University of Latvia

Marats Golovkins

Quantum Automata and Quantum Computing

Relevant Publications

Appendix to Doctoral Thesis

Riga 2002

Relevant Publications

1. M. Golovkins, M. Kravtsev. Probabilistic Reversible Automata and Quantum Automata. *COCOON 2002, Lecture Notes in Computer Science*, Vol. 2387, pp. 574-583, 2002.

2. M. Golovkins. Quantum Pushdown Automata. *SOFSEM 2000, Lecture Notes in Computer Science*, Vol. 1963, pp. 336-346, 2000.

3. A. Ambainis, R. Bonner, R. Freivalds, M. Golovkins, M. Karpinski. Quantum Finite Multitape Automata. *SOFSEM 1999, Lecture Notes in Computer Science*, Vol. 1725, pp. 340-348, 1999.

4. M. Golovkins, M. Kravtsev. Probabilistic Reversibility and Its Relation to Quantum Automata. *Quantum Computation and Learning. 3rd International Workshop. Proceedings*, pp. 1-22, Riga, 2002.

5. M. Golovkins. On Quantum Pushdown Automata. *Quantum Computation and Learning. 2nd International Workshop. Proceedings*, pp. 41-51, Sundbyholms Slott, Sweden, 2000.

6. A. Ambainis, R. Bonner, R. Freivalds, M. Golovkins, M. Karpinski. Quantum vs Probabilistic Finite Multitape Automata. *Quantum Computation and Learning. 1st International Workshop. Proceedings*, pp. 36-43, Riga, 1999.

7. R. Bonner, R. Freivalds, M. Golovkins. Projections of Multitape Languages Recognized by Quantum and Probabilistic Finite Automata. *Quantum Computation and Learning. 1st International Workshop. Proceedings*, pp. 84-92, Riga, 1999.

8. M. Golovkins. An Introduction to Quantum Pushdown Automata. *Quantum Computation and Learning. 1st International Workshop. Proceedings*, pp. 44-52, Riga, 1999.

Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Tokyo

Oscar H. Ibarra Louxin Zhang (Eds.)

Computing and Combinatorics

8th Annual International Conference, COCOON 2002
Singapore, August 15-17, 2002
Proceedings

Springer

Referees

Stephen Alstrup
Luzi Anderegg
Maria Andreou
Dan Archdeacon
Abdullah Arslan
Dorit Batler
Giuseppe Di Battista
Jacir Luiz Bordim
Ran Canetti
Alberto Caprara
Xin Chen
Sung-Woo Cho
Francisco Coelho
Barry Cohen
Zhe Dang
Mart de Graaf
Joerg Derungs
Stephan Eidenbenz
Panagiota Fatourou
Mike Fellows
Vladimir Filkov
Eldar Fischer
Dimitris Fotakis
Pierre Fraignaud
Jozef Gruska
Nicolas Hanusse
Tero Harju
Sariel Har-Peled
Joel Hass
Thomas Hofmeister
Ed Hong
Tao Jiang
Sungwon Jung
Michael Kaminski
George Karakostas
Dimitris Kavvadias
Daesan Kim
Spyros Kontogiannis
Jeff Lagarias
Donghoon Lee
Hanno Lefmann
Chin-Laung Lei
Stefano Lonardi
Hsueh-I Lu
Meena Mahajan
Ross McConnell
Janos Makowski
Pablo Moisset
Tal Mor
Matthias Mueller
Sotiris Nikoletseas
Roderic D. M. Page
Aris Pagourtzis
Vicky Papadopoulou
Jungheum Park
Kunsoo Park
Eynat Rafalin
Md. Saidur Rahman
S. Rajasekaran
B. Ravikumar
Hein Roehrig
Brigitte Servatius
Diane Souvaine
Mike Steel
Pavel Sumazin
Wing Kin Sung
Subhash Suri
Gabor Szabo
Laszlo Szekely
Arie Tamir
Joseph A. Thas
Takeshi Tokuyama
Nicholas Tran
John Tromp
Ming-Jer Tsai
Sam Wagstaff
Yuan-Fang Wang
Birgitta Weber
David Wei
Hongjun Wu
Jihoon Yang
Sheng Yu
Christos Zaroliagis
Shiyu Zhou
Xiao Zhou

Sponsoring Institutions

Department of Mathematics, NUS
Lee Foundation, Singapore

Organizing Institutions

Department of Mathematics, NUS
School of Computing, NUS
The Logistics Insitute - Asia Pacific, NUS

Table of Contents

Invited Lectures

Complexity Theory I

Discrete Algorithms I

Computational Biology and Learning Theory I

Coding Theory and Cryptography

Parallel and Distributed Architectures

Graph Theory

Radio Networks

Automata and Formal Languages

Internet Networks

Computational Geometry I

Computational Biology and Learning Theory II

Discrete Algorithms II

Computational Geometry II

Probabilistic Reversible Automata and Quantum Automata

Marats Golovkins* and Maksim Kravtsev**

Institute of Mathematics and Computer Science, University of Latvia
Raiņa bulv. 29, Riga, Latvia
marats@latnet.lv, maksims@batsoft.lv

Abstract. To study relationship between quantum finite automata and probabilistic finite automata, we introduce a notion of probabilistic reversible automata (PRA, or doubly stochastic automata). We find that there is a strong relationship between different possible models of PRA and corresponding models of quantum finite automata. We also propo a classification of reversible finite 1-way automata.

1 Introduction

Here we introduce common notions used throughout the paper as well as sum marize its contents.

We analyze two models of probabilistic reversible automata in this pape namely, 1-way PRA and 1.5-way PRA.

If not specified otherwise, we denote by Σ an input alphabet of an automaton Every input word is enclosed into *end-marker* symbols # and \$. Therefore w introduce a *working alphabet* as $\Gamma = \Sigma \cup \{\#, \$\}$. By Q we normally understan the set of states of an automaton. By $\overline{L}$ we understand complement of a langua L. Given an input word ω, by $|\omega|$ we understand the number of symbols in ω and with $[\omega]_i$ we denote i-th symbol of ω, counting from the beginning (excludin end-markers). By $q \xrightarrow{S} q'$, $S \subset \Sigma^*$, we denote that there is a positive probabili to get to a state q' by reading some word $\xi \in S$, starting in q.

Let us consider A. Nayak's model of quantum automata with mixed state (QFA-N, [N 99]). (Evolution is characterized by a unitary matrix and subs quent measurements are performed after each step, POVM measurements no being allowed.) If a result of every measurement is a single configuration, no a superposition, and measurements are performed after each step, we actuall get a probabilistic automaton. However, the following property applies to suc probabilistic automata - their evolution matrices are *doubly* stochastic. Thi encourages us to give the following definition for probabilistic reversible automat

* Research partially supported by the Latvian Council of Science, grant No. 01.03 and grant for Ph.D. students; University of Latvia, K. Morbergs grant; Europe Commission, contract IST-1999-11234

** Research partially supported by the Latvian Council of Science, grant No. 01.03 and European Commission, contract IST-1999-11234

O.H. Ibarra and L. Zhang (Eds.): COCOON 2002, LNCS 2387, pp. 574–583, 2002.

Definition 1.1. *A probabilistic automaton is called* reversible *if its linear operator can be described by a doubly stochastic matrix.*

At least two definitions exist, how to interpret word acceptance, and hence, language recognition, for reversible automata.

Definition 1.2. Classical acceptance. *(C-automata) We say that an automaton accepts (rejects) a word classically, if its set of states consists of two disjoint subsets: accepting states and rejecting states, and the following conditions hold:*
-the automaton accepts the word, if it is in accepting state after having read the last symbol of the word;
-the automaton rejects the word, if it is in rejecting state after having read the last symbol of the word.

Definition 1.3. "Decide and halt" acceptance. *(DH-automata) We say that an automaton accepts (rejects) a word in a decide-and-halt manner, if its set of states consists of three disjoint subsets: accepting states, rejecting states and non-halting states, and the following conditions hold:*
-the computation is continued only if the automaton enters a non-halting state;
-if the automaton enters an accepting state, the word is accepted;
-if the automaton enters a rejecting state, the word is rejected.

Having defined word acceptance, we define language recognition in an equivalent way as in [R 63]. We consider only bounded error language recognition in this paper. By $P_{x,A}$ we denote the probability that a word x is accepted by an automaton A.

Definition 1.4. *We say that a language L is recognized with bounded error by an automaton A with interval (p_1, p_2) if $p_1 < p_2$ and $p_1 = \sup\{P_{x,A} \mid x \notin L\}$, $p_2 = \inf\{P_{x,A} \mid x \in L\}$.*

We say that a language is recognized with a probability p if the language is recognized with interval $(1-p, p)$. We say that a language is recognized with probability $1-\varepsilon$, if for every $\varepsilon > 0$ there exists an automaton which recognizes the language with interval $(\varepsilon_1, 1-\varepsilon_2)$, where $\varepsilon_1, \varepsilon_2 \leq \varepsilon$.

In Section 2, we discuss properties of PRA C-automata (PRA-C). We prove that PRA-C recognize the class of languages $a_1^* a_2^* \ldots a_n^*$ with probability $1-\varepsilon$. This class can be recognized by measure-many quantum finite automata [KW 97] (QFA-KW), with worse acceptance probabilities, however [ABFK 99]. This also implies that QFA-N recognize this class of languages with probability $1-\varepsilon$. Further, we show general class of regular languages, not recognizable by PRA-C. In particular, such languages as (a,b)*a and a(a,b)* are in this class. This class has strong similarities with the class of languages, not recognizable by QFA-KW [AKV 00]. We also show that the class of languages recognized by PRA-C is closed under boolean operations. In Section 3 we prove, that PRA DH-automata do not recognize the language (a,b)*a. In Section 4 we discuss some properties of 1-way PRA. We also present an alternative notion of probabilistic reversibility, not connected with quantum automata. In Section 5 we propose a classification of reversible automata (deterministic, probabilistic and quantum).

2 1-Way Probabilistic Reversible C-Automata

Definition 2.1. *1-way probabilistic reversible C-automaton (PRA-C)* $A = (Q, \Sigma, q_0, Q_F, \delta)$ *is specified by a finite set of states* Q, *a finite input alphabet* Σ, *an initial state* $q_0 \in Q$, *a set of accepting states* $Q_F \subseteq Q$, *and a transition function* $\delta : Q \times \Gamma \times Q \longrightarrow \mathbb{R}_{[0,1]}$, *where* $\Gamma = \Sigma \cup \{\#, \$\}$ *is the input tape alphabet of* A *and* $\#$, $\$$ *are end-markers not in* Σ. *Furthermore, transition function satisfies the following requirements:*

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \ \sum_{q \in Q} \delta(q_1, \sigma_1, q) = 1 \tag{1}$$

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \ \sum_{q \in Q} \delta(q, \sigma_1, q_1) = 1 \tag{2}$$

For every input symbol $\sigma \in \Gamma$, the transition function may be determined by a $|Q| \times |Q|$ matrix V_σ, where $(V_\sigma)_{i,j} = \delta(q_j, \sigma, q_i)$.

We define word acceptance as specified in Definition 1.2. The set of rejecting states is $Q \setminus Q_F$. We define language recognition as in Definition 1.4.

Now we present several results on the class of languages recognizable by PRA-C.

Lemma 2.2. *If a language is recognized by a PRA-C* A *with interval* (p_1, p_2), *exists a PRA-C which recognizes the language with probability* p, *where*

$$p = \begin{cases} \frac{p_2}{p_1 + p_2}, & \text{if } p_1 + p_2 \geq 1 \\ \frac{1-p_1}{2-p_1-p_2}, & \text{if } p_1 + p_2 < 1. \end{cases}$$

Theorem 2.3. *If a language is recognized by a PRA-C, it is recognized by PRA-C with probability* $1 - \varepsilon$.

Proof. Idea of the proof. Assume that a language L is recognized by a PRA-C A. The language L is recognized with probability $1 - \varepsilon$, using a system of n identical copies of A. A system of n PRA-C automata may be simulated by a single PRA-C automaton. □

Lemma 2.4. *If a language* L_1 *is recognizable with probability greater than* $\frac{1}{2}$ *and a language* L_2 *is recognizable with probability greater than* $\frac{2}{3}$ *then languages* $L_1 \cap L_2$ *and* $L_1 \cup L_2$ *are recognizable with probability greater than* $\frac{1}{2}$.

Theorem 2.5. *The class of languages recognized by PRA-C is closed under intersection, union and complement.*

Proof. Let us consider languages L_1, L_2 recognized by some PRA-C automata. By Theorem 2.3, these languages is recognizable with probability $1 - \varepsilon$, and therefore by Lemmas 2.2 and 2.4, union and intersection of these languages are also recognizable. If a language L is recognizable by a PRA-C A, we can construct an automaton which recognizes a language $\overline{L}$ just by making accepting states of A to be rejecting, and vice versa.

Lemma 2.6. *If A is a doubly stochastic matrix and X - a vector, then* $\max(X) \geq \max(AX)$ *and* $\min(X) \leq \min(AX)$.

Theorem 2.7. *For every natural positive n, a language $L_n = a_1^* a_2^* \dots a_n^*$ is recognizable by some PRA-C with alphabet $\{a_1, a_2, \dots, a_n\}$.*

Proof. We construct a PRA-C with $n+1$ states, q_0 being the initial state, corresponding to probability distribution vector $\begin{pmatrix} 1 & 0 & \dots & 0 \end{pmatrix}^T$. The transition function is determined by $(n+1) \times (n+1)$ matrices

$$V_{a_1} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \frac{1}{n} & \dots & \frac{1}{n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \frac{1}{n} & \dots & \frac{1}{n} \end{pmatrix}, V_{a_2} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & \dots & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & \dots & 0 \\ 0 & 0 & \frac{1}{n-1} & \dots & \frac{1}{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \frac{1}{n-1} & \dots & \frac{1}{n-1} \end{pmatrix}, \dots, V_{a_n} = \begin{pmatrix} \frac{1}{n} & \dots & \frac{1}{n} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \frac{1}{n} & \dots & \frac{1}{n} & 0 \\ 0 & \dots & 0 & 1 \end{pmatrix}.$$

The accepting states are $q_0 \dots q_{n-1}$, the only rejecting state is q_n. We prove, that the automaton recognizes the language L_n.

Case $\omega \in L_n$. All $\omega \in L_n$ are accepted with probability 1.

Case $\omega \notin L_n$. Consider k such that $\omega = \omega_1 \sigma \omega_2$, $|\omega_1| = k$, $\omega_1 \in L_n$ and $\omega_1 \sigma \notin L_n$. Since all one-letter words are in L_n, $k > 0$. Let $a_t = [\omega]_k$ and $a_s = \sigma$. So we have $s < t$, $1 \leq s \leq n-1$, $2 \leq t \leq n$. The word $\omega_1 a_s$ is accepted with probability $1 - \frac{t-s}{t(n-s+1)}$. By Lemma 2.6, since $\frac{t-s}{t(n-s+1)} < \frac{1}{t}$, reading the symbols succeeding $\omega_1 a_s$ will not increase accepting probability. Therefore, to find maximum accepting probability for words not in L_n, we have to maximize $1 - \frac{t-s}{t(n-s+1)}$, where $s < t$, $1 \leq s \leq n-1$, $2 \leq t \leq n$. We get that the automaton recognizes the language with interval $\left(1 - \frac{1}{\left[\left(\frac{n}{2}\right)^2\right]+n+1},\ 1\right)$. (By Theorem 2.3, L_n can be recognized with probability $1 - \varepsilon$). □

Corollary 2.8. *Quantum finite automata with mixed states (model of Nayak, [N 99]) recognize $L_n = a_1^* a_2^* \dots a_n^*$ with probability $1 - \varepsilon$.*

Proof. This comes from the fact, that matrices $V_{a_1}, V_{a_2}, \dots, V_{a_n}$ from the proof of Theorem 2.7 (as well as tensor powers of those matrices) all have unitary prototypes (see Definition 5.1). □

Now we introduce a general class of regular languages not recognizable by PRA-C.

