Information Processing Letters 115 (2015) 750–753

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

On the de-randomization of space-bounded approximate counting problems

Dean Doron *,[1], Amnon Ta-Shma [1]

The Blavatnik School of Computer Science, Tel-Aviv University, Tel-Aviv, 6997801, Israel

ARTICLE INFO

Article history:
Received 15 September 2014
Received in revised form 11 March 2015
Accepted 12 March 2015
Available online 17 March 2015
Communicated by B. Doerr

Keywords:
Computational complexity
Approximation algorithms
Randomized algorithms
Space bounded computation
Space bounded quantum computation
Space bounded approximation schemes

ABSTRACT

It was recently shown that SVD and matrix inversion can be approximated in quantum log-space [1] for well formed matrices. This can be interpreted as a fully logarithmic quantum approximation scheme for both problems. We show that if $\mathsf{prBQL} = \mathsf{prBPL}$ then every fully logarithmic quantum approximation scheme can be replaced by a probabilistic one. Hence, if classical algorithms cannot approximate the above functions in logarithmic space, then there is a gap already for languages, namely, $\mathsf{prBQL} \neq \mathsf{prBPL}$.

On the way we simplify a proof of Goldreich for a similar statement for time bounded probabilistic algorithms. We show that our simplified algorithm works also in the space bounded setting (for a large set of functions) whereas Goldreich's approach does not seem to apply in the space bounded setting.

1. Introduction

Two well known approximation problems are approximating the singular value decomposition (SVD) of a matrix and approximating matrix inverse. Both problems were extensively studied in the *time bounded* model, e.g., in [2–4]. In the *space bounded* model it was recently shown that SVD and matrix inversion can be additively approximated by a *quantum* algorithm using only logarithmic space [1]. This can be interpreted as a fully logarithmic quantum approximation scheme (with additive accuracy) for the problems.

The result [1] shows a possible gap between quantum and classical algorithms for the task of *approximating a function*. It is not clear, however, whether it also implies a possible gap between the power of quantum and classical log-space algorithms for *decision problems*. Thus, a natural question is the following. Suppose no classical log-space algorithm can approximate the SVD. Does that imply that $\mathsf{prBQL} \neq \mathsf{prBPL}$? In the contra-positive we ask whether a "de-quantumization" of decision classes (i.e., $\mathsf{prBQL} = \mathsf{prBPL}$) implies a de-quantumization of approximation schemes.

The question was also asked in the model of classical time bounded computations. A classical result by Stockmeyer [5] implies that the problem of *approximate counting* of a general NP predicate can be solved in $\mathsf{FBPP}^{\mathsf{NP}}$. Shaltiel and Umans [6] derandomized the result under the assumption that $\mathsf{E}_{\|}^{\mathsf{NP}}$ requires exponential size single-valued nondeterministic circuits.[2]

While in general we do not know how to approximate functions in #P better than in $\mathsf{FBPP}^{\mathsf{NP}}$, there exist #P-complete functions for which there exists a fully polynomial randomized approximation scheme (FPRAS) that

* Corresponding author. Tel.: +972 54 5865755.
E-mail addresses: deandoron@mail.tau.ac.il (D. Doron), amnon@tau.ac.il (A. Ta-Shma).

[1] Supported by the Israel Science Foundation grant no. 994/14 and by the United States–Israel Binational Science Foundation grant no. 2010120.

[2] $\mathsf{E}_{\|}^{\mathsf{NP}}$ is the class E with *non-adaptive* oracle queries to NP. A *single-valued* nondeterministic circuit is the nonuniform analogue of $\mathsf{NP} \cap \mathsf{coNP}$.

http://dx.doi.org/10.1016/j.ipl.2015.03.005
0020-0190/

does not require an oracle access to an NP language. Jerrum, Sinclair and Vigoda [7] proved this for the permanent. Goldreich [8] showed that derandomizing the FPRAS for the permanent (or any other FPRAS) can be done under the assumption that $\mathsf{prBPP} = \mathsf{P}$:

Theorem 1. *(See [8, Corollary 3.6].) If* $\mathsf{prBPP} = \mathsf{P}$ *then every function that has an FPRAS also has such a deterministic scheme. Furthermore, for every polynomial* p*, there exists a polynomial* p' *such that if the probabilistic scheme runs in time* p*, then the deterministic one runs in time* p'*.*

In [8] Goldreich is after a "uniform" pseudorandom generator (see [8] for exact details) and Theorem 1 is a corollary of a more general technique used in his construction of a uniform PRG. Roughly speaking, Goldreich's argument works as follows: Given a probabilistic Turing machine $M(x, y)$ that approximates $f(x)$ well (where x is the input and y is the internal random coins used by M), we construct a specific sequence y_0 such that $M(x, y_0)$ also approximates $f(x)$ well. The bits of y are fixed bit by bit, where each bit is determined by a single query to a certain prBPP problem that depends on the random bits that were set so far.

