Revue française d'automatique, informatique, recherche opérationnelle. Informatique théorique

S. A. Greibach

A note on the recognition of one counter languages

Revue française d'automatique, informatique, recherche opérationnelle. Informatique théorique, tome 9, n° R2 (1975), p. 5-12.

<http://www.numdam.org/item?id=ITA_1975__9_2_5_0>

R.A.I.R.O.
(9e année, août 1975, R-2, p. 5-12)

A NOTE ON THE RECOGNITION OF ONE COUNTER LANGUAGES (*)

par S. A. GREIBACH [1]

Communicated by R. V. BOOK

Abstract. — *Every on-line one counter language can be accepted by a deterministic Turing machine in time n^2. The family of deterministic on-line one counter languages is properly contained in the family of realtime pushdown store acceptor languages. Any off-line nondeterministic one counter machine accepts in time n^3 and space n^2.*

Various results have been established for the complexity of recognition of both on-line and off-line pushdown store languages. For example, it is known that context-free languages (on-line one pushdown store languages) can be recognized by determinsitic Turing machines in time n^3 [1] or in space $(\log n)^2$ [2]. It is not known if either of these results is optimal. A context-free language is known whose time or space complexity is the realizable least upper bound on time or space complexity for the whole family of context-free languages [3]. For some special cases, better results are known; the family of linear context-free languages is recognizable by deterministic Turing machines in time n^2 [4]. Off-line one pda languages can be recognized in space n^2 and time n^4 [5].

In this note we examine briefly the special case of one counter languages, both on-line and off-line. The main results are that every on-line one counter language can be accepted by a deterministic Turing machine in time n^2, and any off-line nondeterministic one counter machine accepts in time n^3 and space n^2. To prove the off-line result, we show that context-free grammars generate in linear time and hence on-line pdas always accept in linear time.

(*) The research in this paper was supported in part by the National Science Foundation under Grant GJ-803.

(1) Department of System Science, University of California, Los Angeles.

The reader is referred to [6] for formal definitions of counters and Turing machines. We assume that our machines accept by empty counter and final state. On-line machines have a one-way input tape reading from left-to-right, while off-line machines are assumed to have a two-way read-only input tape with endmarkers on both sides. A deterministic machine *accepts in time* $T(n)$ if each input w accepted by M is accepted within $T(|w|)$ steps (1). A nondeterministic machine M *accepts in time* $T(n)$ if for each input w accepted by M there is a computation of M on w which accepts in at most $T(|w|)$ steps.

First we establish our result for on-line one counter languages.

Theorem 1. Every on-line one counter language can be accepted by a deterministic Turing machine in time n^2.

Proof

First, if L is an on-line one counter language, we can assume that L is accepted by a nondeterministic on-line one-counter machine M which advances its input tape each unit of time and accepts with the counter empty [7], [8]. Thus M certainly accepts in time n; in an accepting computation on w, the counter never exceeds $|w|/2$.

We now describe a deterministic Turing machine to accept L. Let $*$ be a new symbol and assume that we have an encoding E of subsets of the state set K of M. We start at time 0 with entry $*E(\{q_0\})*$ on the Turing machine tape where q_0 is the initial state of M. Suppose at time t we have $*E(S_0)* \ldots *E(S_i)* \ldots *E(S_{m_t})*$ on the working tape and input a (the $t+1-st$ input symbol) to M. We go through the m_t entries one by one. Entry $E(S_i)$ is replaced by $E(T_i)$ where T_i contains all and only those states $q \varepsilon K$ such that for some $l \varepsilon \{1, 0, -1\}$ and p in S_{i-l}, M on input a has the option of transferring to q adding l to the counter. For $i = 0$ we do not consider $l = 1$ and for $i = m_t$ we do not consider $l = -1$. Finally, if for some p in S_{m_t} and input a, M has the option of transferring to state q adding 1 to the counter, we let T_{m_t+1} be the set of all such states q and add $E(T_{m_t+1})*$ at the end of the tape. Thus we see that at time t entry $E(S_i)$ encodes all states M could be in with counter contents i after reading the first t input symbols. So w is accepted if and only if at time $t = |w|$, the set S_0 contains an accepting state. Clearly $m_t \leqslant t$ and $*E(S_0)* \ldots *E(S_{m_t})*$ can be updated in time cm_t for some constant c. Thus w is accepted or rejected in time

$$c \sum_{t=0}^{|w|} (t+1) = c \frac{(|w|+1)(|w|+2)}{2}$$

(1) For a word w, $|w|$ is the length of w.

which is less than $c\,|w|^2$ for $|w| > 3$. Hence an input of length n is accepted or rejected in time proportional to n^2. ∎

The bound of n^2 is not at all a tight one. To the best of the author's knowledge, it is not known whether there are on-line one counter languages not recognizable in realtime by a deterministic multitape Turing machine. It seems reasonable to conjecture that linear time might suffice.