Definition 2.9. *We say that a regular language is of type $(*)$ if the following is true for the minimal deterministic automaton recognizing this language: Exist three states q, q_1, q_2, exist words x, y such that $q_1 \neq q_2$; $qx = q_1$, $qy = q_2$; $\forall t \in (x,y)^* \ \exists t_1 \in (x,y)^* \ q_1 t t_1 = q_1$; $\forall t \in (x,y)^* \ \exists t_2 \in (x,y)^* \ q_2 t t_2 = q_2$.*

We say that a regular language is of type $(*')$ if the following is true for the minimal deterministic automaton recognizing this language: Exist three states q,

q_1, q_2, *exist words* x, y *such that* $q_1 \neq q_2$; $qx = q_1$, $qy = q_2$; $q_1x = q_1$, $q_1y = q_1$; $q_2x = q_2$, $q_2y = q_2$.
We say that a regular language is of type $(*'')$ *if the following is true for the minimal deterministic automaton recognizing this language: Exist two states* q_1, q_2, *exist words* x, y *such that* $q_1 \neq q_2$; $q_1x = q_2$, $q_2x = q_2$; $q_2y = q_1$.

Type $(*'')$ languages are exactly those languages that violate the partial order condition of [BP 99].

Lemma 2.10. *If* A *is a deterministic finite automaton with a set of states* Q *and alphabet* Σ, *then* $\forall q \in Q\ \forall x \in \Sigma^*\ \exists k > 0\ qx^k = qx^{2k}$.

Lemma 2.11. *A regular language is of type* $(*)$ *iff it is of type* $(*')$ *or type* $(*'')$.

Proof. 1) If a language is of type $(*')$, it is of type $(*)$. Obvious.
2) If a language is of type $(*'')$, it is of type $(*)$. Consider a language of type $(*'')$ with states q_1'', q_2'' and words x'', y''. To build construction of type $(*)$, we take $q = q_1 = q_1''$, $q_2 = q_2''$, $x = x''y''$, $y = x''$. That forms transitions $qx = q_1$, $qy = q_2$, $q_1x = q_1$, $q_1y = q_2$, $q_2x = q_1$, $q_2y = q_2$. We have satisfied all the rules of $(*)$.
3) If a language is of type $(*)$, it is of type $(*')$ or $(*'')$. Consider a language whose minimal deterministic automaton has construction $(*)$. By Lemma 2.10, $\exists s \exists a\ q_1x^a = q_s$ and $q_sx^a = q_s$; $\exists t \exists b\ q_1y^b = q_t$ and $q_ty^b = q_t$; $\exists u \exists c\ q_2x^c = q_u$ and $q_ux^c = q_u$; $\exists v \exists d\ q_2y^d = q_v$ and $q_vy^d = q_v$. If $q_1 \neq q_s$, by the rules of $(*)$, $\exists z\ q_sz = q_1$. Therefore the language is of type $(*'')$. If $q_2 \neq q_u$, by the rules of $(*)$, $\exists z\ q_uz = q_2$, and the language is of type $(*'')$. Likewise, if $q_1 \neq q_t$ or $q_2 \neq q_v$, the language is of type $(*'')$. If $q_1 = q_s = q_t$ and $q_2 = q_u = q_v$, we have $qx^a = q_1$, $qy^d = q_2$, $q_1x^a = q_1y^b = q_1$, $q_2x^c = q_2y^d = q_2$. We get the construction $(*')$ if we take $x' = x^{ac}$, $y' = y^{bd}$. □

We are going to prove that every language of type $(*)$ is not recognizable by any PRA-C. For this purpose, we use several definitions from the theory of finite Markov chains ([KS 76]).

Definition 2.12. *A state* q_j *is accessible from* q_i *(denoted* $q_i \to q_j$*) if there is a positive probability to get from* q_i *to* q_j *(possibly in several steps).*

Definition 2.13. *States* q_i *and* q_j *communicate (denoted* $q_i \leftrightarrow q_j$*) if* $q_i \to q_j$ *and* $q_j \to q_i$.

Definition 2.14. *A Markov chain is called doubly stochastic, if its transition matrix is a doubly stochastic matrix.*

We recall the following theorem from the theory of finite Markov chains:

Theorem 2.15. *If a Markov chain with a matrix* A *is irreducible and aperiodic:*
a) it has a unique stationary distribution Z;
b) $\lim_{n\to\infty} A^n = (Z, \ldots, Z)$;
c) $\forall X \lim_{n\to\infty} A^nX = Z$.

Several facts about doubly stochastic matrices follow from this theorem.

Corollary 2.16. *If a doubly stochastic Markov chain with an $m \times m$ matrix A is irreducible and aperiodic,*

$$a) \lim_{n\to\infty} A^n = \begin{pmatrix} \frac{1}{m} & \cdots & \frac{1}{m} \\ \cdots & \cdots & \cdots \\ \frac{1}{m} & \cdots & \frac{1}{m} \end{pmatrix}; \; b) \forall X \lim_{n\to\infty} A^n X = \begin{pmatrix} \frac{1}{m} \\ \cdots \\ \frac{1}{m} \end{pmatrix}.$$

Lemma 2.17. *If M is a doubly stochastic Markov chain with a matrix A, then $\forall q \; q \to q$.*

Corollary 2.18. *Suppose A is a doubly stochastic matrix. Then exists $k > 0$, such that $\forall i \; (A^k)_{i,i} > 0$.*

Lemma 2.19. *If M is a doubly stochastic Markov chain and $q_a \to q_b$, then $q_a \leftrightarrow q_b$.*

Now, using the facts above, we can prove that any language of type $(*)$ is not recognizable by PRA-C.

Lemma 2.20. *If a regular language is of type $(*')$, it is not recognizable by any PRA-C.*

Proof. Assume from the contrary, that A is a PRA-C automaton which recognizes a language $L \subset \Sigma^*$ of type $(*')$.

Since L is of type $(*')$, it is recognized by a deterministic automaton D which has three states q, q_1, q_2 such that $q_1 \neq q_2$, $qx = q_1$, $qy = q_2$, $q_1x = q_1$, $q_1y = q_1$, $q_2x = q_2$, $q_2y = q_2$, where $x, y \in \Sigma^*$. Furthermore, exists $\omega \in \Sigma^*$ such that $q_0\omega = q$, where q_0 is an initial state of D, and exists a word $z \in \Sigma^*$, such that $q_1z = q_{acc}$ if and only if $q_2z = q_{rej}$, where q_{acc} is an accepting state and q_{rej} is a rejecting state of D. Without loss of generality we assume that $q_1z = q_{acc}$ and $q_2z = q_{rej}$.

The transition function of the automaton A is determined by doubly stochastic matrices $V_{\sigma_1}, \ldots, V_{\sigma_n}$. The words from the construction $(*')$ are $x = \sigma_{i_1} \ldots \sigma_{i_k}$ and $y = \sigma_{j_1} \ldots \sigma_{j_s}$. The transitions induced by words x and y are determined by doubly stochastic matrices $X = V_{\sigma_{i_k}} \ldots V_{\sigma_{i_1}}$ and $Y = V_{\sigma_{j_s}} \ldots V_{\sigma_{j_1}}$. Similarly, the transitions induced by words ω and z are determined by doubly stochastic matrices W and Z. By Corollary 2.18, exists $K > 0$, such that

$$\forall i \; (X^K)_{i,i} > 0 \text{ and } (Y^K)_{i,i} > 0. \tag{3}$$

Consider a relation between the states of the automaton defined as $R = \{(q_i, q_j) \mid q_i \xrightarrow{(x^K, y^K)^*} q_j\}$. By (3), this relation is reflexive. By Lemma 2.19, the relation R is symmetric.

Surely R is transitive. Therefore all states of A may be partitioned into equivalence classes $[q_0], [q_{i_1}], \ldots, [q_{i_n}]$. Let us renumber the states of A in such a way, that states from one equivalence class have consecutive numbers. First come the states in $[q_0]$, then in $[q_{i_1}]$, etc.

Consider the word $x^K y^K$. The transition induced by this word is determined by a doubly stochastic matrix $C = Y^K X^K$. We prove the following proposition. States q_a and q_b are in one equivalence class if and only if $q_a \to q_b$ with matrix C. Suppose $q_a \to q_b$. Then $(q_a, q_b) \in R$, and q_a, q_b are in one equivalence class. Suppose q_a, q_b are in one equivalence class. Then

$$q_a \xrightarrow{\xi_1} q_{i_1}, q_{i_1} \xrightarrow{\xi_2} q_{i_2}, \ldots, q_{i_{k-1}} \xrightarrow{\xi_k} q_b, \text{ where } \xi_s \in \{x^K, y^K\}. \qquad (4)$$

By (3), $q_i \xrightarrow{x^K} q_i$ and $q_j \xrightarrow{y^K} q_j$. Therefore, if $q_i \xrightarrow{x^K} q_j$, then $q_i \xrightarrow{x^K y^K} q_j$, and again, if $q_i \xrightarrow{y^K} q_j$, then $q_i \xrightarrow{x^K y^K} q_j$. That transforms (4) to $q_a \xrightarrow{(x^K y^K)^t} q_b$, where $t > 0$. We have proved the proposition.

By the proved proposition, due to the renumbering of states, matrix C is a block diagonal matrix, where each block corresponds to an equivalence class of the relation R. Let us identify these blocks as $C_0, C_1, \ldots, C_n$. By (3), a Markov chain with matrix C is aperiodic. Therefore each block C_r corresponds to an aperiodic irreducible doubly stochastic Markov chain with states $[q_{i_r}]$. By Corollary 2.16, $\lim_{m\to\infty} C^m = J$, J is a block diagonal matrix, where for each $(p \times p)$ block C_r $(C_r)_{i,j} = \frac{1}{p}$. Relation $q_i \xrightarrow{(y^K)^*} q_j$ is a subrelation of R, therefore Y^K is a block diagonal matrix with the same block ordering and sizes as C and J. (This does not eliminate possibility that some block of Y^K is constituted of smaller blocks, however.) Therefore $JY^K = J$, and $\lim_{m\to\infty} Z(Y^K X^K)^m W = \lim_{m\to\infty} Z(Y^K X^K)^m Y^K W = ZJW$. So $\forall \varepsilon > 0\ \exists m\ \left\| \left(Z(Y^K X^K)^m W - Z(Y^K X^K)^m Y^K W\right) Q_0 \right\| < \varepsilon$. However, by construction $(*')$, $\forall k\ \forall m\ \omega(x^k y^k)^m z \in L$ and $\omega y^k (x^k y^k)^m z \notin L$. This requires existence of $\varepsilon > 0$, such that $\forall m\ \left\| \left(Z(Y^K X^K)^m W - Z(Y^K X^K)^m Y^K W\right) Q_0 \right\| > \varepsilon$. This is a contradiction. □

Lemma 2.21. *If a regular language is of type $(*'')$, it is not recognizable by any PRA-C.*

Proof. Proof is nearly identical to that of Lemma 2.20.

Theorem 2.22. *If a regular language is of type $(*)$, it is not recognizable by any PRA-C.*

Proof. By Lemmas 2.11, 2.20, 2.21.

We proved (Lemma 2.11) that the construction of type $(*)$ is a generalization the construction proposed by [BP 99]. Also it can be easily noticed, that the type $(*)$ construction is a generalization of construction proposed by [AKV 00]. (Constructions of [BP 99] and [AKV 00] characterize languages, not recognized by measure-many quantum finite automata of [KW 97].)

Corollary 2.23. *Languages $(a,b)^*a$ and $a(a,b)^*$ are not recognized by PRA-C.*

Proof. Both languages are of type $(*)$.

3 1-Way Probabilistic Reversible DH-Automata

Definition 3.1. *The definition differs from one for PRA-C (Definition 2.1) by the following: languages are recognized according to Definition 1.3.*

It is easy to see that the class of languages recognized by PRA-C is a proper subclass of languages recognized by PRA-DH. For example, the language a(a,b)* is recognizable by PRA-DH. However, the following theorem holds:

Theorem 3.2. *Language (a,b)*a is not recognized by PRA-DH.*

Proof. Assume from the contrary that such automaton exists. While reading any sequence of a and b, this automaton can halt only with some probability p strictly less then 1, so accepting and rejecting probabilities may differ only by 1-p, because any word belonging to the language is not dependent on any prefix. Therefore for each $\varepsilon > 0$ we can find that after reading a prefix of certain length, the total probability to halt while continue reading the word is less then ε. In this case we can apply similar techniques as in the proof of Lemma 2.20. □

4 Alternative Approach to Finite Reversible Automata and 1.5-Way Probabilistic Reversible Automata

Let us consider an automaton $A' = (Q, \Sigma, q_0, Q_F, \delta')$ that can be obtained from a probabilistic automaton $A = (Q, \Sigma, q_0, Q_F, \delta)$ by specifying $\delta'(q, \sigma, q') = \delta(q', \sigma, q)$ for all q', σ and q. If A' is valid probabilistic automaton then we can call A and A' probabilistic reversible automata.

Definition 4.1. *An automaton of some type is called **weakly reversible** if the reverse of its transition function corresponds to the transition function of a valid automaton of the same type.*

Note: in case of deterministic automaton where $\delta : Q \times \Gamma \times Q \longrightarrow \{0, 1\}$ this property means that A' is still deterministic automaton, not nondeterministic. In case of one-way automata it is easy to check that this definition is equivalent to the one in Section 2. We give an example that illustrates that in case of 1.5-way automata these definitions are different.