In our case we deal with space bounded problems. It seems Goldreich's approach does not generalize to the space bounded case, as the random coins string is kept on the work tape, and its length is bounded by the time complexity of the Turing machine – which can be polynomial.

In this note we give an alternative (and simpler) proof that directly computes the approximated value using prBPP oracle calls, and we show this approach does generalize to a large class of functions in the space bounded model. In particular we show that if no probabilistic algorithm can approximately compute the SVD of certain well formed matrices[3] – a task that quantum algorithms can do with logarithmic space – then there is already a separation of the decision classes and $\mathsf{prBQL} \neq \mathsf{prBPL}$. We give an intuitive explanation of our proof technique at the beginning of Section 3 followed by a formal argument.

Throughout the paper we use the notations of common complexity classes freely. Exact definitions can be found, for example, in [9]. For a complexity decision class $\mathcal{C}$, we denote $\mathsf{pr}\mathcal{C}$ as the corresponding promise class and $\mathsf{F}\mathcal{C}$ as the corresponding function class. In Section 2 we define space bounded approximation schemes. We remark that we are not aware of any previous definition of space bounded approximation schemes. In Section 3 we present and prove our result.

2. Definitions

We first define approximation for real valued functions $f : \{0,1\}^\star \to \mathbb{R}$. We have two notions of approximation: additive and multiplicative. We say y additively approximates $f(x)$ with accuracy δ if $y \in [f(x) \pm \delta]$. We say y multiplicatively approximates $f(x)$ with accuracy δ if $y \in [(1 \pm \delta) f(x)]$.

We assume (as is customary in previous works) that the approximation algorithm M outputs a dyadic rational number, i.e., we interpret the string $M(x)$ as the rational number whose binary representation is the binary string $M(x)$. This, in particular, implies that if M has time complexity $t(n)$ then M can only output integers whose absolute value is at most $2^{t(n)}$, and this assumption is implicitly used in previous works. This assumption also implies that if $M(x)$ is a non-zero rational number then $|M(x) - 0| \geq 2^{-t(n)}$.

We say a function f is $R(n)$-bounded if $|f(x)| \leq R(n)$ for every $x \in \{0,1\}^n$ and some known uniform function R. Since we are only interested in functions that can be approximated by polynomial time algorithms we can assume w.l.o.g. that $R = 2^{\mathrm{poly}(n)}$, because no polynomial time algorithm can approximate a higher value. However, for specific functions f sometimes a better bound can be taken.

We begin with the time bounded model where it is customary to use *multiplicative* approximation:

Definition 2. A fully polynomial randomized approximation scheme (FPRAS) for an $R(n)$-bounded function $f : \{0,1\}^n \to [0, R(n)]$ is a randomized Turing machine M that on input x, an accuracy parameter δ and a confidence parameter ε, runs in $\mathrm{poly}(|x|, \delta^{-1}, \log \varepsilon^{-1}, \log R(n))$ time and outputs $M(x) \in [(1 \pm \delta) f(x)]$ with probability at least $1 - \varepsilon$, where we interpret the string $M(x)$ as the dyadic rational number whose binary representation is given by the binary string $M(x)$.

A fully polynomial (deterministic) approximation scheme (FPAS) is obtained if M in the above definition is deterministic, ε is set to 0, and the dependence on ε is removed.

In this paper we define log-space approximation schemes, but use *additive approximation* rather than multiplicative approximation, mainly because the major examples we have for such approximation schemes only achieve additive accuracy. We define:

Definition 3. A fully logarithmic randomized (resp. quantum) approximation scheme for an $R(n)$-bounded function $f : \{0,1\}^n \to [-R(n), R(n)]$ is a randomized (resp. quantum) Turing machine M that on input x, an error parameter δ and a confidence parameter ε, runs in $\mathrm{poly}(|x|, \delta^{-1}, \log \varepsilon^{-1}, \log R(n))$ time, uses $O(\log |x| + \log \delta^{-1} + \log \log \varepsilon^{-1} + \log \log R(n))$ space and outputs $M(x) \in [f(x) \pm \delta]$ with probability at least $1 - \varepsilon$.

The quantum space model we use has classical control and allows intermediate measurements, see [10,1]. A fully logarithmic (deterministic) approximation scheme is obtained if M in the above definition is deterministic, ε is set to 0, and the dependence on ε is removed. We let FLAS abbreviate "fully logarithmic approximation scheme" and FLRAS (resp. FLQAS) the randomized (resp. quantum) versions.

[3] Specifically, bounded norm matrices with well-separated singular values. An $n \times n$ matrix has well-separated singular values if there exists some constant c such that for all $i \neq j$, $|\sigma_i - \sigma_j| \geq n^{-c}$.