Information and Computation 253 (2017) 163–178

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Alternating space is closed under complement and other simulations for sublogarithmic space [☆], [☆☆]

Viliam Geffert

Department of Computer Science, P. J. Šafárik University, Jesenná 5, 04154 Košice, Slovakia

ARTICLE INFO

Article history:
Received 16 September 2016
Received in revised form 30 January 2017
Available online 21 February 2017

Keywords:
Computational complexity
Alternation
Sublogarithmic space

ABSTRACT

We present some new simulations for $\mathrm{ASPACE}(s(n))$, the class of languages accepted by alternating Turing machines with $O(s(n))$ space, with absolutely no assumptions on $s(n)$. These simulations provide the following inclusions:
(a) $\mathrm{ASPACE}(s(n)) \subseteq \mathrm{DTIME}(n \cdot 2^{O(s(n))})$. This extends, to sublogarithmic space bounds, the classic result stating that $\mathrm{ASPACE}(s(n)) \subseteq \mathrm{DTIME}(2^{O(s(n))})$, proved under the assumption $s(n) \geq \log n$.
(b) $\mathrm{ASPACE}(s(n)) \subseteq \mathrm{NTIMESPACE}(n \cdot 2^{O(s(n))}, 2^{O(s(n))})$, a simulation by nondeterministic machines with simultaneous bounds on time and space. This improves the known inclusion, stating that $\mathrm{ASPACE}(s(n)) \subseteq \mathrm{NSPACE}(2^{O(s(n))})$, proved under the assumption $s(n) \geq \log\log n$.
(c) $\mathrm{ASPACE}(s(n)) = \text{co-}\mathrm{ASPACE}(s(n))$, i.e., the alternating space is closed under complement, independently of whether $s(n)$ is above $\log n$ and of whether the original machine can get into an infinite loop. This solves a long-standing open problem. Quite surprisingly, this complementary simulation does not eliminate infinite loops—the new machine itself goes to infinite loops along some computation paths.

1. Introduction

Space complexity of a computation, introduced in [16,17] in 1965, is the second in importance among various computational complexity measures, right after the time complexity. It turns out that $\log n$ is the most significant boundary among all space complexity bounds, since the space complexity classes below $\log n$ are radically different from those above. Due to the impossibility of storing pointers to input positions or counting the number of executed steps, many of the standard techniques do not work. Results that are "easy" above $\log n$ either become difficult to prove, or are not valid any more, or are open questions at the moment.

For example, if $s(n) \geq \Omega(\log n)$, it is trivial to show that $\mathrm{DSPACE}(s(n))$ is closed under complement. However, the trivial argument does not work below $\log n$, because the machine may reject by getting into an infinite loop and we do not have enough space to detect such loops by counting executed steps, up to $n \cdot 2^{\Omega(s(n))}$, the number of possible configurations. To show that $\mathrm{DSPACE}(s(n)) = \text{co-}\mathrm{DSPACE}(s(n))$ without any assumptions on $s(n)$, a more sophisticated simulation technique

[☆] A preliminary and weaker version of this work was presented at the 20-th International Conference on Developments in Language Theory (DLT 2016), July 25–28, 2016, Montréal, Canada [Vol. 9840 of Lect. Notes Comput. Sci. Springer-Verlag, pp. 190–202].

[☆☆] Supported by the Slovak Grant Agency for Science (VEGA) under contract 1/0142/15 "Combinatorial Structures and Complexity of Algorithms" and by the Slovak Research and Development Agency under contract APVV-15-0091 "Efficient Algorithms, Automata, and Data Structures".

E-mail address: viliam.geffert@upjs.sk.

http://dx.doi.org/10.1016/j.ic.2017.02.001
0890-5401/© 2017 Elsevier Inc. All rights reserved.

was necessary [30]. The classic Savitch's simulation of nondeterministic machines by the deterministic ones [29] requires actually space of size $O((s(n)+\log n)^2)$ because, among others, we have to remember input head positions for machine's configurations on the input of length n. If $s(n)$ is above $\log n$, the $\log n$ penalty in the big-O notation disappears. By a more complicated technique [28], the space has been reduced to $O(s(n)\cdot(s(n)+\log n))$.

In the nondeterministic case, we have the surprisingly short proof showing that $\mathrm{NS{\small PACE}}(s(n))$ is closed under complement for $s(n)\geq\Omega(\log n)$ [21,32] but the problem is still open for $s(n)$ below $\log n$.

The same problem arises for the alternating machines, introduced in [7] to provide a neater view of the relationship between time and space complexity classes, by generalization of nondeterminism and parallelism at the same time. It is trivial to invert the roles of existential and universal decisions and the roles of accepting and rejecting states, which gives a machine accepting the complement of the original language, *if the original machine never gets into an infinite loop*. Thus, $\mathrm{AS{\small PACE}}(s(n))$ is closed under complement for $s(n)\geq\Omega(\log n)$, since we can force the machine to halt [7, Thm. 2.6]. This does not imply anything for $s(n)$ below $\log n$. For example, by inductive counting [21,32] (see also [33]), the hierarchy of $s(n)$ space bounded machines making a constant number of alternations collapses to the first level for $s(n)\geq\Omega(\log n)$, and hence the classes Σ_k- and Π_k-$\mathrm{S{\small PACE}}(s(n))$ are closed under complement for each $k\geq 1$, while the corresponding sublogarithmic space classes are provably not closed under complement for each $k\geq 2$ [6,12,25].