Definition 4.2. *1.5-way probabilistic weakly reversible C-automaton $A = (Q, \Sigma, q_0, Q_F, \delta)$ is specified by Q, Σ, q_0, Q_F defined as in 1-way PRA-C Definition 2.1, and a transition function $\delta : Q \times \Gamma \times Q \times D \longrightarrow \mathbb{R}_{[0,1]}$, where Γ defined as in 1-way PRA-C definition and $D = \{0, 1\}$ denotes whether automaton stays on the same position or moves one letter ahead on the input tape. Furthermore, transition function satisfies the following requirements:*

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \sum_{q \in Q, d \in D} \delta(q_1, \sigma_1, q, d) = 1;$$

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \sum_{q \in Q, d \in D} \delta(q, \sigma_1, q_1, d) = 1$$

Definition 4.3. *1.5-way probabilistic reversible C-automaton* $A = (Q, \Sigma, q_0, Q_F, \delta)$ *is specified by* Q, Σ, q_0, Q_F *defined as in 1-way PRA-C Definition 2.1, and a transition function* $\delta : Q \times \Gamma \times Q \times D \longrightarrow \mathbb{R}_{[0,1]}$, *where* Γ *defined as in 1-way PRA-C definition and* $D = \{0, 1\}$ *denotes whether automaton stays on the same position or moves one letter ahead on the input tape. Furthermore, transition function satisfies the following requirements:*

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \sum_{q \in Q, d \in D} \delta(q_1, \sigma_1, q, d) = 1;$$

$$\forall (q_1, \sigma_1, \sigma_2) \in Q \times \Gamma^2 \sum_{q \in Q} \delta(q, \sigma_1, q_1, 0) + \sum_{q \in Q, \sigma \in \Gamma} \delta(q, \sigma_2, q_1, 1) = 1$$

Theorem 4.4. *Language* $(a,b)^*a$ *is recognizable by 1.5-way weakly reversible PRA-C.*

Proof. The $Q = \{q_0, q_1\}$, $Q_F = \{q_1\}$, δ is defined as follows; $\delta(q_0, a, q_0, 0) = \frac{1}{2}$, $\delta(q_0, a, q_1, 1) = \frac{1}{2}$, $\delta(q_1, a, q_0, 0) = \frac{1}{2}$, $\delta(q_1, a, q_1, 1) = \frac{1}{2}$, $\delta(q_0, b, q_0, 1) = \frac{1}{2}$, $\delta(q_0, b, q_1, 0) = \frac{1}{2}$, $\delta(q_1, b, q_0, 1) = \frac{1}{2}$, $\delta(q_1, b, q_1, 0) = \frac{1}{2}$, $\delta(q_0, \$, q_0, 1) = 1$, $\delta(q_1, \$, q_1, 1) = 1$. It easy to check that such automaton moves ahead according to the transition of the following deterministic automaton; $\delta(q_0, a, q_1, 1) = 1$, $\delta(q_1, a, q_1, 1) = 1$, $\delta(q_0, b, q_0, 1) = 1$, $\delta(q_1, b, q_0, 1) = 1$, $\delta(q_0, \$, q_0, 1) = 1$, $\delta(q_1, \$, q_1, 1) = 1$. So the probability of wrong answer is 0. The probability to be at the m-th position of the input tape after n steps of calculation for $m \leq n$ is C_n^m. Therefore it is necessary no more then $O(n * \log(p))$ steps to reach the end of the word of length n (and so obtain correct answer) with probability $1 - \frac{1}{p}$. □

5 A Classification of Reversible Automata

We propose the following classification for finite 1-way reversible automata:

	C-Automata	DH-Automata
Deterministic Automata	Permutation Automata [HS 66,T 68] (DRA-C)	Reversible Finite Automata [AF 98] (DRA-DH)
Quantum Automata with Pure States	Measure-Once Quantum Finite Automata [MC 97] (QRA-P-C)	Measure-Many Quantum Finite Automata [KW 97] (QRA-P-DH)
Probabilistic Automata	Probabilistic Reversible C-Automata (PRA-C)	Probabilistic Reversible DH-Automata (PRA-DH)
Quantum Finite Automata with Mixed States	not considered yet (QRA-M-C)	Enhanced Quantum Finite Automata [N 99] (QRA-M-DH)

Language class problems are solved for DRA-C, DRA-DH, QRA-P-C, for the rest types they are still open. Every type of DH-automata may simulate the corresponding type of C-automata.

In general, language classes recognized by C-automata are closed under boolean operations (though this is open for QRA-M-C), while DH-automata are not (though this is open for QRA-M-DH and possibly for PRA-DH).

Definition 5.1. *We say that a unitary matrix U is a* prototype *for a doubly stochastic matrix S, if $\forall i,j\ |U_{i,j}|^2 = S_{i,j}$.*

Not every doubly stochastic matrix has a unitary prototype. Such matrix is, for example, $\begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$. In Introduction, we demonstrated some relation between PRA-C and QRA-M-DH (and hence, QRA-M-C). However, due to the example above, we do not know exactly, whether every PRA-C can be simulated by QRA-M-C, or whether every PRA-DH can be simulated by QRA-M-DH.

Theorem 5.2. *If all matrices of a PRA-C have unitary prototypes, then the PRA-C may be simulated by a QRA-M-C and by a QRA-M-DH.*
If all matrices of a PRA-DH have unitary prototypes, then the PRA-DH may be simulated by a QRA-M-DH.

References

[ABFK 99] A. Ambainis, R. Bonner, R. Freivalds, A. Ķikusts. Probabilities to Accept Languages by Quantum Finite Automata. *COCOON 1999, Lecture Notes in Computer Science*, 1999, Vol. 1627, pp. 174-183. http://arxiv.org/abs/quant-ph/9904066

[AF 98] A. Ambainis, R. Freivalds. 1-Way Quantum Finite Automata: Strengths, Weaknesses and Generalizations. *Proc. 39th FOCS*, 1998, pp. 332-341. http://arxiv.org/abs/quant-ph/9802062

[AKV 00] A. Ambainis, A. Ķikusts, M. Valdats. On the Class of Languages Recognizable by 1-Way Quantum Finite Automata. *STACS 2001, Lecture Notes in Computer Science*, 2001, Vol. 2010, pp. 75-86. http://arxiv.org/abs/quant-ph/0009004

[BP 99] A. Brodsky, N. Pippenger. Characterizations of 1-Way Quantum Finite Automata. http://arxiv.org/abs/quant-ph/9903014

[HS 66] J. Hartmanis, R. E. Stearns. Algebraic Structure Theory of Sequential Machines. *Prentice Hall*, 1966.

[KS 76] J. G. Kemeny and J. L. Snell. Finite Markov Chains. *Springer Verlag*, 1976.

[KW 97] A. Kondacs, J. Watrous. On The Power of Quantum Finite State Automata. *Proc. 38th FOCS*, 1997, pp. 66-75.

[MC 97] C. Moore, J. P. Crutchfield. Quantum Automata and Quantum Grammars. *Theoretical Computer Science*, 2000, Vol. 237(1-2), pp. 275-306. http://arxiv.org/abs/quant-ph/9707031

[N 99] A. Nayak. Optimal Lower Bounds for Quantum Automata and Random Access Codes. *Proc. 40th FOCS*, 1999, pp. 369-377. http://arxiv.org/abs/quant-ph/9904093

[R 63] M. O. Rabin. Probabilistic Automata. *Information and Control*, 1963, Vol. 6(3), pp. 230-245.

[T 68] G. Thierrin. Permutation Automata. *Mathematical Systems Theory*, Vol. 2(1), pp. 83-90

Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Singapore
Tokyo

Václav Hlaváč Keith G. Jeffery
Jiří Wiedermann (Eds.)

SOFSEM 2000: Theory and Practice of Informatics

27th Conference on Current Trends
in Theory and Practice of Informatics
Milovy, Czech Republic, November 25 – December 2, 2000
Proceedings

Springer

Subreferees

Peter Borovanský
Michael Butler
David R. Gilbert
Dušan Guller
Neil Henderson
Juraj Hromkovič
Petr Klán
Ralf Klasing
Ivan Kopeček
Dariusz Kowalski
Mojmír Křetínský
Antonín Kučera
Adam Malinowski
Luboš Popelínský
Zdenko Staníček
Ján Šturc
Miroslav Tůma
Walter Unger
Pavol Voda
Jana Zvárová

Organization

SOFSEM 2000 is organized by

CLRC RAL, Chilton, Didcot, Oxon, UK
Czech Technical University, Prague, Czech Republic
Czech Society for Computer Science
Faculty of Informatics, Masaryk University, Brno, Czech Republic
Institute of Computer Science, Academy of Science, Prague, Czech Republic
Institute of Computer Science, Masaryk University, Brno, Czech Republic

in cooperation with

Czech ACM Chapter
Czechoslovak Chapter of the IEEE Computer Society
Slovak Society for Computer Science

Organizing Committee

Jan Staudek, *chair*
Miroslav Bartošek
Petr Hanáček
Dana Komárková
Zdeněk Malčík
Tomáš Pitner
Jaromír Skřivan
Petr Sojka
Tomáš Staudek

Sponsoring Institutions

ApS Brno, s.r.o.
Compaq Computer, s.r.o.
ERCIM, the European Research Consortium for Informatics and Mathematics
Hewlett Packard, s.r.o.
IBM Czech Republic, s.r.o.
Oracle Czech, s.r.o.

Table of Contents

Computational Perception

Soft Computing

CONTRIBUTED PAPERS

Quantum Pushdown Automata*

Marats Golovkins

Institute of Mathematics and Computer Science
University of Latvia, Raiņa bulv. 29, Rīga, Latvia
marats@cclu.lv

Abstract. Quantum finite automata, as well as quantum pushdown automata were first introduced by C. Moore, J. P. Crutchfield [13]. In this paper we introduce the notion of quantum pushdown automata (QPA) in a non-equivalent way, including unitarity criteria, by using the definition of quantum finite automata of [11]. It is established that the unitarity criteria of QPA are not equivalent to the corresponding unitarity criteria of quantum Turing machines [4]. We show that QPA can recognize every regular language. Finally we present some simple languages recognized by QPA, two of them are not recognizable by deterministic pushdown automata and one seems to be not recognizable by probabilistic pushdown automata as well.

1 Introduction

Nobel prize winner physicist R. Feynman asked in 1982, what effects may have the principles of quantum mechanics on computation [8]. He gave arguments that it may require exponential time to simulate quantum mechanical processes on classical computers. This served as a basis to the opinion that quantum computers may have advantages versus classical ones. It was in 1985, when D. Deutsch introduced the notion of quantum Turing machine [6] and proved that quantum Turing machines compute the same recursive functions as classical deterministic Turing machines do. P. Shor discovered that by use of quantum algorithms it is possible to factorize large integers and compute discrete logarithms in a polynomial time [14], what resulted into additional interest in quantum computing and attempts to create quantum computers. First steps have been made to this direction, and first quantum computers which memory is limited by a few quantum bits have been constructed.

For the analysis of the current situation in quantum computation and information processing and main open issues one could see [9].

Opposite to quantum Turing machines, quantum finite automata (QFA) represent the finite model of quantum computation. QFA were first introduced by [13] (measure-once QFA), which were followed by a more elaborated model

* Research partially supported by the Latvian Council of Science, grant 96-0282 and grant for Ph.D. students; European Commission, contract IST-1999-11234; Swedish institute, project ML2000.

V. Hlaváč, K. G. Jeffery, and J. Wiedermann (Eds.): SOFSEM 2000, LNCS 1963, pp. 336–346, 2000.

of [11] (measure-many quantum finite automata). Since then QFA have been studied a lot, various properties of these automata are considered in [2,3,5,15].

The purpose of this paper is to introduce a quantum counterpart of pushdown automata, the next most important model after finite automata and Turing machines. The first definition of quantum pushdown automata was suggested by [13], but here the authors actually deal with the so-called generalized quantum pushdown automata, which evolution does not have to be unitary. However a basic postulate of quantum mechanics imposes a strong constraint on any quantum machine model: it has to be unitary, otherwise it is questionable whether we can speak about *quantum* machine. That's why it was considered necessary to re-introduce quantum pushdown automata by giving a definition which would conform unitarity requirement. Such definition would enable us to study the properties of quantum pushdown automata.

The following notations will be used further in the paper: z^* is the complex conjugate of a complex number z; U^* is the Hermitian conjugate of a matrix U; I is the identity matrix; ϵ is empty word.

Definition 1. Matrix U is called *unitary*, if $UU^* = U^*U = I$.

If U is a finite matrix, then $UU^* = I$ iff $U^*U = I$. However this is not true for infinite matrices:

Example 1.

$$U = \begin{pmatrix} \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \dots \\ \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

Here $U^*U = I$ but $UU^* \neq I$.

Lemma 1. *The matrix U is unitary iff $U^*U = I$ and its rows are normalized.*

This result is very similar to Lemma 1 of [7].

2 Quantum Pushdown Automata

Definition 2. A *quantum pushdown automaton* (QPA) $A = (Q, \Sigma, T, q_0, Q_a, Q_r, \delta)$ is specified by a finite set of states Q, a finite input alphabet Σ and a stack alphabet T, an initial state $q_0 \in Q$, sets $Q_a \subset Q$, $Q_r \subset Q$ of accepting and rejecting states, respectively, with $Q_a \cap Q_r = \emptyset$, and a transition function

$$\delta : Q \times \Gamma \times \Delta \times Q \times \{\downarrow, \rightarrow\} \times \Delta^* \longrightarrow \mathbb{C}_{[0,1]},$$

where $\Gamma = \Sigma \cup \{\#, \$\}$ is the input tape alphabet of A and $\#, \$$ are end-markers not in Σ, $\Delta = T \cup \{Z_0\}$ is the working stack alphabet of A and $Z_0 \notin T$ is the stack base symbol; $\{\downarrow, \rightarrow\}$ is the set of directions of input tape head. The automaton

must satisfy *conditions of well-formedness*, which will be expressed below. Furthermore, the transition function is restricted to a following requirement:
If $\delta(q, \alpha, \beta, q', d, \omega) \neq 0$, then

1. $|\omega| \leq 2$,
2. if $|\omega| = 2$, then $\omega_1 = \beta$,
3. if $\beta = Z_0$, then $\omega \in Z_0 T^*$,
4. if $\beta \neq Z_0$, then $\omega \in T^*$.

Here ω_1 is the first symbol of a word ω. Definition 2 utilizes that of classical pushdown automata from [10].

Let us assume that an automaton is in a state q, its input tape head is above a symbol α and the stack head is above a symbol β. Then the automaton undertakes the following actions with an amplitude $\delta(q, \alpha, \beta, q', d, \omega)$:

1. goes into the state q',
2. if $d = ' \rightarrow '$, moves the input tape head one cell forward,
3. takes out of the stack the symbol β (deletes it and moves the stack head one cell backwards),
4. starting with the first empty cell, puts into the stack the string ω, moving the stack head $|\omega|$ cells forward.

Definition 3. The *configuration* of a pushdown automaton is a pair $|c\rangle = |\nu_i q_j \nu_k, \omega_l\rangle$, where the automaton is in a state $q_j \in Q$, $\nu_i \nu_k \in \#\Sigma^*\$$ is a finite word on the input tape, $\omega_l \in Z_0 T^*$ is a finite word on the stack tape, the input tape head is above the first symbol of the word ν_k and the stack head is above the last symbol of the word ω_l.

We shall denote by C the set of all configurations of a pushdown automaton. The set C is countably infinite. Every configuration $|c\rangle$ denotes a basis vector in the Hilbert space $H_A = l_2(C)$. Therefore a global state of A in the space H_A has a form $|\psi\rangle = \sum_{c \in C} \alpha_c |c\rangle$, where $\sum_{c \in C} |\alpha_c|^2 = 1$ and $\alpha_c \in \mathbb{C}$ denotes the amplitude of a configuration $|c\rangle$. If an automaton is in its global state (superposition) $|\psi\rangle$, then its further step is equivalent to the application of a linear operator (evolution) U_A over the space H_A.

Definition 4. A linear operator U_A is defined as follows:

$$U_A|\psi\rangle = \sum_{c \in C} \alpha_c U_A |c\rangle .$$

If a configuration $c = |\nu_i q_j \sigma \nu_k, \omega_l \tau\rangle$, then

$$U_A|c\rangle = \sum_{(q,d,\omega) \in Q \times \{\downarrow, \rightarrow\} \times \Delta^*} \delta(q_j, \sigma, \tau, q, d, \omega) | f(|c\rangle, d, q), \omega_l \omega\rangle$$

where

$$f(|\nu_i q_j \sigma \nu_k, \omega_l \tau\rangle, d, q) = \begin{cases} \nu_i q \sigma \nu_k, \text{if } d = \downarrow \\ \nu_i \sigma q \nu_k, \text{if } d = \rightarrow . \end{cases}$$

Remark 1. Although a QPA evolution operator matrix is infinite, it has a finite number of nonzero elements in each row and column, as it is possible to reach only a finite number of other configurations from a given configuration within one step, all the same, within one step the given configuration is reachable only from a finite number of different configurations.