Using the techniques of Theorem 1 we can establish similar results for time bounded multicounter languages. First we establish a result for multihead finite state machines as Corollary 1, and then use known connections between these machines and polynomially time bounded multicounter machines to yield Corollary 2.

Consider a k-head finite state acceptor; see [10] for precise definitions and details. Such a machine must accept in time $c_1 n^k$ for some constant c_1. Let us extend the construction of Theorem 1 to use a Turing machine, this time with a k-dimensional storage tape. In entry $(i_1, \ldots, i_k)$ we place an encoding not only of the states the machine could be in with head j on square i_j but also the symbol on square i_j, $1 \leqslant j \leqslant k$. An update cycle takes time at most $c_2 n^k$ for some constant c_2; at the same time a counter can count up to $c_1 n^k$ update cycles. Thus the machine needs at most time $c_3 n^{2k}$ and tape n^k for an appropriate constant c_3. Hence an on-line Turing machine with one dimensional storage takes time proportional to n^{3k}.

Corollary 1. A language accepted by a k-head finite state machine can be accepted by a deterministic Turing machine in time n^{3k}.

If a language can be accepted by an off-line nondeterministic machine with r counters in time n^k, it can be accepted by a $(rk + 1)$ – head finite state acceptor [6], [10], [11]. Hence we have :

Corollary 2. A language accepted by an off-line nondeterministic r-counter machine in time n^k can be accepted by a deterministic Turing machine in time $n^{3(k+1)}$.

In the deterministic one counter case we can do better than Theorem 1. Given a deterministic on-line pushdown store acceptor (pda), we can construct an equivalent one which advances its input tape whenever it is not erasing the store [9]; this construction takes a counter into a counter. But if a deterministic on-line one-counter machine ever performs more subtractions than it has states without advancing its input tape, it will erase the whole counter. Further for any pair of states q and q' there are integers $m(q, q')$ and $n(q, q')$ such that it will start in q and complete emptying the counter in q' if and only if the counter has contents $x \equiv m(q, q') \pmod{n(q, q')}$. Thus a simulating pda could keep track of the mod $n(q, q')$ congruence of x in its finite state control and instead of erasing put down a new « Begin » symbol. Hence it would operate in realtime (assuming it accepts by final state rather than empty store and final state.) Thus every deterministic one counter language is a realtime pda language; the converse is obviously false as the language $\{ wcw^R \mid w \in \{ a, b \}^* \}$ shows.

Corollary 3. Any deterministic on-line one-counter language can be accepted in realtime by a pda.

Now off-line one counter languages are a special case of off-line pda languages and hence can be accepted by deterministic multitape Turing machines in time n^4 [5]. We shall prove (Theorem 3) something stronger, namely that an off-line one counter machine always accepts in time n^3 and space n^2. This will follow from a result on derivation lengths in context-free grammars : any context-free grammar produces words in linear time in the sense that for any context-free grammar G there is a constant k such that if G generates a word w then some derivation of w takes at most $k\,|w|$ steps. Applied to pushdown store acceptors this says that any on-line pda in fact accepts in linear time.

Let us use the following notation for context-free grammars. In a context-free grammar $G = (V, \Sigma, P, S)$, V is a finite vocabulary, $\Sigma \subseteq V$ is the *terminal* vocabulary, $S \in V - \Sigma$ the *start* symbol, and $P \subseteq (V - \Sigma) \times V^*$ a finite set of *productions* or *rules*. If $(Z, y) \in P$, usually written $Z \to y$, and $u, v \in V^*$, we write $uZv \Rightarrow uyv$; if $u \in \Sigma^*$, we can also write $uZv \overset{L}{\Rightarrow} uyv$. Then $\overset{*}{\Rightarrow}$ ($\overset{L*}{\Rightarrow}$) is the transitive reflexive closure of $\Rightarrow$ ($\overset{L}{\Rightarrow}$). The language generated by G is $L(G) = \{ w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w \}$. A derivation $Z \overset{L*}{\Rightarrow} w$ is called *left-to-right*.