The importance of even the lowest levels of space bounded computations is established by several results. For example, we know that $\mathrm{NS{\small PACE}}(\log n)$ separates from $\mathrm{DS{\small PACE}}(\log n)$ if and only if $\mathrm{NS{\small PACE}}(\log\log n)$ separates from $\mathrm{DS{\small PACE}}(\log\log n)$ on unary languages [13]. The sublogarithmic alternating space classes may actually be quite strong, e.g., there exists a binary NP-complete language such that its unary coded version is in $\mathrm{AS{\small PACE}}(\log\log n)$ [15].

In this paper, we shall present some new simulations for $\mathrm{AS{\small PACE}}(s(n))$, with absolutely no assumptions on $s(n)$—not excluding even functions that are not monotone, e.g., functions that are unbounded, but with $s(n)=0$ infinitely many times. First, we shall provide a time-efficient simulation of alternating machines with small space by deterministic machines. Namely, we shall show that

$$\mathrm{AS{\small PACE}}(s(n)) \quad\subseteq\quad \mathrm{DT{\small IME}}(n\cdot 2^{O(s(n))})\,. \tag{1}$$

This extends, to sublogarithmic space bounds, the classic result [7, Thm. 3.3] stating that $\mathrm{AS{\small PACE}}(s(n))\subseteq\mathrm{DT{\small IME}}(2^{O(s(n))})$ for $s(n)\geq\log n$.

Our deterministic simulation within $n\cdot 2^{O(s(n))}$ time uses superlinear space, namely, $n\cdot 2^{\Omega(s(n))}$. However, it turns out that the simulating machine has several special properties, and hence it can be replaced by more powerful machine models, space efficiently. Based on this, we shall derive that

$$\mathrm{AS{\small PACE}}(s(n)) \quad\subseteq\quad 1\text{-}\mathrm{NT{\small IME}S{\small PACE}}^{\mathrm{dm}}(n\cdot 2^{O(s(n))}, 2^{O(\lfloor s(n)\rfloor)})\,,$$

which represents a simulation by one-way nondeterministic machines not executing more than $n\cdot 2^{O(s(n))}$ steps along any computation path and using initially delimited worktapes of size $2^{k\cdot\lfloor s(n)\rfloor}$, for some integer constant $k\geq 1$. This improves the known inclusion $\mathrm{AS{\small PACE}}(s(n))\subseteq\mathrm{NS{\small PACE}}(2^{O(s(n))})$ that was proved for $s(n)\geq\Omega(\log\log n)$ under some weak constructibility assumption [31]. After some modification, we can obtain a simulation by nondeterministic machines with simultaneous bounds on time and space using worktapes that are initially empty. However, such machines are no longer one-way:

$$\mathrm{AS{\small PACE}}(s(n)) \quad\subseteq\quad \mathrm{NT{\small IME}S{\small PACE}}(n\cdot 2^{O(s(n))}, 2^{O(s(n))})\,.$$

Finally, by converting the deterministic machine—the one introduced by (1) and using $n\cdot 2^{\Omega(s(n))}$ space—back into an alternating device, we can convert a two-way alternating machine into a two-way alternating machine for the complement of the original language, keeping the space bound $O(s(n))$. That is, the alternating space is closed under complement:

$$\mathrm{AS{\small PACE}}(s(n)) \quad=\quad \text{co-}\mathrm{AS{\small PACE}}(s(n))\,,\qquad \text{for each } s(n)\,.$$

This conversion works with no additional assumptions—it does not depend on whether $s(n)$ is above $\log n$ nor on whether the original machine gets into infinite loops. This solves a long-standing open problem [6]. Quite surprisingly, our technique of complementing does not eliminate infinite loops—the new machine itself rejects by going into infinite loops along some computation paths.

It turns out that all these simulations become conceptually much simpler if we consider another reasonable way to define space complexity, studied, e.g., in [2,8]: the classes[1] $\mathrm{AS{\small PACE}}^{\mathrm{dm}}(s(n))$. For this reason, in Section 2, we first present simulations for the classes $\mathrm{AS{\small PACE}}^{\mathrm{dm}}(s(n))$, using also some weak constructibility assumptions. After that, in Section 3, all results will be updated for the classic complexity classes $\mathrm{AS{\small PACE}}(s(n))$, where the worktape is initially empty and the value $s(n)$ is not known in advance.

[1] By $X\mathrm{S{\small PACE}}^{\mathrm{dm}}(s(n))$, for $X\in\{\mathrm{D},\mathrm{N},\mathrm{A}\}$, we denote the classes of languages accepted by deterministic, nondeterministic, and alternating Turing machines starting with a worktape consisting of $\lfloor s(n)\rfloor$ blank cells delimited by endmarkers, as opposed to the more common complexity classes $X\mathrm{S{\small PACE}}(s(n))$ where the worktape is initially empty and the machine must use its own computational power to make sure that it respects, along each computation path on each input of length n, the space bound $s(n)$. Clearly, $X\mathrm{S{\small PACE}}(s(n))\subseteq X\mathrm{S{\small PACE}}^{\mathrm{dm}}(s(n))$ and these two classes are equal if $\lfloor s(n)\rfloor$ is fully space constructible. The notation "dm" derives from "Demon" Turing Machines [8].