We can speak about a *quantum* pushdown automaton only if its evolution operator is unitary. However, evolution operator matrix is infinite, so we need some criteria (well-formedness conditions) to verify its unitarity.

Well-formedness conditions 1.

1. Local probability condition. $\forall(q_1,\sigma_1,\tau_1)\in Q\times\Gamma\times\Delta$

$$\sum_{(q,d,\omega)\in Q\times\{\downarrow,\rightarrow\}\times\Delta^*}|\delta(q_1,\sigma_1,\tau_1,q,d,\omega)|^2=1\,. \tag{1}$$

2. Orthogonality of column vectors condition. For all triples $(q_1,\sigma_1,\tau_1)\neq(q_2,\sigma_1,\tau_2)$ in $Q\times\Gamma\times\Delta$

$$\sum_{(q,d,\omega)\in Q\times\{\downarrow,\rightarrow\}\times\Delta^*}\delta^*(q_1,\sigma_1,\tau_1,q,d,\omega)\delta(q_2,\sigma_1,\tau_2,q,d,\omega)=0\,. \tag{2}$$

3. Row vectors norm condition. $\forall(q_1,\sigma_1,\sigma_2,\tau_1,\tau_2)\in Q\times\Gamma^2\times\Delta^2$

$$\sum_{(q,\tau,\omega)\in Q\times\Delta\times\{\varepsilon,\tau_2,\tau_1\tau_2\}}|\delta(q,\sigma_1,\tau,q_1,\rightarrow,\omega)|^2+|\delta(q,\sigma_2,\tau,q_1,\downarrow,\omega)|^2=1\,. \tag{3}$$

4. Separability condition I. $\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_1,\tau_2)\in Q\times\Gamma\times\Delta,\forall\tau_3\in\Delta$

$$\sum_{(q,d,\tau)\in Q\times\{\downarrow,\rightarrow\}\times\Delta}\delta^*(q_1,\sigma_1,\tau_1,q,d,\tau)\delta(q_2,\sigma_1,\tau_2,q,d,\tau_3\tau)+$$
$$+\sum_{(q,d)\in Q\times\{\downarrow,\rightarrow\}}\delta^*(q_1,\sigma_1,\tau_1,q,d,\varepsilon)\delta(q_2,\sigma_1,\tau_2,q,d,\tau_3)=0\,, \tag{4}$$

$$\sum_{(q,d)\in Q\times\{\downarrow,\rightarrow\}}\delta^*(q_1,\sigma_1,\tau_1,q,d,\varepsilon)\delta(q_2,\sigma_1,\tau_2,q,d,\tau_2\tau_3)=0\,. \tag{5}$$

5. Separability condition II. $\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_2,\tau_2)\in Q\times\Gamma\times\Delta$

$$\sum_{(q,\omega)\in Q\times\Delta^*}\delta^*(q_1,\sigma_1,\tau_1,q,\downarrow,\omega)\delta(q_2,\sigma_2,\tau_2,q,\rightarrow,\omega)=0\,. \tag{6}$$

6. Separability condition III. $\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_2,\tau_2)\in Q\times\Gamma\times\Delta$, $\forall\tau_3\in\Delta$, $\forall d_1,d_2\in\{\downarrow,\rightarrow\}$, $d_1\neq d_2$

$$\sum_{(q,\tau)\in Q\times\Delta}\delta^*(q_1,\sigma_1,\tau_1,q,d_1,\tau)\delta(q_2,\sigma_2,\tau_2,q,d_2,\tau_3\tau)+$$
$$+\sum_{q\in Q}\delta^*(q_1,\sigma_1,\tau_1,q,d_1,\varepsilon)\delta(q_2,\sigma_2,\tau_2,q,d_2,\tau_3)=0\,, \tag{7}$$

$$\sum_{q \in Q} \delta^*(q_1, \sigma_1, \tau_1, q, d_1, \varepsilon)\delta(q_2, \sigma_2, \tau_2, q, d_2, \tau_2\tau_3) = 0\,. \tag{8}$$

Lemma 2. *1. The columns system of a QPA evolution matrix is normalized iff the condition (1), i. e., local probability condition, is satisfied.*
2. The columns system of a QPA evolution matrix is orthogonal iff the conditions (2,4,5,6,7,8), i. e., orthogonality of column vectors and separability conditions, are satisfied.
3. The rows system of a QPA evolution matrix is normalized iff the condition (3), i. e., row vectors norm condition, is satisfied.

Theorem 1. *Well-formedness conditions 1 are satisfied iff the evolution operator U_A is unitary.*

Proof. Lemma 2 implies that Well-formedness conditions 1 are satisfied iff the columns of the evolution matrix are orthonormal and rows are normalized. In compliance with Lemma 1, columns are orthonormal and rows are normalized iff the matrix is unitary. □

Remark 2. Well-formedness conditions 1 contain the requirement that rows system has to be normalized, which is not necessary in the case of quantum Turing machine [4]. Here is taken into account the fact that the evolution of QPA can violate the unitarity requirement if the row vectors norm condition is omitted.

Example 2. A QPA, whose evolution matrix columns are orthonormal, however the evolution is not unitary.

$$\begin{array}{ll} Q = \{q\},\ \Sigma = \{1\},\ T = \{1\}\,. & \\ \delta(q, \#, Z_0, q, \rightarrow, Z_0 1) = 1\,, & \delta(q, \#, 1, q, \rightarrow, 11) = 1\,, \\ \delta(q, 1, Z_0, q, \rightarrow, Z_0 1) = 1\,, & \delta(q, 1, 1, q, \rightarrow, 11) = 1\,, \\ \delta(q, \$, Z_0, q, \rightarrow, Z_0 1) = 1\,, & \delta(q, \$, 1, q, \rightarrow, 11) = 1\,, \end{array}$$

other values of arguments yield $\delta = 0$.

By Well-formedness conditions 1, the columns of the evolution matrix are orthonormal, but the matrix is not unitary, because the norm of the rows specified by the configurations $|\omega, Z_0\rangle$ is 0.

Even in a case of trivial QPA, it is a cumbersome task to check all the conditions of well-formedness 1. It is possible to relax the conditions slightly by introducing a notion of *simplified* QPA.

Definition 5. We shall say that a QPA is *simplified*, if there exists a function $D : Q \longrightarrow \{\downarrow, \rightarrow\}$, and $\delta(q_1, \sigma, \tau, q, d, \omega) = 0$, if $D(q) \neq d$. Therefore the transition function of a simplified QPA is $\varphi(q_1, \sigma, \tau, q, \omega) = \delta(q_1, \sigma, \tau, q, D(q), \omega)$.

Taking into account Definition 5, following well-formedness conditions correspond to simplified QPA:

Well-formedness conditions 2.

1. Local probability condition. $\forall(q_1, \sigma_1, \tau_1) \in Q \times \Gamma \times \Delta$

$$\sum_{(q,\omega)\in Q\times\Delta^*} |\varphi(q_1, \sigma_1, \tau_1, q, \omega)|^2 = 1\,. \tag{9}$$

2. Orthogonality of column vectors condition. For all triples $(q_1, \sigma_1, \tau_1) \neq (q_2, \sigma_1, \tau_2)$ in $Q \times \Gamma \times \Delta$

$$\sum_{(q,\omega)\in Q\times\Delta^*} \varphi^*(q_1, \sigma_1, \tau_1, q, \omega)\varphi(q_2, \sigma_1, \tau_2, q, \omega) = 0\,. \tag{10}$$

3. Row vectors norm condition. $\forall(q_1, \sigma_1, \tau_1, \tau_2) \in Q \times \Gamma \times \Delta^2$

$$\sum_{(q,\tau,\omega)\in Q\times\Delta\times\{\varepsilon,\tau_2,\tau_1\tau_2\}} |\varphi(q, \sigma_1, \tau, q_1, \omega)|^2 = 1\,. \tag{11}$$

4. Separability condition. $\forall(q_1, \sigma_1, \tau_1), (q_2, \sigma_1, \tau_2) \in Q \times \Gamma \times \Delta$, $\forall\tau_3 \in \Delta$

$$\sum_{(q,\tau)\in Q\times\Delta} \varphi^*(q_1, \sigma_1, \tau_1, q, \tau)\varphi(q_2, \sigma_1, \tau_2, q, \tau_3\tau) + \\ + \sum_{q\in Q} \varphi^*(q_1, \sigma_1, \tau_1, q, \varepsilon)\varphi(q_2, \sigma_1, \tau_2, q, \tau_3) = 0\,, \tag{12}$$

$$\sum_{q\in Q} \varphi^*(q_1, \sigma_1, \tau_1, q, \varepsilon)\varphi(q_2, \sigma_1, \tau_2, q, \tau_2\tau_3) = 0\,. \tag{13}$$

Theorem 2. *The evolution of a simplified QPA is unitary iff Well-formedness conditions 2 are satisfied.*

Proof. By Theorem 1 and Definition 5. □

3 Language Recognition

Language recognition for QPA is defined as follows. For a QPA $A = (Q, \Sigma, T, q_0, Q_a, Q_r, \delta)$ we define $C_a = \{|\nu_i q \nu_k, \omega_l\rangle \in C \mid q \in Q_a\}$, $C_r = \{|\nu_i q \nu_k, \omega_l\rangle \in C \mid q \in Q_r\}$, $C_n = C \setminus (C_a \cup C_r)$. E_a, E_r, E_n are subspaces of H_A spanned by C_a, C_r, C_n respectively. We use the observable $\mathcal{O}$ that corresponds to the orthogonal decomposition $H_A = E_a \oplus E_r \oplus E_n$. The outcome of each observation is either "accept" or "reject" or "non-halting". The language recognition is now defined as follows: For an $x \in \Sigma^*$ we consider as an input $\#x\$$, and assume that the computation starts with A being in the configuration $|q_0\#x\$, Z_0\rangle$. Each computation step consists of two parts. At first the linear operator U_A is applied to the current global state and then the resulting superposition

is observed using the observable $\mathcal{O}$ as defined above. If the global state before the observation is $\sum_{c \in C} \alpha_c |c\rangle$, then the probability that the resulting superposition is projected into the subspace E_i, $i \in \{a, \tau, n\}$, is $\sum_{c \in C_i} |\alpha_c|^2$. The computation continues until the result of an observation is "accept" or "reject".

Definition 6. We shall say that an automaton is a *deterministic reversible pushdown automaton* (RPA), if it is a simplified QPA with $\varphi(q_1, \sigma, \tau, q, \omega) \in \{0, 1\}$ and there exists a function $f : Q \times \Gamma \times \Delta \longrightarrow Q \times \Delta^*$, such that $f(q_1, \sigma, \tau) = (q, \omega)$ if and only if $\varphi(q_1, \sigma, \tau, q, \omega) = 1$.

We can regard f as a transition function of a RPA. Note that the local probability condition (9) is satisfied automatically for RPA.

Theorem 3. *Every regular language is recognizable by some QPA.*

Proof. It is sufficient to prove that any deterministic finite automaton (DFA) can be simulated by RPA. Let us consider a DFA with n states $A_{DFA} = (Q_{DFA}, \Sigma, q_0, Q_F, \delta)$, where $\delta : Q_{DFA} \times \Sigma \longrightarrow Q_{DFA}$. To simulate A_{DFA} we shall construct a RPA $A_{RPA} = (Q, \Sigma, T, q_0, Q_a, Q_r, \varphi)$ with the number of states $2n$. The set of states is $Q = Q_{DFA} \cup Q'_{DFA}$, where $Q_{DFA} \cap Q'_{DFA} = \emptyset$ and Q'_{DFA} are the newly introduced states, which are linked to Q_{DFA} by a one-to-one relation $\{(q_i, q'_i) \in Q_{DFA} \times Q'_{DFA}\}$. Thus Q_F has one-to-one relation to $Q'_F \subset Q'_{DFA}$. The stack alphabet is $T = Ind(Q_{DFA})$, where $\forall i \; Ind(q_i) = i$; the set of accepting states is $Q_a = Q'_F$ and the set of rejecting states is $Q_r = Q'_{DFA} \setminus Q'_F$. As for the function D, $D(Q_{DFA}) = \{\rightarrow\}$ and $D(Q'_{DFA}) = \{\downarrow\}$. We shall define sets R and $\overline{R}$ as follows:

$$R = \{(q'_j, \sigma, i) \in Q'_{DFA} \times \Sigma \times T \mid \delta(q_i, \sigma) = q_j\},$$
$$\overline{R} = \{(q'_j, \sigma, i) \in Q'_{DFA} \times \Sigma \times T \mid \delta(q_i, \sigma) \neq q_j\}.$$

The construction of the transition function f is performed by the following rules:

1. $\forall (q_i, \sigma, \tau) \in Q_{DFA} \times \Sigma \times \Delta \;\; f(q_i, \sigma, \tau) = (\delta(q_i, \sigma), \tau i)$,
2. $\forall (q'_j, \sigma, i) \in R \;\; f(q'_j, \sigma, i) = (q'_i, \varepsilon)$,
3. $\forall (q'_j, \sigma, i) \in \overline{R} \;\; f(q'_j, \sigma, i) = (q_j, i)$,
4. $\forall (q'_j, \sigma) \in Q'_{DFA} \times \Sigma \;\; f(q'_j, \sigma, Z) = (q_j, Z)$,
5. $\forall (q, \tau) \in Q \times \Delta \;\; f(q, \#, \tau) = (q, \tau)$,
6. $\forall (q_i, \tau) \in Q_{DFA} \times \Delta \;\; f(q_i, \$, \tau) = (q'_i, \tau)$,
7. $\forall (q'_i, \tau) \in Q'_{DFA} \times \Delta \;\; f(q'_i, \$, \tau) = (q_i, \tau)$,.

Thus we have defined f for all the possible arguments. Our automaton simulates the DFA. Note that the automaton may reach a state in Q'_{DFA} only by reading the end-marking symbol \$ on the input tape. As soon as A_{RPA} reaches the end-marking symbol \$, it goes to an accepting state, if its current state is in Q_F, and goes to a rejecting state otherwise. The construction is performed in a way so that Well-formedness conditions 2 are satisfied. As we know, RPA automatically satisfies the local probability condition (9).

Let us prove, that the automaton satisfies the orthogonality condition (10). For RPA, the condition (10) is equivalent to the requirement that for all triples $(q_1, \sigma_1, \tau_1) \neq (q_2, \sigma_1, \tau_2)$ $f(q_1, \sigma_1, \tau_1) \neq f(q_2, \sigma_1, \tau_2)$.

Let us consider the case when $(q_1, \sigma_1, \tau_1), (q_2, \sigma_1, \tau_2) \in R$. We shall denote q_1, q_2 as q'_i, q'_j respectively. Let us assume from the contrary that $f(q'_i, \sigma_1, \tau_1) = f(q'_j, \sigma_1, \tau_2)$. By rule 2, $(q'_{\tau_1}, \varepsilon) = (q'_{\tau_2}, \varepsilon)$. Hence $\tau_1 = \tau_2$. By the definition of R, $\delta(q_{\tau_1}, \sigma_1) = q_i$ and $\delta(q_{\tau_2}, \sigma_1) = q_j$. Since $\tau_1 = \tau_2$, $q_i = q_j$. Therefore $q'_i = q'_j$, i. e., $q_1 = q_2$. We have come to a contradiction with the fact that $(q_1, \sigma_1, \tau_1) \neq (q_2, \sigma_1, \tau_2)$. In other cases, proof is straightforward.

