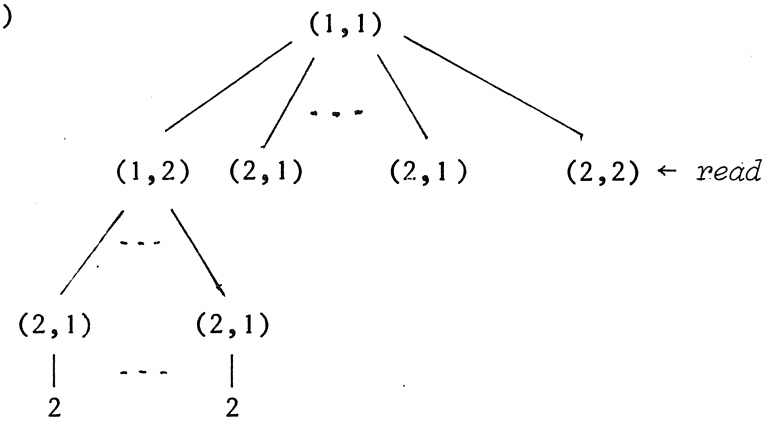
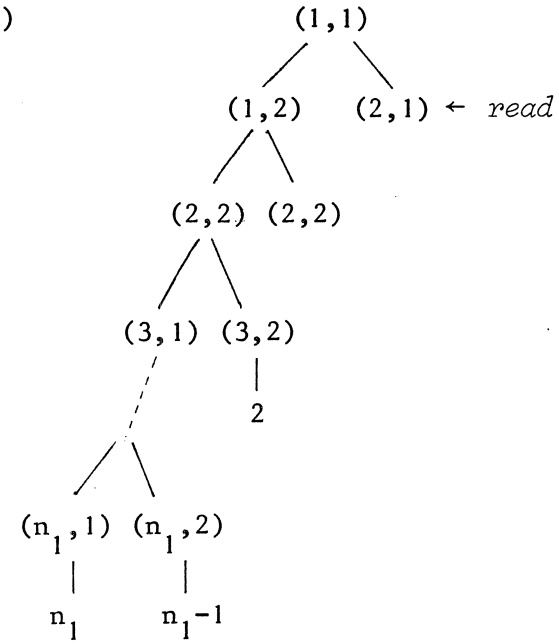

(1,1)
(1,n2) ← read
(1,2) (1,2) (1,2)
n2-1
(1,3) (1,3)
(1,n2-1)
(2,1) (2,1) (2,n2) ← pop
1
n2-1
(2,2) (2,2)
(2,n2-1)
(3,1) (3,1) (3,n2) ← pop
2
(n1-1,n2-1)
(n1,1) (n1,1) (n1,n2) ← pop
n2-1
n1-1
(n1,n2-1) (n1,n2-1)
n1 n1
pop

Fig.1.

(ii)


(1,1)
(1,2) (2,1) (2,1) (2,2) ← read
(2,1) (2,1)
2 2

(iv)


(1,1)
(1,2) (2,1) ← read
(2,2) (2,2)
(3,1) (3,2)
2
(n1,1) (n1,2)
n1 n1-1

(v)

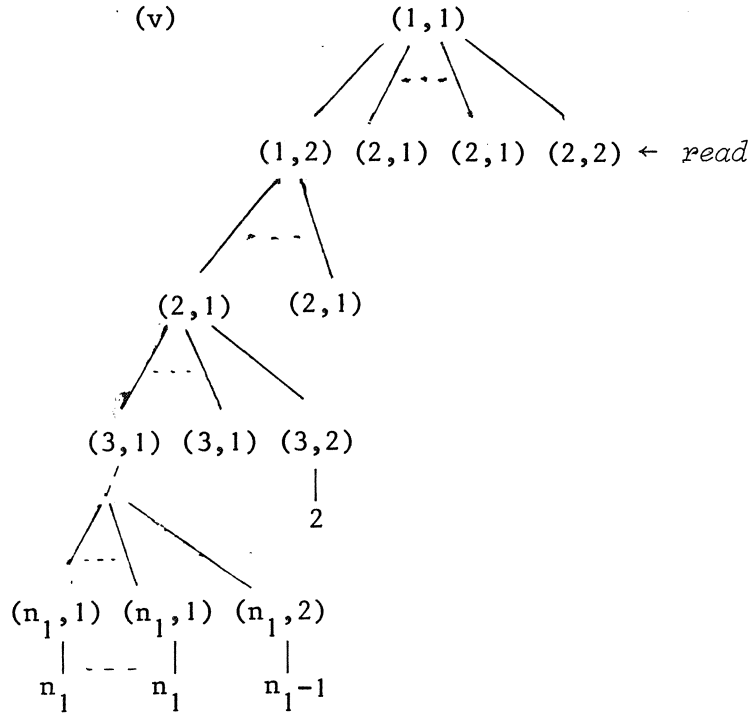

(1,1)
(1,2) (2,1) (2,1) (2,2) ← read
(2,1) (2,1)
(3,1) (3,1) (3,2)
2
(n1,1) (n1,1) (n1,2)
n1 n1 n1-1

Fig. 2.

REFERENCES.

1. AHO A.V. and J.D. ULLMAN, *The Theory of Parsing, Translation and Compiling*, vols 1 & 2. Prentice Hall (1972)

2. GINSBURG S. and S.A. GREIBACH, *Deterministic contextfree languages*, Inf. and Control 9 (1966) 620-648.

3. HOFCROFT J.E. and J.D. ULLMAN, *Formal Languages and their Relation to Automata*, Addison-Wesley (1969).

4. LEEUWEN J. van and C. SMITH, *An improved bound for looping configurations in deterministic PDA's*, Inf. Proc. Letters 3 (1974) 22-24.

5. SCHÜTZENBERGER M.P., *On contextfree languages and pushdown automata*, Inf. and Control 6 (1963) 246-264.

6. VALIANT L.G., *Decision procedures for families of deterministic pushdown automata*, Ph.D. Thesis, Dept. Comp. Sci., Univ. of Warwick, England (1973).