Available online at www.sciencedirect.com
ScienceDirect
Information Processing Letters 106 (2008) 162–163
Information Processing Letters
www.elsevier.com/locate/ipl

Fooling Turing machines with sublogarithmic space: a note on 'For completeness, sublogarithmic space is no space' by M. Agrawal

Andrzej Szepietowski

Institute of Computer Science, University of Gdańsk, ul. Wita Stwosza 57, 80952 Gdańsk, Poland

Received 18 May 2007; received in revised form 6 September 2007; accepted 10 November 2007

Available online 17 November 2007

Communicated by L.A. Hemaspaandra

Keywords: Computational complexity; Sublogarithmic space

1. Introduction

M. Agrawal in his paper [1] uses the following lemma in order to prove his, main, Theorem 3.

Lemma 1. *Let M be a sublogarithmic space DTM (deterministic Turing machine). Then, there is a constant N such that for every $n \geqslant N$, for all strings z_1, z_2, w, and for every $\ell \geqslant 0$, the space used by M on the input $z_1 w^{n+\ell n!} z_2$ is the same as the space used on the input $z_1 w^n z_2$.*

He claims that this technical lemma was proved in [4], where Liśkiewicz and Reischuk proved that the alternating hierarchy for sublogarithmic space is infinite (similar results were obtained in [2] and [3]). But in [4] there were additional assumptions not mentioned explicitly in the lemma. In this paper we show that the lemma in the form presented in [1] is not valid. This does not mean that Theorem 3 of [1] is wrong. In fact, it is valid and can be proved using essentially the same proof; only instead of Lemma 1 one should use directly the $n \to n + n!$ technique (described in [4] or [3]) or the following improved version of Lemma 1.

Lemma 2. *Let M be a sublogarithmic space DTM (deterministic Turing machine) and let S be a space bound in $o(\log n)$. Then, there is a constant N such that for every $n \geqslant N$, for all strings z_1, z_2, w, and for every $\ell \geqslant 0$, if $Space_M(z_1 w^n z_2) \leqslant S(n)$ then the space used by M on the input $z_1 w^{n+\ell n!} z_2$ is the same as the space used on the input $z_1 w^n z_2$.*

Here $Space_M(x)$ denotes the number of cells that M visits on the work tape during the computation on input x.

2. Counter-example

We consider the Turing machine model with a two-way, read-only input tape and a separate two-way, read-write work tape. The number of tape cells used on the work tape, called space, is our measure of computational complexity. A Turing machine is said to be $L(n)$ space-bounded if no computation on any input of length n, uses more than $L(n)$ space (see [5] for more details).

We shall describe a Turing machine M which is a counter-example for Lemma 1. The input tape alphabet of M is $\Sigma = \{0, 1, \#, a\}$, the work tape alphabet is

E-mail address: matszp@univ.gda.pl.

0020-0190/$ – see front matter
doi:10.1016/j.ipl.2007.11.005

$\Gamma = \{0, 1, b, B\}$, where B is the blank symbol. At the beginning the input tape contains $\$w\$$, where w is an input word and \$ is the special marker. Machine M works in the following way:

First checks if the input w is of the form

$$w = b_0\#b_1\#\cdots\#b_k\#a^n,$$

where $k \geqslant 0$ and $n \geqslant 1$ are natural numbers and $b_i \in \{0, 1\}^*$ is the binary description of the number i. Machine M checks one by one if every number b_i is by one greater than its predecessor b_{i-1}. In order to do this, M walks between b_i and b_{i-1} and compares one by one the digits of both numbers. It has to remember the digit which is compared and the position of the digit in b_i and b_{i-1}. The digit is remembered in the final state and the position is remembered in binary on the work tape. M stops (in a rejecting state) if b_i is not by one greater than b_{i-1}. M stops without using any space if the input does not begin with 0#.

The length of the binary description b_i of the natural number $i \geqslant 1$ is $|b_i| = \lfloor \log i \rfloor + 1$. Thus, M uses exactly $\lfloor \log(\lfloor \log k \rfloor + 1) \rfloor + 1$ space if the input is of the form $w_k x$ with $w_k = b_0\#b_1\#\cdots\#b_k\#$, and $x \in \{0, 1, \#, a\}^*$ does not begin with $b_{k+1}\#$. Hence, the number of cells used in the first stage, say s, is equal to the length of b_k and is bounded by $s \leqslant c \cdot \log\log k \leqslant c \cdot \log\log |w_k x|$ for some constant $c > 0$. At the end of the first stage M marks these s used cells by writing the symbol b on them.

Next machine M counts a's standing at the end of the input. To do this it goes right through a's and after each step it increases the number written in binary on the marked cells. If it tries to use more than s marked cells, i.e. if $n > 2^s - 1$, then M stops in a rejecting state not using more than s cells. If $n \leqslant 2^s - 1$, then marks one cell more on the work tape and stops.

The deterministic Turing machine M is $O(\log\log n)$ space bounded. Let N be a number mentioned by Lemma 1 and let $n > N$ be an arbitrary number of the form $n = 2^s - 1$, where s is a natural number. Consider now the input

$$z_1 w^n z_2,$$

where $w = a$, $z_2 = \lambda$ (the empty word),

$$z_1 = b_0\#b_1\#\cdots\#b_k\#$$

and $k = 2^n - 1$. The length of b_k is equal to $|b_k| = n = 2^s - 1$ and M uses and marks exactly s cells on the work tape in the first stage, and the number of a's equal to $n = 2^s - 1$ and can be written within marked space, hence M uses exactly $s + 1$ space on the input $z_1 w^n z_2$. On the other hand for every $\ell \geqslant 1$ the space used on the input $z_1 w^{n+\ell n!} z_2$ is equal to s, because then the number of a's is too big and its binary description is longer than the marked space s. This is a contradiction with Lemma 1.

Acknowledgements

I would like to thank the anonymous referees for pointing out how the proof of Theorem 3 in Agrawal's paper can be corrected.

References

[1] M. Agrawal, For completeness, sublogarithmic space is no space, Inform. Process. Lett. 82 (2002) 321–325.
[2] B. von Braunmühl, R. Gengler, R. Rettinger, The Alternation hierarchy for sublogarithmic space is infinite, Comput. Complexity 3 (1993) 207–230.
[3] V. Geffert, A hierarchy that does not collapse, alternation in low level space, Theoret. Inform. Appl. 28 (5) (1994) 465–512.
[4] M. Liśkiewicz, R. Reischuk, The complexity world below logarithmic space, in: Proceedings of the Structure in Complexity Theory Conference, 1993, pp. 64–78.
[5] A. Szepietowski, Turing Machines with Sublogarithmic Space, Lecture Notes in Comput. Sci., vol. 843, Springer-Verlag, Berlin, 1994.