The compliance with row vectors norm condition (11) and separability conditions (12) and (13) is proved in the same way. □

Example 3. Let us consider a language $L_1 = (0,1)^*1$, for which we know that it is not recognizable by QFA [11]. This language is recognized by a deterministic finite automaton with two states q_0, q_1 and the following transitions: $\delta(q_0, 0) = q_0$, $\delta(q_0, 1) = q_1$, $\delta(q_1, 0) = q_0$, $\delta(q_1, 1) = q_1$. By Theorem 3 it is possible to transform this automaton to the following RPA: $Q = \{q_0, q_1, q'_0, q'_1\}$, $Q_a = \{q'_1\}$, $Q_r = \{q'_0\}$, $\Sigma = \{0, 1\}$, $T = \{0, 1\}$, $D(q_0) = \rightarrow$, $D(q_1) = \rightarrow$, $D(q'_0) = \downarrow$, $D(q'_1) = \downarrow$. By the construction rules, $\forall q \in Q$ $\forall \sigma \in \Sigma$, $\forall \tau \in \Delta$

$$
\begin{array}{lll}
f(q_0, 0, \tau) = (q_0, \tau 0) & f(q_1, 0, \tau) = (q_0, \tau 1) & f(q_0, 1, \tau) = (q_1, \tau 0) \\
f(q_1, 1, \tau) = (q_1, \tau 1) & f(q'_0, 0, 0) = (q'_0, \varepsilon) & f(q'_1, 1, 0) = (q'_0, \varepsilon) \\
f(q'_0, 0, 1) = (q'_1, \varepsilon) & f(q'_1, 1, 1) = (q'_1, \varepsilon) & f(q'_0, 1, 0) = (q_0, 0) \\
f(q'_1, 0, 0) = (q_1, 0) & f(q'_0, 1, 1) = (q_0, 1) & f(q'_1, 0, 1) = (q_1, 1) \\
f(q'_0, \sigma, Z) = (q_0, Z) & f(q'_1, \sigma, Z) = (q_1, Z) & f(q, \#, \tau) = (q, \tau) \\
f(q_0, \tau) = (q'_0, \tau) & f(q_1, \tau) = (q'_1, \tau) & f(q'_0, \tau) = (q_0, \tau) \\
f(q'_1, \tau) = (q_1, \tau). & &
\end{array}
$$

Let us consider a language which is not regular, namely,

$$L_2 = \{\omega \in (a, b)^* \mid |\omega|_a = |\omega|_b\},$$

where $|\omega|_i$ denotes the number of occurrences of the symbol i in the word ω.

Lemma 3. *Language L_2 is recognizable by a RPA.*

Proof. Our RPA has four states q_0, q_1, q_2, q_3, where q_2 is an accepting state, whereas q_3 — rejecting one. Stack alphabet T consists of two symbols 1, 2. Stack filled with 1's means that the processed part of the word ω has more occurrences of a's than b's, whereas 2's means that there are more b's than a's. Furthermore, length of the stack word is equal to the difference of number of a's and b's. Empty stack denotes that the number of a's and b's is equal. Values of the transition

function follow: $\forall q \in Q\ \forall \tau \in \Delta$

$$
\begin{array}{lll}
f(q,\#,\tau) = (q,\tau) & f(q_0,a,Z) = (q_0,Z1) & D(q_0) = \rightarrow \\
f(q_0,b,Z) = (q_0,Z2) & f(q_0,\$,Z) = (q_2,Z1) & D(q_1) = \downarrow \\
f(q_0,a,1) = (q_0,11) & f(q_0,b,1) = (q_1,\varepsilon) & D(q_2) = \downarrow \\
f(q_0,\$,1) = (q_3,1) & f(q_0,a,2) = (q_1,\varepsilon) & D(q_3) = \downarrow \\
f(q_0,b,2) = (q_0,22) & f(q_0,\$,2) = (q_3,2) & f(q_1,a,Z) = (q_0,Z) \\
f(q_1,b,Z) = (q_0,Z) & f(q_1,\$,\tau) = (q_1,\tau) & f(q_1,a,1) = (q_3,12) \\
f(q_1,b,1) = (q_0,1) & f(q_1,a,2) = (q_0,2) & f(q_1,b,2) = (q_3,21) \\
f(q_2,a,Z) = (q_3,Z2) & f(q_2,b,Z) = (q_3,Z1) & f(q_2,\$,Z) = (q_0,Z) \\
f(q_2,a,1) = (q_2,\epsilon) & f(q_2,b,1) = (q_0,12) & f(q_2,\$,1) = (q_0,1) \\
f(q_2,a,2) = (q_0,21) & f(q_2,b,2) = (q_2,\epsilon) & f(q_2,\$,2) = (q_0,2) \\
f(q_3,a,Z) = (q_3,Z) & f(q_3,b,Z) = (q_3,Z) & f(q_3,\$,Z) = (q_3,Z) \\
f(q_3,a,1) = (q_3,1) & f(q_3,b,1) = (q_3,11) & f(q_3,\$,1) = (q_2,1) \\
f(q_3,a,2) = (q_3,22) & f(q_3,b,2) = (q_3,2) & f(q_3,\$,2) = (q_2,2)\,.
\end{array}
$$

□

It is doubtful whether the language L_2 can be recognized with probability 1 by QPA with stack alphabet T containing only one symbol, i.e, by quantum finite one counter automata [12].

Lemma 4. *Pumping lemma for context-free languages. Every context free language L has a positive integer constant m with the following property. If ω is in L and $|\omega| \geq m$, then ω can be written as $uvxyz$, where uv^kxy^kz is in L for each $k \geq 0$. Moreover, $|vxy| \leq m$ and $|vy| > 0$.*

The pumping lemma is from [10], p. 123. Let us consider a language L_3 which is not recognizable by any deterministic pushdown automaton:

Theorem 4. *Language $L_3 = \{\omega \in (a,b,c)^* |\ |\omega|_a = |\omega|_b = |\omega|_c\}$ is recognizable by a QPA with probability $\frac{2}{3}$.*

Proof (Sketch). The automaton takes three equiprobable actions, during the first action it compares $|\omega|_a$ to $|\omega|_b$, whereas during the second action $|\omega|_b$ to $|\omega|_c$ is compared. Input word is rejected if the third action is chosen. Acceptance probability totals $\frac{2}{3}$. By Lemma 4, the language L_3 is not a context-free language (take $\omega = a^m b^m c^m$). Hence it is not recognizable by deterministic pushdown automata. □

Theorem 5. *Language $L_4 = \{\omega \in (a,b,c)^* : |\omega|_a = |\omega|_b \ xor\ |\omega|_a = |\omega|_c\}$ is recognizable by a QPA with probability $\frac{4}{7}$.*

Proof (sketch). The automaton starts the following actions with the following amplitudes:

a) with an amplitude $\sqrt{\frac{2}{7}}$ compares $|\omega|_a$ to $|\omega|_b$,

b) with an amplitude $-\sqrt{\frac{2}{7}}$ compares $|\omega|_a$ to $|\omega|_c$,

c) with an amplitude $\sqrt{\frac{3}{7}}$ accepts the input.

If exactly one comparison gives positive answer, input is accepted with probability $\frac{4}{7}$. If both comparisons gives positive answer, amplitudes, which are chosen to be opposite, annihilate and the input is accepted with probability $\frac{3}{7}$. □

Language L_4 cannot be recognized by deterministic pushdown automata. (By Lemma 4, take $\omega = a^{m+m!}b^m c^{m+m!}$) It even seems that this language is not recognizable by probabilistic pushdown automata either. In this case this result would be similar to that of [1], where the properties of quantum finite multitape automata are considered.

References

1. A. Ambainis, R. Bonner, R. Freivalds, M. Golovkins, M. Karpinski: Quantum Finite Multitape Automata. *Lecture Notes in Computer Science*, 1999, Vol. 1725, pp. 340–348.
2. A. Ambainis, R. Bonner, R. Freivalds, A. Kikusts: Probabilities to Accept Languages by Quantum Finite Automata. *Lecture Notes in Computer Science*, 1999, Vol. 1627, pp. 174–183.
3. A. Ambainis, R. Freivalds: 1-Way Quantum Finite Automata: Strengths, Weaknesses and Generalizations. *Proc. 39th FOCS*, 1998, pp. 332–341.
4. E. Bernstein, U. Vazirani: Quantum Complexity Theory. *SIAM Journal on Computing*, 26:1411–1473, 1997.
5. A. Brodsky, N. Pippenger: Characterizations of 1-Way Quantum Finite Automata. http://xxx.lanl.gov/abs/quant-ph/9903014.
6. D. Deutsch: Quantum Theory, the Church-Turing principle and the Universal Quantum Computer. *Proc. Royal Society London, A400*, 1985. pp. 96–117.
7. C. Dürr, M. Santha: A Decision Procedure for Unitary Linear Quantum Cellular Automata. *Proc. 37th FOCS*, 1996, pp. 38–45.
8. R. Feynman: Simulating Physics with Computers. *International Journal of Theoretical Physics*, 1982, vol. 21, No 6/7, pp. 467–488.
9. J. Gruska: Quantum Challenges. *Lecture Notes in Computer Science*, 1999, Vol. 1725, pp. 1–28.
10. E. Gurari: An Introduction to the Theory of Computation. *Computer Science Press*, 1989.
11. A. Kondacs, J. Watrous: On The Power of Quantum Finite State Automata. In *Proc. 38th FOCS*, 1997, pp. 66–75.
12. M. Kravtsev: Quantum Finite One-Counter Automata. *Lecture Notes in Computer Science*, 1999, Vol. 1725, pp. 431–440.
13. C. Moore, J. P. Crutchfield: Quantum Automata and Quantum Grammars. http://xxx.lanl.gov/abs/quant-ph/9707031.
14. P. W. Shor: Algorithms for Quantum Computation: Discrete Logarithms and Factoring. *Proc. 35th FOCS*, 1994, pp. 124–134.

15. M. Valdats: The Class of Languages Recognizable by 1-Way Quantum Finite Automata is not Closed Under Union. *Proc. Quantum Computation and Learning. International Workshop*, 2000, pp. 52 – 64. E-print: http://xxx.lanl.gov/abs/quant-ph/0001005.

Use of Dependency Microcontexts in Information Retrieval*

Martin Holub

Department of Software Engineering, Faculty of Mathematics and Physics,
Charles University, Prague, Czech republic
holub@ksi.ms.mff.cuni.cz

Abstract. This paper focuses especially on two problems that are crucial for retrieval performance in information retrieval (IR) systems: the lack of information caused by document pre-processing and the difficulty caused by homonymous and synonymous words in natural language. Author argues that traditional IR methods, i.e. methods based on dealing with individual terms without considering their relations, can be overcome using natural language processing (NLP). In order to detect the relations among terms in sentences and make use of lemmatisation and morphological and syntactic tagging of Czech texts, author proposes a method for construction of dependency word microcontexts fully automatically extracted from texts, and several ways how to exploit the microcontexts for the sake of increasing retrieval performance.

1 Introduction

Empirical methods in natural language processing (NLP) employ learning techniques to automatically extract linguistic knowledge from natural language corpora; for an overview of this field see (Brill and Mooney 1997). This paper wants to show their usefulness in the field of *information retrieval* (IR). A textual IR system stores a collection of documents and special data structures for effective searching. A textual document is a sequence of terms. When analysing the content of a document, terms are the basic processed units — usually they are words of natural language. When retrieving, the IR system returns documents presumed to be of interest to the user in response to a query. The user's query is a formal statement of user's information need. The documents that are interesting for the user (relative to the put query) are relevant; the others are non-relevant. The effectiveness of IR systems is usually measured in terms of *precision*, the percentage of retrieved documents that are relevant, and *recall*, the percentage of relevant documents that are retrieved.

The starting point of our consideration on IR was a critique of word-based retrieval techniques. Traditional IR systems treat the query as a pattern of words to be matched by documents. Unfortunately, the effectiveness of these word-matching systems is mostly poor because the system retrieves only the documents which contain words that occur also in the query. However, in fact, the user

* This study has been supported by MŠMT (the FRVŠ grant no 1909).

V. Hlaváč, K. G. Jeffery, and J. Wiedermann (Eds.): SOFSEM 2000, LNCS 1963, pp. 347–355, 2000.

Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Singapore
Tokyo

Jan Pavelka Gerard Tel
Miroslav Bartošek (Eds.)

SOFSEM'99: Theory and Practice of Informatics

26th Conference on Current Trends
in Theory and Practice of Informatics
Milovy, Czech Republic, November 27 - December 4, 1999
Proceedings

Springer

Table of Contents

Invited Talks

From Data to Knowledge

Applications

Contributed Papers

Quantum Finite Multitape Automata

Andris Ambainis[1*], Richard Bonner[2], Rūsiņš Freivalds[3**], Marats Golovkins[3***], and Marek Karpinski[4†]

[1] Computer Science Division, University of California, Berkeley, CA 94720-2320
ambainis@cs.berkeley.edu
[2] Department of Mathematics and Physics, Mälardalens University
richard.bonner@mdh.se
[3] Institute of Mathematics and Computer Science, University of Latvia
Raiņa bulv. 29, Riga, Latvia
rusins@cclu.lv
marats@cclu.lv
[4] Department of Computer Science, University of Bonn
53117, Bonn, Germany
marek@cs.bonn.edu

Abstract. Quantum finite automata were introduced by C. Moore, J. P. Crutchfield [4], and by A. Kondacs and J. Watrous [3]. This notion is not a generalization of the deterministic finite automata. Moreover, in [3] it was proved that not all regular languages can be recognized by quantum finite automata. A. Ambainis and R. Freivalds [1] proved that for some languages quantum finite automata may be exponentially more concise rather than both deterministic and probabilistic finite automata. In this paper we introduce the notion of quantum finite multitape automata and prove that there is a language recognized by a quantum finite automaton but not by deterministic or probabilistic finite automata. This is the first result on a problem which can be solved by a quantum computer but not by a deterministic or probabilistic computer. Additionally we discover unexpected probabilistic automata recognizing complicated languages.

1 Introduction

The basic model, i.e., quantum finite automata (QFA), were introduced twice. First this was done by C. Moore and J. P. Crutchfield [4]. Later in a different and non-equivalent way these automata were introduced by A. Kondacs and J. Watrous [3].

* Supported by Berkeley Fellowship for Graduate Studies.
** Research supported by Grant No. 96.0282 from the Latvian Council of Science
*** Research supported by Grant No. 96.0282 from the Latvian Council of Science
† Research partially supported by the International Computer Science Institute, Berkeley, California, by the DFG grant KA 673/4-1, and by the ESPRIT BR Grants 7079 and ECUS030

J. Pavelka, G. Tel, M. Bartošek (Eds.): SOFSEM'99, LNCS 1725, pp. 340–348, 1999.

The first definition just mimics the definition of 1-way finite probabilistic only substituting *stochastic* matrices by *unitary* ones. To define quantum finite multitape automata we generalize a more elaborated definition [3].

We are using these notations in the following definition:
z^* is the complex conjugate of a complex number z.
$M =_{def} \{1, 2, \ldots, m\}$.
The k-th component of an arbitrary vector s will be defined as s^k.
We shall understand by I an arbitrary element from the set $\mathcal{P}(M) \setminus \{\emptyset\}$.