For a context-free grammar $G = (V, \Sigma, P, S)$, let $v_G = \#\,(V - \Sigma)$ and $k_G = \mathrm{Max}\,\{ |y| \mid \exists Z (Z, y) \in P \}$ (2). In a derivation $\gamma : y_0 \Rightarrow y_1 \Rightarrow \ldots \Rightarrow y_n$ let $n(\gamma) = n$ and $l(\gamma) = \mathrm{Max}\,\{ |y_i| \mid 1 \leqslant i \leqslant n \}$. For $Z \in V - \Sigma$, $w \in V^*$, if $Z \overset{*}{\Rightarrow} W$, let $f_G(Z, w) = \mathrm{Min}\,\{ n(\gamma) \mid \gamma : Z \overset{*}{\Rightarrow} w \}$ and if $w \in \Sigma^*$, let $h_G(Z, w)$ be the least $l(\gamma)$ for any left-to-right derivation $\gamma : Z \overset{L*}{\Rightarrow} w$.

Theorem 2. Let $G = (V, \Sigma, P, S)$ be a context-free grammar. Let $m_0 = k_G^{v_G}$, and $m_1 = (1 + (k_G - 1)m_0)$. For $Z \in V - \Sigma$ and $w \in V^*$, if $Z \overset{*}{\Rightarrow} w$, then

$$f_G(Z, w) \leqslant \begin{cases} m_0 & w = e\ (^3) \\ (v_G - 1)m_1 & w \in V - \Sigma \\ v_G m_1 & w \in \Sigma \\ (v_G + k_G v_G) m_1\,|w| & |w| \geqslant 2 \end{cases}$$

(2) For a finite set A, $\#\,(A)$ is the number of members of A.

(3) We use the symnol e for the empty tape; note that $|w| = 0$ if and only if $w = e$.

and if $w \in \Sigma^*$

$$h_G(Z, w) \leqslant \begin{cases} (v_G - 1)(k_G - 1) + 1 & w = e \\ [(2v_G - 1)(k_G - 1) + 1]\,|w| & |w| \geqslant 1 \end{cases}$$

Proof

Call a node in a derivation tree *expanding* if it has two sons each of which has descendent leaves not labeled by the empty string. We proceed by induction on $E(\gamma)$, the number of expanding nodes in a derivation tree of derivation $\gamma : Z \overset{*}{\Rightarrow} w$, to show that

$$f_G(Z, w) \leqslant (v_G + k_G v_G) m_1 \operatorname{Max}(1, E(\gamma))$$

and if $w \in \Sigma^*$, then

$$h_G(Z, w) \leqslant \begin{cases} (v_G - 1)(k_G - 1) + 1 & w = e \\ [(2v_G - 1)(k_G - 1) + 1]\,|w|, & w \neq e \end{cases}$$

The result follows from the proof of the special case $E(\gamma) = 0$, and the fact that $E(\gamma) \leqslant |w| - 1$ for $w \neq e$.

First consider $E(\gamma) = 0$. There are two cases, $w = e$ and $w \in V$. In the first case consider the tree corresponding to a shortest derivation for $Z \overset{*}{\Rightarrow} e$. No nonterminal can appear twice in any path in this tree. Hence each path in the tree has length at most v_G. In the corresponding left-to-right derivation $\gamma : Z \overset{L*}{\Rightarrow} e$, $n(\gamma) \leqslant m_0 = k_G^{v_g}$ and

$$l(\gamma) \leqslant k_G + (k_G - 1)(v_G - 2) = (v_G - 1)(k_G - 1) + 1.$$

Now suppose $w = A \in V$. Consider the smallest derivation tree for A from Z. The path from Z to A has length at most v_G ($v_G - 1$ if $A \in V - \Sigma$) and all the brothers of nodes on that path generate the empty string. Hence there is a corresponding derivation $\gamma : Z \overset{*}{\Rightarrow} A$, which is left-to-right if $A \in \Sigma$, such that $n(\gamma) \leqslant v_G(1 + (k_G - 1)m_0) = v_G m_1$, if $A \in \Sigma$ and

$$n(\gamma) \leqslant (v_G - 1)(1 + (k_G - 1)m_0) = (v_G - 1)m_1$$

if $A \in V - \Sigma$.

