A HARDEST LANGUAGE RECOGNIZED BY TWO-WAY NONDETERMINISTIC PUSHDOWN AUTOMATA

Wojciech RYTTER
Institute of Informatics, Warsaw University, PKiN VIII p, 00-901 Warszawa, Poland

Received 8 July 1981; revised version received 9 November 1981

Language recognition, complexity, pushdown automata

In this paper we present a language G_0 which is 'hardest' in the class 2NPDA of languages recognized by two-way nondeterministic pushdown automata with one reading head (in short 2npda's). This language is hardest in the sense that any reasonable deterministic recognition procedure for the class 2NPDA will have as its time complexity the time it takes to recognize G_0. The language G_0 is also hardest in the sense of space complexity. The language G_0 is especially attractive because of its relation to the hardest context-free language L_0 from [2]. We consider languages over some fixed alphabet V which do not contain the empty word. The special symbol '\$' does not belong to V. This symbol is used as the endmarker of all input words w. The input word w is given to the automaton in the form \$w\$. Let V^+ denote the set of all nonempty words over the alphabet V and for every language $L \subseteq V^+$ let $\sqrt[+]{L} = \{w \mid w^n \in L \text{ for some } n \geq 1\}$. We take $G_0 = \sqrt[+]{L_0}$. The hardness of G_0 expresses the power of this operation. Note that the class of regular languages is closed under the operation $\sqrt[+]{L}$.

We introduce a new type of pushdown automata useful in proving the hardness of G_0. These automata are equivalent to 2npda's but are 'almost' one-way. The iterative nondeterministic pushdown automaton (in short inpda) is a one-way nondeterministic pushdown automaton with the additional possibility to move its reading head in one step from the right endmarker to the left endmarker. We assume that the input to the automaton is of the form \$w\$. We can treat inpda's as one-way npda's acting on the cyclic word.

Lemma 1. For every 2npda A there exists an equivalent inpda A′.

Proof. We construct inpda A′ simulating A. There is no problem to simulate a left-to-right move of A. We outline how A′ simulates a right-to-left move of the reading head. For this purpose A′ uses the special stack symbol, called the counting symbol. {j is the position of the reading head}.

(1) A′ moves its reading head to the right endmarker and puts onto the stack the number of counting symbols equal to the traversed distance;

(2) A′ moves its reading head in one step from the right endmarker to the left endmarker;

(3) while there is a counting symbol on the stack A′ moves its head to the right and, simultaneously, in each step A′ pops one counting symbol;

(4) A′ moves its head to the right endmarker in every step pushing one counting symbol. Next A′ pops one counting symbol; {there are $j-1$ counting symbols on the stack};

(5) A′ in one step moves its head to the left endmarker;

(6) while there is a counting symbol on the stack, A′ moves its head to the right, in every step popping one counting symbol; {$j-1$ is the position of the reading head}.

A′ simulates the right-to-left move of the reading head of A performing steps (1–6).

Lemma 2. For every $L \in$ 2NPDA there exists a context-free language L′ such that for every nonempty word w, $w \in L$ iff $w\$ \in \sqrt[+]{L'}$.

0020-0190/81/0000–0000/\$02.75 © 1981 North-Holland

Proof. Let A′ be an inpda A′ from Lemma 1, recognizing the language L. Assume that if A accepts a word then its head goes to the right of the input. We construct a one-way npda B which simulates A′ on the input string of the form $w_1\$w_2\$\cdots w_n\$$ (it rejects any string of another form). If A′ moves its head from the right endmarker to the left endmarker then B does not move its head but only changes the state and the stack in the same manner as A′. We can consider B as the 'deceived' automaton A′ in the sense that whenever A′ moves its head to the beginning of the same input word then B has its head already positioned at the beginning of some word w_{i+1} which may be different from w_i. Nevertheless, for every $w \in V^+$, A′ accepts w iff B accepts some word of the form $w_1\$w_2\$\cdots w_n\$$ where $n \geqslant 1$ and $w_1 = w_2 = \cdots = w_n = w$. Hence A′ accepts w iff w\$ belongs to the language $\sqrt[+]{L'}$, where L′ is the context-free language recognized by B.

Theorem 1. For every language $L \in$ 2NPDA there exists a homomorphism h such that for every nonempty word $w \in V^+$, $w \in L$ iff $h(w\$) \in G_0$. ($G_0$ is the hardest language in the class 2NPDA.)

Proof. Take the language L′ from Lemma 2. It was proved in [2] that there exists a homomorphism h such that for every nonempty word v, $v \in L$ iff $h(v) \in L_0$. Lemma 2 implies that $w \in L$ iff there exists $n \geqslant 1$ such that $(w\$)^n \in L'$. Hence $w \in L$ iff $h((w\$)^n) \in L_0$ for some $n \geqslant 1$. Note that $h((w\$)^n) = (h(w\$))^n$. Hence $w \in L$ iff $h(w\$) \in \sqrt[+]{L_0} = G_0$. On the other hand we can easily see that $\sqrt[+]{L_0} \in$ 2NPDA.

Denote by 2DPDA the class of languages accepted by two-way deterministic pushdown automata with one reading head.

Theorem 2. 2NPDA = 2DPDA iff $G_0 \in$ 2DPDA.

Proof. The thesis follows directly from Theorem 1.

Remark. Denote by CFL the class of context-free languages. It is an interesting question whether the class 2DPDA is closed under the operation $\sqrt[+]{\ }$. If this is so then two open problems about two-way pushdown automata are equivalent, namely 2NPDA = 2DPDA iff CFI $\subseteq$ 2DPDA.

References

[1] Z. Galil, Some open problems in the theory of computation as questions about two-way deterministic pushdown automaton languages, Math. Systems Theory 10 (1977) 211–228.

[2] S. Greibach, The hardest context-free language, SIAM J. Comput. 2 (4) (1973).

[3] S. Ginsburg and G. Rose, On the existence of generators for certain AFL, Inform. Sci. 2 (1970) 431–466.