$R_I =_{def} A_1 \times A_2 \times \cdots \times A_m$, where $A_i = \begin{cases} \{\downarrow, \rightarrow\}, & \text{if } i \notin I \\ \{\text{"nothing"}\}, & \text{if } i \in I. \end{cases}$

$T_I =_{def} B_1 \times B_2 \times \cdots \times B_m$, where $B_i = \begin{cases} \{\downarrow, \rightarrow\}, & \text{if } i \in I \\ \{\text{"nothing"}\}, & \text{if } i \notin I. \end{cases}$

The function $R_i \times T_i \xrightarrow{d_I} \{\downarrow, \rightarrow\}^m$ is defined as follows:

$d_I(r,t) =_{def} (d_I^1(r,t), d_I^2(r,t), \ldots, d_I^m(r,t))$, where $d_I^i(r,t) = \begin{cases} r^i, & \text{if } i \notin I \\ t^i, & \text{if } i \in I. \end{cases}$

Definition 1. *A quantum finite multitape automaton (QFMA)* $A = (Q; \Sigma; \delta; q_0; Q_a; Q_r)$ *is specified by the finite input alphabet* Σ*, the finite set of states* Q*, the initial state* $q_0 \in Q$*, the sets* $Q_a \subset Q$*,* $Q_r \subset Q$ *of accepting and rejecting states, respectively, with* $Q_a \cap Q_r = \emptyset$*, and the transition function*

$$\delta\colon Q \times \Gamma^m \times Q \times \{\downarrow, \rightarrow\}^m \longrightarrow \mathbb{C}_{[0,1]},$$

where m *is the number of input tapes,* $\Gamma = \Sigma \cup \{\#, \$\}$ *is the tape alphabet of* A *and* $\#, \$$ *are end-markers not in* Σ*, which satisfies the following conditions (of well-formedness):*

1. *Local probability condition.*

$$\forall (q_1, \sigma) \in Q \times \Gamma^m\colon \sum_{(q,d) \in Q \times \{\downarrow, \rightarrow\}^m} |\delta(q_1, \sigma, q, d)|^2 = 1.$$

2. *Orthogonality of column vectors condition.*

$$\forall q_1, q_2 \in Q, q_1 \neq q_2, \forall \sigma \in \Gamma^m\colon \sum_{(q,d) \in Q \times \{\downarrow, \rightarrow\}^m} \delta^*(q_1, \sigma, q, d)\delta(q_2, \sigma, q, d) = 0.$$

3. *Separability condition.*

$$\forall I \in \mathcal{P}(M) \setminus \{\emptyset\}\ \forall q_1, q_2 \in Q$$
$$\forall \sigma_1, \sigma_2 \in \Gamma^m, \text{where } \forall i \notin I\ \sigma_1^i = \sigma_2^i$$
$$\forall t_1, t_2 \in T_I, \text{ where } \forall j \in I\ t_1^j \neq t_2^j$$
$$\sum_{(q,r) \in Q \times R_I} \delta^*(q_1, \sigma_1, q, d_I(r, t_1))\delta(q_2, \sigma_2, q, d_I(r, t_2)) = 0.$$

States from $Q_a \cup Q_r$ are called halting states and states from $Q_{non} = Q \setminus (Q_a \cup Q_r)$ are called non-halting states.

To process an input word vector $x \in (\Sigma^*)^m$ by A it is assumed that the input is written on every tape k with the end-markers in the form $w_x^k = \#x^k\$$ and that every such a tape, of length $|x^k| + 2$, is circular, i. e., the symbol to the right of \$ is #.

For the fixed input word vector x we can define $n \in \mathbb{N}^m$ to be an integer vector which determines the length of input word on every tape. So for every n we can define C_n to be the set of all possible configurations of A where $|x^i| = n^i$. $|C_n| = |Q| \prod_{i=1}^{m} (n^i + 2)$. Every such a configuration is uniquely determined by a pair $|q, s\rangle$, where $q \in Q$ and $0 \leq s^i \leq |x^i| + 1$ specifies the position of head on the i-th tape.

Every computation of A on an input x, $|x^i| = n^i$, is specified by a unitary evolution in the Hilbert space $H_{A,n} = l_2(C_n)$. Each configuration $c \in C_n$ corresponds to the basis vector in $H_{A,n}$. Therefore a global state of A in the space $H_{A,n}$ has a form $\sum_{c \in C_n} \alpha_c |c\rangle$, where $\sum_{c \in C_n} |\alpha_c|^2 = 1$. If the input word vector is x and the automaton A is in its global state $|\psi\rangle = \sum_{c \in C_n} \alpha_c |c\rangle$, then its further step is equivalent to the application of a linear operator U_x^δ over Hilbert space $l_2(C_n)$.

Definition 2. *The linear operator U_x^δ is defined as follows:*

$$U_x^\delta |\psi\rangle = \sum_{c \in C_n} \alpha_c U_x^\delta |c\rangle.$$

If a configuration $c = |q', s\rangle$, then

$$U_x^\delta |c\rangle = \sum_{(q,d) \in Q \times \{\downarrow, \rightarrow\}^m} \delta(q', \sigma(s), q, d) \mid q, \tau(s, d)\rangle,$$

where $\sigma(s) = (\sigma^1(s), \ldots, \sigma^m(s))$, $\sigma^i(s)$ specifies the s^i-th symbol on the i-th tape, and $\tau(s, d) = (\tau^1(s, d), \ldots, \tau^m(s, d))$,

$$\tau^i(s, d) = \begin{cases} (s^i + 1) \bmod (n^i + 2), & \text{if } d^i = \text{`} \rightarrow \text{'} \\ s^i, & \text{if } d^i = \text{`} \downarrow \text{'}. \end{cases}$$

Lemma 1. *The well-formedness conditions are satisfied iff for any input x the mapping U_x^δ is unitary.*

Language recognition for QFMA is defined as follows. For each input x with the corresponding vector n, $n^i = |x^i|$, and a QFMA $A = (Q; \Sigma; \delta; q_0; Q_a; Q_r)$ we define $C_n^a = \{(q, s) \mid (q, s) \in C_n, q \in Q_a\}$, $C_n^r = \{(q, s) \mid (q, s) \in C_n, q \in Q_r\}$, $C_n^{non} = C_n \setminus (C_n^a \cup C_n^r)$. E_a, E_r, E_{non} are the subspaces of $l_2(C_n)$ spanned by C_n^a, C_n^r, C_n^{non} respectively. We use the observable $\mathcal{O}$ that corresponds to the orthogonal decomposition $l_2(C_n) = E_a \oplus E_r \oplus E_{non}$. The outcome of each observation is either "accept" or "reject" or "non-halting".

The language recognition is now defined as follows: For an $x \in (\Sigma^*)^m$ we consider as the input ω_x, $\omega_x^k = \#x^k\$$, and assume that the computation starts with A being in the configuration $|q_0, \{0\}^m\rangle$. Each computation step consists of two parts. At first the linear operator $U_{\omega_x}^{\delta}$ is applied to the current global state and then the resulting superposition, i.e., global state, is observed using the observable $\mathcal{O}$ as defined above. If the global state before the observation is $\sum_{c \in C_n} \alpha_c |c\rangle$, then the probability that the subspace E_i, $i \in \{a, r, non\}$, will be chosen is $\sum_{c \in C_n^i} |\alpha_c|^2$. The computation continues until the result of an observation is "accept" or "reject".

Definition 3. *A QFMA $A = (Q; \Sigma; \delta; q_0; Q_a; Q_r)$ is simple if for each $\sigma \in \Gamma^m$ there is a linear unitary operator V_σ over the inner-product space $l_2(Q)$ and a function $D: Q \longrightarrow \{\downarrow, \rightarrow\}^m$, such that*

$$\forall q_1 \in Q \; \forall \sigma \in \Gamma^m \quad \delta(q_1, \sigma, q, d) = \begin{cases} \langle q|V_\sigma|q_1\rangle, & \text{if } D(q) = d \\ 0, & \text{otherwise.} \end{cases}$$

Lemma 2. *If the automaton A is simple, then conditions of well-formedness are satisfied iff for every σ V_σ is unitary.*

We shall deal only with simple multitape automata further in the paper.

2 Quantum vs. Probabilistic Automata

Definition 4. *We shall say that an automaton is deterministic reversible finite multitape automaton (RFMA), if it is a simple QFMA with $\delta(q_1, \sigma, q, d) \in \{0, 1\}$.*

Definition 5. *We say that a language L is $[m, n]$-deterministically recognizable if there are n deterministic automata A_1, A_2, A_n such that:*

a) if the input is in the language L, then all n automata $A_1, \ldots, A_n$ accept the input;
b) if the input is not in the language L, then at most m of the automata $A_1, \ldots, A_n$ accept the input.

Definition 6. *We say that a language L is $[m, n]$-reversibly recognizable if there are n deterministic reversible automata A_1, A_2, A_n such that:*

a) if the input is in the language L, then all n automata $A_1, \ldots, A_n$ accept the input;
b) if the input is not in the language L, then at most m of the automata $A_1, \ldots, A_n$ accept the input.

Lemma 3. *If a language L is $[1,n]$-deterministically recognizable by 2-tape finite automata, then L is recognizable by a probabilistic 2-tape finite automaton with probability $\frac{n}{n+1}$.*

Proof. The probabilistic automaton starts by choosing a random integer $1 \leq r \leq (n+1)$. After that, if $r \leq n$, then the automaton goes on simulating the deterministic automaton A_r, and, if $r = n+1$, then the automaton rejects the input. The inputs in L are accepted with probability $\frac{n}{n+1}$, and the inputs not in the language are rejected with a probability no less than $\frac{n}{n+1}$. □

Lemma 4. *If a language L is $[1,n]$-reversibly recognizable by 2-tape finite automata, then L is recognizable by a quantum 2-tape finite automaton with probability $\frac{n}{n+1}$.*

Proof. In essence the algorithm is the same as in Lemma 3. The automaton starts by taking $n+1$ different actions with amplitudes $\frac{1}{\sqrt{n+1}}$. (It is possible to construct a unitary matrix to make such a choice feasible.) After that the automaton simultaneously goes on simulating all the deterministic reversible automata A_r, $1 \leq r \leq (n+1)$, where the automaton A_{n+1} rejects an input. The simulation of each deterministic reversible automaton uses its own accepting and rejecting states. (Hence the probabilities are totaled, not the amplitudes.) □

First, we discuss the following 2-tape language

$$L_1 = \{(x_1 \nabla x_2, y) \mid x_1 = x_2 = y\},$$

where the words x_1, x_2, y are unary.

Lemma 5. *(Proved by R. Freivalds [2].) For arbitrary natural n, the language L_1 is $[1,n]$-deterministically recognizable.*

Lemma 6. *For arbitrary natural n, the language L_1 is $[1,n]$-reversibly recognizable.*

Proof. By Lemma 5, the language L_1 is $[1,n]$-deterministically recognizable. However it is easy enough to make the construction of the automata $A_1, \ldots, A_n$ in the following manner:

a) every automaton is reversible;
b) if a word pair is in the language L_1, then every automaton consumes the same number of steps to accept the word pair.

The last requirement will be essential further in the paper. If at least the first requirement is met, then the language is $[1,n]$-reversibly recognizable. □

Theorem 1. *The language L_1 can be recognized with arbitrary probability $1-\epsilon$ by a probabilistic 2-tape finite automaton but this language cannot be recognized by a deterministic 2-tape finite automaton.*

Theorem 2. *The language L_1 can be recognized with arbitrary probability $1-\epsilon$ by a quantum 2-tape finite automaton.*

Proof. By Lemmas 4 and 6. □

In an attempt to construct a 2-tape language recognizable by a quantum 2-tape finite automaton but not by probabilistic 2-tape finite automata we consider a similar language

$$L_2 = \{(x_1 \nabla x_2 \nabla x_3, y) \mid \text{there are exactly 2 values of } x_1, x_2, x_3 \text{ such that they equal } y\},$$

where the words x_1, x_2, x_3, y are unary.

Theorem 3. *A quantum automaton exists which recognises the language L_2 with a probability $\frac{9}{16} - \epsilon$ for arbitrary positive ϵ.*

Proof. This automaton takes the following actions with the following amplitudes:

a) $\frac{\sqrt{3}}{4} \cdot 1$ – compares $x_1 = x_2 = y$,
b) $\frac{\sqrt{3}}{4} \cdot (\cos\frac{2\pi}{3} + i \sin\frac{2\pi}{3})$ – compares $x_2 = x_3 = y$,
c) $\frac{\sqrt{3}}{4} \cdot (\cos\frac{4\pi}{3} + i \sin\frac{4\pi}{3})$ – compares $x_1 = x_3 = y$,
d) $\frac{\sqrt{7}}{4}$ – says "accept".

By Theorem 2 comparison in actions a), b), c) can be accomplished. By construction in Lemma 4 the comparison in each action a), b), c) is implemented by starting $n+1$ different branches. Therefore in any action i), $i \in \{a, b, c\}$, if a comparison is successful, the automaton will come respectively into non-halting states $q_{a,1}, \ldots, q_{a,n}$, $q_{b,1}, \ldots, q_{b,n}$, $q_{c,1}, \ldots, q_{c,n}$, reaching the symbol pair (\$, \$) on the tapes. The transition (\$, \$) for every $k = 1, \ldots, n$ is as follows:

	$q_{a,k}$	$q_{b,k}$	$q_{c,k}$
$q_{a1,k}$	$\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}$
$q_{r,k}$	$\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}(\cos\frac{4\pi}{3} + i \sin\frac{4\pi}{3})$	$\frac{1}{\sqrt{3}}(\cos\frac{2\pi}{3} + i \sin\frac{2\pi}{3})$
$q_{a2,k}$	$\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}(\cos\frac{2\pi}{3} + i \sin\frac{2\pi}{3})$	$\frac{1}{\sqrt{3}}(\cos\frac{4\pi}{3} + i \sin\frac{4\pi}{3})$

Here $q_{a1,k}, q_{a2,k}$ are accepting states and $q_{r,k}$ are rejecting states. If y equals all 3 words x_1, x_2, x_3, then it is possible to ensure that it takes the same time to reach the end-marking symbol pair in every action on every branch. Therefore the input is accepted with probability $\frac{7}{16} + \epsilon$ (since the amplitudes of the actions a), b), c) total to 0). If y equals 2 out of 3 words x_1, x_2, x_3, then the input is accepted with probability $\frac{9}{16} - \epsilon$. If y equals at most one of the words x_1, x_2, x_3, then the input is accepted with probability $\frac{7}{16} + \epsilon$ (only if the action d) is taken). □

Unfortunately, the following theorem holds.

Theorem 4. *A probabilistic automaton exists which recognizes the language L_2 with a probability $\frac{21}{40}$.*

Proof. The probabilistic automaton with probability $\frac{1}{2}$ takes an action A or B:

A) Choose a random j and compare $x_j = y$. If yes, accept with probability $\frac{19}{20}$. If no, accept with probability $\frac{1}{20}$.
B) Choose a random pair j, k and compare $x_j = x_k = y$. If yes, reject. If no, accept with probability $\frac{12}{20}$.

If y equals all 3 words x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability $\frac{19}{20}$. If y equals all 3 words x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability 0. This gives the acceptance probability in the case if y equals all 3 words x_1, x_2, x_3, to be $\frac{19}{40}$ and the probability of the correct result "no" to be $\frac{21}{40}$.