If $A \in \Sigma$, then in the worst case the left-to-right derivation might have an intermediate string y_i containing $v_G(k_G - 1)$ symbols for the path from Z to A of length v_G plus $(v_G - 1)(k_G - 1) + 1$ symbols for the erasing of a left brother of A. Hence $h_G(Z, A) \leqslant l(\gamma) \leqslant (2v_G - 1)(k_G - 1) + 1$.

Now suppose that $E \geqslant 1$, that we have shown the result for all $E' < E$, and that we have for $Z \overset{*}{\Rightarrow} w$ a shortest derivation $\gamma : Z \overset{*}{\Rightarrow} w$, such that $n(\gamma) = f_G(Z, w)$ and $E(\gamma) = E$. We can divide γ into :

$$\begin{aligned}
&\gamma_1 : Z \overset{*}{\Rightarrow} A \\
&\gamma_2 : A \Rightarrow x_1 Y_1 x_2 Y_2 \dots x_l Y_l x_{l+1} \\
&\gamma_i' : x_i \overset{*}{\Rightarrow} e, \qquad 1 \leqslant i \leqslant l + 1 \\
&\gamma_i : Y_i \overset{*}{\Rightarrow} w_i, \qquad 1 \leqslant i \leqslant l
\end{aligned}$$

where $w = w_1 \dots w_l$, $w_i \neq e$, A, $Y_i \in V$, $1 \leqslant i \leqslant l$, and $l \geqslant 2$ and $l + |x_1 \dots x_{l+1}| \leqslant k_G$. (It is possible that $Y_i = w_i$ for all but two values of i.) Thus A labels the first expanding node. Hence $E(\gamma_1') + \dots + E(\gamma_l') = E(\gamma) - 1$.

By the previous results for $E(\gamma) = 0$ and the induction hypothesis :

$$\begin{aligned}
&n(\gamma_1) \leqslant (v_G - 1)m_1 \\
&n(\gamma_2) = 1 \\
&n(\gamma_i') \leqslant |x_i|\, m_0, \qquad 1 \leqslant i \leqslant l + 1
\end{aligned}$$

and

$$n(\gamma_1'') \leqslant \begin{cases} v_G m_1 & \text{if} \quad E(\gamma_i'') = 0 \\ (v_G + k_G v_G) m_1 E(\gamma_i'') & \text{if} \quad E(\gamma_i'') \geqslant 1 \end{cases}$$

Let r be the number of γ_i'' with $E(\gamma_i'') = 0$. Then

$$n(\gamma_1'') + \dots + n(\gamma_l'') \leqslant r v_G m_1 + (v_G + k_G v_G) m_1 (E(\gamma) - 1)$$

and

$$\begin{aligned}
n(\gamma) &\leqslant (v_G - 1)m_1 + 1 + (k_G - l)m_0 + r v_G m_1 + (v_G + k_G v_G) m_1 (E(\gamma) - 1) \\
&\leqslant v_G m_1 + (k_G - l)m_0 + l v_G m_1 + (v_G + k_G v_G) m_1 (E(\gamma) - 1) \\
&\leqslant v_G m_1 + k_G v_G m_1 + (v_G + k_G v_G) m_1 (E(\gamma) - 1) \\
&= (v_G + k_G v_G) m_1 E(\gamma).
\end{aligned}$$

If $w \in \Sigma^*$, consider the corresponding left-to-right derivation

$$\gamma : Z \overset{L*}{\Rightarrow} w.$$

Let $s_1 = (v_G - 2)(k_G - 1)$, $s_2 = (v_G - 1)(k_G - 1) + 1$,
and

$$g = s_1 + s_2 + 2(k_G - 1) = (2v_G - 1)(k_G - 1) + 1.$$

In the worst case we have $Z \overset{L*}{\Rightarrow} A\alpha$ for $\alpha \neq e$; by our previous reasoning for the case $E(\gamma) = 0$, $|\alpha| \leqslant s_1$. Then we have $A\alpha \overset{L}{\Rightarrow} x_1 Y_1 x_2 \dots x_l Y_l x_{l+1}\alpha$. Recall $l \geqslant 2$ and $|x_1 \dots x_{l+1}| \leqslant k_G - l$. Now while we expand each x_i the intermediate strings are certainly bounded in length by

$$\begin{aligned}
|\alpha| + |w_1 \dots w_{i-1}| + s_2 + |x_i Y_i \dots Y_l x_{l+1}| - 1 &< |w| + s_1 + s_2 + k_G - 1 \\
&\leqslant |w| + g < g\,|w|,
\end{aligned}$$

since $|w| \geqslant 2$, and $g \geqslant 2$.