If y equals 2 words out of x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability $\frac{13}{20}$. If y equals 2 words out of x_1, x_2, x_3 and the action B is taken, then the input is accepted with relative probability $\frac{8}{20}$. This gives the acceptance probability in the case if y equals 2 words out of x_1, x_2, x_3, to be $\frac{21}{40}$.

If y equals only 1 word out of x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability $\frac{7}{20}$. If y equals only 1 word out of x_1, x_2, x_3 and the action B is taken, then the input is accepted with relative probability $\frac{12}{20}$. This gives the acceptance probability in the case if y equals only 1 word out of x_1, x_2, x_3, to be $\frac{19}{40}$ and the probability of the correct result "no" to be $\frac{21}{40}$.

If y equals no word of x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability $\frac{1}{20}$. If y equals no word of x_1, x_2, x_3 and the action B is taken, then the input is accepted with relative probability $\frac{12}{20}$. This gives the acceptance probability in the case if y equals no word of x_1, x_2, x_3, to be $\frac{13}{40}$ and the probability of the correct result "no" to be $\frac{27}{40}$. □

Now we consider a modification of the language L_2 which might be more difficult for a probabilistic recognition:

$$L_3 = \{(x_1 \nabla x_2 \nabla x_3, y_1 \nabla y_2) \mid \text{there is exactly one value } k \text{ such that there are exactly two values } j \text{ such that } x_j = y_k\}$$

Theorem 5. *A quantum finite 2-tape automaton exists which recognizes the language L_3 with a probability $\frac{36}{67} - \epsilon$ for arbitrary positive ϵ.*

However this language also can be recognized by a probabilistic 2-tape finite automaton.

Theorem 6. *A probabilistic finite 2-tape automaton exists which recognizes the language L_3 with a probability $\frac{13}{25} - \epsilon$ for arbitrary positive ϵ.*

Proof. The probabilistic automaton with probability $\frac{6}{25}$ takes action A or B or C or with probability $\frac{7}{25}$ takes action D:

A) Choose a random k and two values of j. Then compare $x_j = y_k$. If yes, accept. If no, reject.

B) Chose a random k and compare $x_1 = x_2 = x_3 = y_k$. If yes, reject. If no, accept.
C) Choose two values j and m. Then compare $x_j = x_m = y_1 = y_2$. If yes, reject. If no, accept.
D) Says "reject".

Notice that the actions A, B, C are probabilistic, and they can be performed only with probability $1 - \epsilon$ (actions A and B are described in the proof of Theorem 1 and action C is similar).
The acceptance probabilities equal:

	A	B	C	total
no y_k equals 2 or 3 x_j	0	1	1	$\frac{12}{25}$
one y_k equals 2 x_j	$\frac{1}{6}$	1	1	$\frac{13}{25}$
one y_k equals 3 x_j	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{12}{25}$
two y_k equal 2 x_j	$\frac{1}{3}$	1	$\frac{2}{3}$	$\frac{12}{25}$
all y_k equal all x_j	1	0	0	$\frac{0}{25}$

□

Finally we consider a modification of the languages above which recognition indeed is impossible by probabilistic automata:

$$L_4 = \{(x_1 \nabla x_2, y) \mid \text{there is exactly one value } j \text{ such that } x_j = y\}$$

where the words x_1, x_2, y are binary.

Theorem 7. *A quantum finite 2-tape automaton exists which recognizes the language L_4 with a probability $\frac{4}{7}$.*

Proof. The automaton has two accepting q_{a1}, q_{a2} and three rejecting states q_{r1}, q_{r2}, q_{r3} and starts the following actions by reading the pair $(\#, \#)$ with the following amplitudes:

a) with an amplitude $\sqrt{\frac{2}{7}}$ compares x_1 to y,

b) with an amplitude $-\sqrt{\frac{2}{7}}$ compares x_2 to y,

c) with an amplitude $\sqrt{\frac{3}{7}}$ immediately goes to the state q_{a1}.

Actions a) and b) use different non-halting states to process the word pair. All these actions the automaton processes simultaneously. In actions a) and b), if no (not equal), it goes accordingly to the states q_{r1} or q_{r2}, if yes, then reaches correspondent non-halting states q_α or q_β, while the symbol pair on the tapes is $(\$, \$)$. The transition for $(\$, \$)$ and states $q_\alpha, q_\beta, q_{a2}, q_{r3}$ is as follows:

	q_α	q_β
q_{a2}	$\frac{1}{\sqrt{2}}$	$\frac{1}{\sqrt{2}}$
q_{r3}	$\frac{1}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}$

If all the words are equal, it is possible to ensure that it takes the same time to reach the end-markers on both tapes, therefore the automaton reaches the superposition $\sqrt{\frac{2}{7}}|q_\alpha, s, t\rangle - \sqrt{\frac{2}{7}}|q_\beta, s, t\rangle$, where s and t specify the place of \$ on each tape, and the input is accepted with probability $\frac{3}{7}$. (Since the amplitudes of the actions a) and b) equal to 0.) If one of the words x_i equals y, then the input is accepted with probability $\frac{4}{7}$. If none of the words x_i equals y, then the input is accepted with probability $\frac{3}{7}$. □

References

1. Andris Ambainis and Rūsiņš Freivalds. 1-way quantum finite automata: strengths, weaknesses and generalizations. *Proc. 39th FOCS*, 1998, pp. 332–341. http://xxx.lanl.gov/abs/quant-ph/9802062.
2. Rūsiņš Freivalds. Fast probabilistic algorithms. *Lecture Notes in Computer Science*, 1979, Vol. 74, pp. 57–69.
3. Attila Kondacs and John Watrous. On the power of quantum finite state automata. In *Proc. 38th FOCS*, 1997, pp. 66–75.
4. Christopher Moore, James P. Crutchfield Quantum automata and quantum grammars. http://xxx.lanl.gov/abs/quant-ph/9707031.

Decomposable Bulk Synchronous Parallel Computers*

Martin Beran

Faculty of Mathematics and Physics, Charles University
Malostranské nám. 25, 118 00 Praha 1, the Czech Republic
beran@ss1000.ms.mff.cuni.cz

Abstract. The Bulk Synchronous Parallel (BSP) computer is a generally accepted realistic model of parallel computers introduced by Valiant in 1990. We present an extension to the BSP model – a decomposable BSP (dBSP for short). Performance of several elementary algorithms, namely broadcasting, prefix computation, and matrix multiplication, is analyzed on BSP and dBSP models. For a suitable setting of parameters, these algorithms run asymptotically faster on dBSP than on BSP. We also show how space-bounded sequential algorithms can be transformed into pipelined ones with bounded period on dBSP. Such a transformation is proved impossible for the BSP model. Finally, we present an algorithm for the simulation of dBSP on BSP.

1 Introduction

The bulk synchronous parallel (BSP) model of parallel computation was defined by Valiant in 1990 in [14]. A BSP computer consists of p processors with local memories, which can communicate by sending messages via a router. A computation consists of S supersteps and is periodically synchronized after each superstep. During the i-th superstep, each processor makes w_i local operations and sends or receives h_i messages[1] (such a communication request is called the h-relation). The sent messages are available at the destination processors in the beginning of the next superstep. The superstep is finished by a barrier synchronization. The computation takes time $T^{\mathrm{BSP}} = \sum_{i=1}^{S}(w_i + h_i g(p) + l(p))$. The nondecreasing functions $g(p)$ (network bandwidth per processor) and $l(p)$ (communication latency and barrier synchronization time) are machine dependent parameters defining performance of the router. An extensive research on BSP algorithms and implementation of the model on real computers has been done in recent years [2,3,4,6,9,10,12].

The standard BSP charges communication between any pair of processors equally. Thus, it cannot exploit communication locality present in many algorithms. By communication locality we mean that a processor communicates not

* This research was supported by the GA ČR grant No. 201/98/0717.

[1] w_i and h_i are both maximum over all the processors.

J. Pavelka, G. Tel, M. Bartošek (Eds.): SOFSEM'99, LNCS 1725, pp. 349–359, 1999.

Richard Bonner and Rūsiņš Freivalds (Eds.)

Quantum Computation and Learning

International Workshop
25-26 May 2002
Riga, Latvia

Proceedings

Contents

Probabilistic Reversibility and Its Relation to Quantum Automata

Marats Golovkins * and Maksim Kravtsev **

Institute of Mathematics and Computer Science, University of Latvia
Raiņa bulv. 29, Riga, Latvia
marats@latnet.lv, maksims@batsoft.lv

Abstract. To study relationship between quantum finite automata and probabilistic finite automata, we introduce a notion of probabilistic reversible automata (PRA, or doubly stochastic automata). We find that there is a strong relationship between different possible models of PRA and corresponding models of quantum finite automata. We also propose a classification of reversible finite 1-way automata.

1 Introduction

Here we introduce common notions used throughout the paper as well as summarize its contents.

We analyze two models of probabilistic reversible automata in this paper, namely, 1-way PRA and 1.5-way PRA.

In this section, we define notions applicable to both models in a quasi-formal way, including a general definition for probabilistic reversibility. These notions are defined formally in further sections.

If not specified otherwise, we denote by Σ an input alphabet of an automaton.

Every input word is enclosed into *end-marker* symbols # and \$. Therefore we introduce a *working alphabet* as $\Gamma = \Sigma \cup \{\#, \$\}$.

By Q we normally understand the set of states of an automaton.

By $\overline{L}$ we understand complement of a language L.

Given an input word ω, by $|\omega|$ we understand the number of symbols in ω and with $[\omega]_i$ we denote i-th symbol of ω, counting from the beginning (excluding end-markers).

Definition 1.1. *A* configuration *of a finite automaton is* $c = \langle \nu_i q_j \nu_k \rangle$, *where the automaton is in a state* $q_j \in Q$, $\nu_i \nu_k \in \#\Sigma^*\$$ *is a finite word on the input tape and input tape head is above the first symbol of the word* ν_k.

* Research partially supported by the Latvian Council of Science, grant No. 01.0354 and grant for Ph.D. students; University of Latvia, K. Morbergs grant; European Commission, contract IST-1999-11234

** Research partially supported by the Latvian Council of Science, grant No. 01.0354 and European Commission, contract IST-1999-11234

By C we denote the set of all configurations of an automaton. This set is countably infinite.

After its every step, a probabilistic automaton is in some probability distribution $p_0c_0 + p_1c_1 + \ldots + p_nc_n$, where $p_0 + p_1 + \ldots + p_n = 1$. Such probability distribution is called a *superposition of configurations*. Given an input word ω, the number of configurations in every accessible superposition does not exceed $|Q|$ in case of 1-way automata, and $|\omega||Q|$ in case of 1.5-way automata.

A linear closure of C forms a linear space, where every configuration can be viewed as a basis vector. This basis is called a *canonical basis*. Every probabilistic automaton defines a linear operator over this linear space.

Let us consider A. Nayak's model of quantum automata with mixed states. (Evolution is characterized by a unitary matrix and subsequent measurements are performed after each step, POVM measurements not being allowed, [N 99].) If a result of every measurement is a single configuration, not a superposition, and measurements are performed after each step, we actually get a probabilistic automaton. However, the following property applies to such probabilistic automata - their evolution matrices are *doubly* stochastic. This encourages us to give the following definition for probabilistic reversible automata:

Definition 1.2. *A probabilistic automaton is called* reversible *if its linear operator can be described by a doubly stochastic matrix, using canonical basis.*

To make accessible configurations of type $\langle q_i\#\omega\$\rangle$, we assume that every word is written on a circular tape, and after the right end-marker \$ the next symbol is the left end-marker #. Such precondition is the same as used for quantum finite automata. (See, for example, [KW 97].)

At least two definitions exist, how to interpret word acceptance, and hence, language recognition, for reversible automata.

Definition 1.3. Classical acceptance. *We say that an automaton accepts (rejects) a word classically, if its set of states consists of two disjoint subsets: accepting states and rejecting states, and the following conditions hold:*

- *the automaton accepts the word, if it is in accepting state after having read the last symbol of the word;*
- *the automaton rejects the word, if it is in rejecting state after having read the last symbol of the word.*

We refer to the classical acceptance automata as C-automata further in the paper.

Definition 1.4. "Decide and halt" acceptance. *We say that an automaton accepts (rejects) a word in a decide-and-halt manner, if its set of states consists of three disjoint subsets: accepting states, rejecting states and non-halting states, and the following conditions hold:*

- *the computation is continued only if the automaton enters a non-halting state.*
- *if the automaton enters an accepting state, the word is accepted;*

– *if the automaton enters a rejecting state, the word is rejected.*

We refer to the decide-and-halt automata as DH-automata further in the paper.

Having defined word acceptance, we define language recognition in an equivalent way as in [R 63]. We consider only bounded error language recognition in this paper.

By $P_{x,A}$ we denote the probability that a word x is accepted by an automaton A.

Definition 1.5. *We say that a language L is recognized with bounded error by an automaton A with interval (p_1, p_2) if $p_1 < p_2$ and $p_1 = \sup\{P_{x,A} \mid x \notin L\}$, $p_2 = \inf\{P_{x,A} \mid x \in L\}$.*

Definition 1.6. *We say that a language is recognized with a probability p if the language is recognized with interval $(1-p, p)$.*

Definition 1.7. *We say that a language is recognized with probability $1-\varepsilon$, if for every $\varepsilon > 0$ there exists an automaton which recognizes the language with interval $(\varepsilon_1, 1-\varepsilon_2)$, where $\varepsilon_1, \varepsilon_2 \leq \varepsilon$.*

Definition 1.8. *By $q \xrightarrow{S} q'$, $S \subset \Sigma^*$, we denote that there is a positive probability to get to a state q' by reading a single word $\xi \in S$, starting in a state q.*

We refer to several existing models of quantum finite automata:

1. Measure-once quantum finite automata [MC 97] (QFA-MC);
2. Measure-many quantum finite automata [KW 97] (QFA-KW);
3. Enhanced quantum finite automata [N 99] (QFA-N).

Following the notions above, QFA-MC can be characterized as C-automata whereas QFA-KW and QFA-N as DH-automata.

In Section 2, we discuss properties of PRA C-automata (PRA-C). We prove that PRA-C recognize the class of languages $a_1^* a_2^* \dots a_n^*$ with probability $1-\varepsilon$. This class can be recognized by QFA-KW, with worse acceptance probabilities, however [ABFK 99]. This also implies that QFA-N recognize this class of languages with probability $1-\varepsilon$.

Further, we show general class of regular languages, not recognizable by PRA-C. In particular, such languages as (a,b)*a and a(a,b)* are in this class. This class has strong similarities with the class of languages, not recognizable by QFA-KW [AKV 00].

We also show that the class of languages recognized by PRA-C is closed under boolean operations, inverse homomorphisms and word quotient, but is not closed under homomorphisms.

In Section 3 we prove, that PRA DH-automata do not recognize the language (a,b)*a.

In Section 4 we discuss some properties of 1.5-way PRA. We also present an alternative notion of probabilistic reversibility, not connected with quantum automata.

In Section 5 we propose a classification of reversible automata (deterministic, probabilistic and quantum).