By the induction hypothesis applied to $Y_i \overset{L^*}{\Rightarrow} w_i$, while we expand Y_i the strings are bounded in length by

$$\begin{aligned}|w_1 \dots w_{i-1}| + |\alpha| + g\,|w_i| + |x_{i+1} Y_{i+1} \dots x_{l+1}| \\ \leqslant |w_1 \dots w_{i-1}| + g\,|w_i| + g - 1 \leqslant 1 + g(|w| - 1) + g - 1 = g\,|w|.\end{aligned}$$

Hence

$$h_G(Z, w) \leqslant g\,|w|. \quad \blacksquare$$

In the present instance we need only a simplification of this theorem which we present as a corollary.

Corollary 1. For a context-free grammar $G = (V, \Sigma, P, S)$ there are constants c_1 and c_2, with c_2 independent of v_G, such that w is in $L(G)$ if and only of there is a left-to-right derivation $\gamma : S \overset{L^*}{\Rightarrow} w$, with $n(\gamma) \leqslant c_1$ Max $(|w|, 1)$ and $l(\gamma) \leqslant c_2 v_G$ Max $(|w|, 1)$.

Stated in terms of on-line pdas we have :

Corollary 2. Given an on-line pda M with q states, there are constants c_1 and c_2, with c_2 independent of q, such that for all inputs w, M accepts w if and only if there is an accepting computation of M on w taking at most c_1 Max $(1, |w|)$ steps in which the pushdown store word never exceeds in length $c_2 q^2$ Max $(1, |w|)$.

Proof

In the standard construction of a context-free grammar G_M for M, if M has r pushdown store symbols, then $v_{G_M} \leqslant rq^2$ and a step in a computation of M corresponds exactly to a step in a derivation of G_m (see [12]). $\blacksquare$

For off-line pdas we have the following corollary.

Corollary 3. If M is an off-line pda with k reading heads on its input tape, there is a constant c such that M accepts in space cn^{2k}.

Proof

If M has q states and acts on an input w, we can construct an on-line pda M_m with $q(\text{Max}\,(1, |w|))^k$ states which accepts the empty word if and only if M accepts w. Since M_w uses its pushdown store just as M uses its store on w, the space used by M on w is the same as that used by M_w on the empty word. $\blacksquare$

We state the next corollary as a separate theorem.

Theorem 3. If M is an off-line k-head one-counter machine, then there is a constant c such that M accepts in space cn^{2k} and time cn^{3k}.

REMARK. A deterministic k-head pda must accept in space cn^k for some constant c (or it finds itself in a loop; cf. [5] for details). Hence a deterministic off-line k-head one counter machine must accept in space cn^k and time cn^{2k} for some constant c.

BIBLIOGRAPHY

[1] D. H. Younger, Recognition and parsing of context-free languages in time n^3, *Information and Control*, (1967), 10, 189-208.

[2] P. M. Lewis, R. E. Stearns and J. Hartmanis, Memory bounds for recognition of context-free and context-sensitive languages, *IEEE Conference Record on Switching Circuit Theory and Logical Design*, Ann. Arbor, Michigan, 1965, 191-202.

[3] S. A. Greibach, The Hardest Context-free Language, *SIAM J. Computing*, (1973), 2, 304-310.

[4] T. Kasami, A note on computing time for recognition of languages generated by linear grammars, *Information and Control*, (1967), 10, 209-214.

[5] A. V. Aho, J. E. Hopcroft and J. D. Ullman, Time and tape complexity of pushdown automaton languages, *Information and Control*, (1968), 13, 186-206.

[6] P. C. Fischer, A. R. Meyer and A. L. Rosenberg, Counter machines and counter languages, *Math Systems Theory*, (1968), 2, 265-283.

[7] S. A. Greibach, Erasable context-free languages, to appear.

[8] S. Ginsburg and G. F. Rose, The equivalence of stack counter acceptors and quasi-realtime acceptors, *J. Computer System Sciences*, (1974), 8, 243-269.

[9] S. Ginsburg and S. Greibach, Deterministic context-free languages, *Information and Control*, (1966), 9, 620-648.

[10] O. H. Ibarra, On two-way multihead automata, *J. Computer System Sciences*, (1973), 7, 28-36.

[11] S. Greibach, Remarks on the complexity of nondeterministic counter languages, to appear.

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