2 1-way Probabilistic Reversible C-Automata

Definition 2.1. *1-way probabilistic reversible C-automaton (PRA-C)* $A = (Q, \Sigma, q_0, Q_F, \delta)$ *is specified by a finite set of states* Q*, a finite input alphabet* Σ*, an initial state* $q_0 \in Q$*, a set of accepting states* $Q_F \subseteq Q$*, and a transition function*

$$\delta : Q \times \Gamma \times Q \longrightarrow \mathbb{R}_{[0,1]},$$

where $\Gamma = \Sigma \cup \{\#, \$\}$ *is the input tape alphabet of* A *and* $\#$*,* $\$$ *are end-markers not in* Σ*. Furthermore, transition function satisfies the following requirements:*

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \ \sum_{q \in Q} \delta(q_1, \sigma_1, q) = 1 \tag{1}$$

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \ \sum_{q \in Q} \delta(q, \sigma_1, q_1) = 1 \tag{2}$$

For every input symbol $\sigma \in \Gamma$, the transition function may be determined by a $|Q| \times |Q|$ matrix V_σ, where $(V_\sigma)_{i,j} = \delta(q_j, \sigma, q_i)$.

Lemma 2.2. *All matrices* V_σ *are doubly stochastic iff conditions (1) and (2) of Definition 2.1 hold.*

Proof. Trivial. □

We define word acceptance as specified in Definition 1.3. The set of rejecting states is $Q \setminus Q_F$. We define language recognition as in Definition 1.5.

A linear operator U_A corresponds to the automaton A. Formal definition of this operator follows:

Definition 2.3. *Given a configuration* $c = \langle \nu_i q_j \sigma \nu_k \rangle$*,*

$$U_A c \stackrel{\text{def}}{=} \sum_{q \in Q} \delta(q_j, \sigma, q) \langle \nu_i \sigma q \nu_k \rangle.$$

Given a superposition of configurations $\psi = \sum_{c \in C} p_c c$*,*

$$U_A \psi \stackrel{\text{def}}{=} \sum_{c \in C} p_c U_A c.$$

Using canonical basis, U_A is described by an infinite matrix M_A.

To comply with Definition 1.2, we have to state the following:

Lemma 2.4. *Matrix* M_A *is doubly stochastic iff conditions (1) and (2) of Definition 2.1 hold.*

Proof. Condition (1) takes place if and only if the sum of elements in every column in M_A equal to 1. Condition (2) takes place if and only if the sum of elements in every row in M_A equal to 1. □

This completes our formal definition of PRA-C.

Use of end-markers does not affect computational power of PRA-C. For every PRA-C with end-markers which recognizes some language it is possible to construct a PRA-C without end-markers which recognizes the same language. (Number of states needed may increase, however.) See Appendix for further details.

Lemma 2.5. *If a language is recognized by a PRA-C A with interval (p_1, p_2), exists a PRA-C which recognizes the language with probability p, where*

$$p = \begin{cases} \frac{p_2}{p_1+p_2}, & \text{if } p_1 + p_2 \geq 1 \\ \frac{1-p_1}{2-p_1-p_2}, & \text{if } p_1 + p_2 < 1. \end{cases}$$

Proof. Let us assume, that the automaton A has $n-1$ states. We consider the case $p_1 + p_2 > 1$.

Informally, having read end-marker symbol $\#$, we simulate the automaton A with probability $\frac{1}{p_1+p_2}$ and reject input with probability $\frac{p_1+p_2-1}{p_1+p_2}$.

Formally, to recognize the language with probability $\frac{p_2}{p_1+p_2}$, we modify the automaton A. We add a new state $q_r \notin Q_F$, and change the transition function in the following way:

- $\forall \sigma, \sigma \neq \#, \delta(q_r, \sigma, q_r) \stackrel{\text{def}}{=} 1$;
- $\delta(q_0, \#, q_r) \stackrel{\text{def}}{=} \frac{p_1+p_2-1}{p_1+p_2}$;
- $\forall q, q \neq q_r, \delta(q_0, \#, q) \stackrel{\text{def}}{=} \frac{1}{p_1+p_2}\delta_{old}(q_0, \#, q)$.

Now the automaton has n states. Since end-marker symbol $\#$ is read only once at the beginning of an input word, we can disregard the rest of transition function values, associated with $\#$: $\forall q_i, q_j$, where $q_i \neq q_0$, $\delta(q_i, \#, q_j) \stackrel{\text{def}}{=} \frac{1-\delta(q_0,\#,q_j)}{n-1}$.

The transition function satisfies the requirements of Definition 2.1 and the constructed automaton recognizes the language with probability $\frac{p_2}{p_1+p_2}$.

The case $p_1 + p_2 < 1$ is very similar. Informally, having read end-marker symbol $\#$, we simulate the automaton A with probability $\frac{1}{2-p_1-p_2}$ and accept input with probability $\frac{1-p_1-p_2}{2-p_1-p_2}$. □

Theorem 2.6. *If a language is recognized by a PRA-C, it is recognized by PRA-C with probability $1-\varepsilon$.*

Proof. We assume that a language L is recognized by a PRA-C automaton $A = (Q, \Sigma, q_0, Q_F, \delta)$ with interval (p_1, p_2). Let $\delta = \frac{1}{2}(p_1 + p_2)$.

Let us consider a system of m copies of the automaton A, denoted as A_m. We say that our system has accepted (rejected) a word if more (less or equal) than $m\delta$ automata in the system have accepted (rejected) the word. We define language recognition as in Definition 1.5.

Let us consider a word $\omega \in L$. The automaton A accepts ω with probability $p_\omega \geq p_2$. As a result of reading ω, μ_m^ω automata of the system accept the word, and the rest reject it. The system has accepted the word, if $\frac{\mu_m^\omega}{m} > \delta$. Let us take

η_0, such that $0 < \eta_0 < p_2 - \delta \leq p_\omega - \delta$. Estimating the probability that $\frac{\mu_m^\omega}{m} > \delta$, we have

$$P\left\{\frac{\mu_m^\omega}{m} > \delta\right\} \geq P\left\{p_\omega - \eta_0 < \frac{\mu_m^\omega}{m} < p_\omega + \eta_0\right\} = P\left\{\left|\frac{\mu_m^\omega}{m} - p_\omega\right| < \eta_0\right\} \quad (3)$$

In case of m Bernoulli trials, Chebyshev's inequality may be used to prove the following ([GS 97], p. 312):

$$P\left\{\left|\frac{\mu_m^\omega}{m} - p_\omega\right| \geq \eta_0\right\} \leq \frac{p_\omega(1-p_\omega)}{m\eta_0^2} \leq \frac{1}{4m\eta_0^2} \quad (4)$$

The last inequality induces that

$$P\left\{\left|\frac{\mu_m^\omega}{m} - p_\omega\right| < \eta_0\right\} \geq 1 - \frac{1}{4m\eta_0^2} \quad (5)$$

Finally, putting (3) and (5) together,

$$P\left\{\frac{\mu_m^\omega}{m} > \delta\right\} \geq 1 - \frac{1}{4m\eta_0^2} \quad (6)$$

Inequality (6) is true for every $\omega \in L$.

On the other hand, let us consider a word $\xi \notin L$. The automaton A accepts ξ with probability $p_\xi \leq p_1$. If we take the same η_0, $0 < \eta_0 < \delta - p_1 \leq \delta - p_\xi$ and for every ξ we have

$$P\left\{\frac{\mu_m^\xi}{m} > \delta\right\} \leq P\left\{\left|\frac{\mu_m^\xi}{m} - p_\xi\right| \geq \eta_0\right\} \leq \frac{1}{4m\eta_0^2} \quad (7)$$

Due to (6) and (7), for every $\varepsilon > 0$, if we take $n > \frac{1}{4\varepsilon\eta_0^2}$, we get a system A_n which recognizes the language L with interval $(\varepsilon_1, 1-\varepsilon_2)$, where $\varepsilon_1, \varepsilon_2 < \varepsilon$.

Let us show that A_n can be simulated by a PRA-C. The automaton $A' = (Q', \Sigma, q_0', Q_F', \delta')$ is constructed as follows:

$Q' \stackrel{\text{def}}{=} \{\langle q_{s_1} q_{s_2} \dots q_{s_n}\rangle \mid 0 \leq s_i \leq |Q| - 1\}$; $q_0' \stackrel{\text{def}}{=} \langle q_0 q_0 \dots q_0\rangle$.

A sequence $\langle q_{s_1} q_{s_2} \dots q_{s_n}\rangle$ is an accepting state of A' if more than $n\delta$ elements in the sequence are accepting states of A. We have defined the set Q_F'.

Given $\sigma \in \Gamma$, $\delta'(\langle q_{a_1} q_{a_2} \dots q_{a_n}\rangle, \sigma, \langle q_{b_1} q_{b_2} \dots q_{b_n}\rangle) \stackrel{\text{def}}{=} \prod_{i=1}^{n} \delta(q_{a_i}, \sigma, q_{b_i})$.

In essence, Q' is n-th Cartesian power of Q and the linear space formed by A' is n-th tensor power of the linear space formed by A. If we take a symbol $\sigma \in \Gamma$, transition is determined by $|Q|^n \times |Q|^n$ matrix V_σ', which is n-th matrix direct power of V_σ, i.e, $V_\sigma' = \bigotimes_{i=1}^{n} V_\sigma$.

A' simulates the system A_n. Since matrix direct product of two doubly stochastic matrices is a doubly stochastic matrix, $\forall\sigma$ V_σ' are doubly stochastic matrices. Therefore our automaton A' is a PRA-C.

We have proved that $\forall\varepsilon > 0$ the language L is recognized by some PRA-C with interval $(\varepsilon_1, 1-\varepsilon_2)$, where $\varepsilon_1, \varepsilon_2 < \varepsilon$. Therefore the language L is recognized with probability $1 - \varepsilon$. □

Lemma 2.7. *If a language L_1 is recognizable with probability greater than $\frac{2}{3}$ and a language L_2 is recognizable with probability greater than $\frac{2}{3}$ then languages $L_1 \cap L_2$ and $L_1 \cup L_2$ are recognizable with probability greater than $\frac{1}{2}$.*

Proof. Let us consider automata $A = (Q_A, \Sigma, q_{0,A}, Q_{F,A}, \delta_A)$ and $B = (Q_B, \Sigma, q_{0,B}, Q_{F,B}, \delta_B)$ which recognize the languages L_1, L_2 with probabilities $p_1, p_2 > \frac{2}{3}$, respectively. Let us assume that A, B have m and n states, respectively. Without loss of generality we can assume that $p_1 \leq p_2$.

Informally, having read end-marker symbol $\#$, with probability $\frac{1}{2}$ we simulate the automaton A_1 and with the same probability we simulate the automaton A_2.

Formally, we construct an automaton $C = (Q, \Sigma, q_0, Q_F, \delta)$ with the following properties.
$Q \stackrel{def}{=} Q_A \cup Q_B$; $q_0 \stackrel{def}{=} q_{0,A}$; $Q_F \stackrel{def}{=} Q_{F,A} \cup Q_{F,B}$; $\delta \stackrel{def}{=} \delta_A \cup \delta_B$, with an exception that:

- $\delta(q_0, \#, q_{i,A}) = \frac{1}{2}\delta_A(q_0, \#, q_{i,A})$;
- $\delta(q_0, \#, q_{i,B}) = \frac{1}{2}\delta_B(q_0, \#, q_{i,B})$;
- $\forall q_i,\ q_i \neq q_0,\ \delta(q_i, \#, q) = \frac{1-\delta(q_0, \#, q)}{m+n-1}$.

Since δ satisfies Definition 2.1, our construction of PRA-C is complete.

The automaton C recognizes the language $L_1 \cap L_2$ with interval $(p, \frac{p_1+p_2}{2})$, where $p \leq 1 - \frac{1}{2}p_1$. (Since $p_1, p_2 > \frac{2}{3}$, $1 - \frac{1}{2}p_1 < \frac{p_1+p_2}{2}$)

The automaton C recognizes the language $L_1 \cup L_2$ with interval $(\frac{2-p_1-p_2}{2}, p)$, where $p \geq \frac{1}{2}p_1$. (Again, $\frac{2-p_1-p_2}{2} < \frac{1}{2}p_1$)

Therefore by Lemma 2.5, the languages $L_1 \cap L_2$ and $L_1 \cup L_2$ are recognizable with probabilities greater than $\frac{1}{2}$. □

Theorem 2.8. *The class of languages recognized by PRA-C is closed under intersection, union and complement.*

Proof. Let us consider languages L_1, L_2 recognized by some PRA-C automata. By Theorem 2.6, these languages is recognizable with probability $1 - \varepsilon$, and therefore by Lemmas 2.5 and 2.7, union and intersection of these languages are also recognizable. If a language L is recognizable by a PRA-C A, we can construct an automaton which recognizes a language $\overline{L}$ just by making accepting states of A to be rejecting, and vice versa. □

It is natural to ask what are the languages recognized by PRA-C with probability exactly 1.

Theorem 2.9. *If a language is recognized by a PRA-C with probability 1, the language is recognized by a permutation automaton.*

Proof. Let us consider a language L and a PRA-C A, which recognizes L with probability 1.

If a word is in L, the automaton A has to accept the word with probability 1. Conversely, if a word is not in L, the word must be accepted with probability 0. Therefore,

$$\forall q \in Q\ \forall \omega \in \Sigma^* \text{ either } q\omega \subseteq Q_F, \text{ or } q\omega \subseteq \overline{Q}_F. \tag{8}$$

Consider a relation between the states of A defined as

$R = \{(q_i, q_j) \mid \forall \omega\ q_i\omega \subseteq Q_F \Leftrightarrow q_j\omega \subseteq Q_F\}$. R is symmetric, reflexive and transitive, therefore Q can be partitioned into equivalence classes $Q/R = \{[q_0], [q_{i_1}], \ldots, [q_{i_k}]\}$. Suppose A is in a state q. Due to (8), $\forall \omega\ \exists n\ q\omega \subseteq [q_{i_n}]$. In fact, having read a symbol in the alphabet, A goes from one equivalence class to another with probability 1.

Hence it is possible to construct the following deterministic automaton D, which simulates A. The states are $s_0, \ldots, s_k$ and $s_n\sigma = s_m$ iff $[q_{i_n}]\sigma \subseteq [q_{i_m}]$ and s_n is an accepting state iff $[q_{i_n}] \subseteq Q_F$. Since all transition matrices of A are doubly stochastic, all transition matrices of D are permutation matrices. □

Theorem 2.10. *The class of languages recognized by PRA-C is closed under inverse homomorphisms.*

Proof. Let us consider finite alphabets Σ, T, a homomorphism $h : \Sigma \longrightarrow T^*$, a language $L \subseteq T^*$ and a PRA-C $A = (Q, T, q_0, Q_F, \delta)$, which recognizes L with interval (p_1, p_2). We prove that exists an automaton $B = (Q, \Sigma, q_0, Q_F, \delta')$ which recognizes the language $h^{-1}(L)$.

Transition function δ of A sets transition matrices V_τ, where $\tau \in T$. To determine δ', we define transition matrices V_σ, $\sigma \in \Sigma$. Let us define a transition matrix V_{σ_k}:

$$V_{\sigma_k} = V_{[h(\sigma_k)]_m} V_{[h(\sigma_k)]_{m-1}} \cdots V_{[h(\sigma_k)]_1},$$

where $m = |h(\sigma_k)|$. Multiplication of two doubly stochastic matrices is a doubly stochastic matrix, therefore B is a PRA-C. Automaton B recognizes $h^{-1}(L)$ with the same interval (p_1, p_2). □

Corollary 2.11. *The class of languages recognized by PRA-C is closed under word quotient.*

Proof. This follows from closure under inverse homomorphisms and presence of end-markers #, \$. □

Even if PRA-C without end-markers are considered, closure under word quotient remains true. See Appendix for details.

Lemma 2.12. *If A is a doubly stochastic matrix and X - a vector, then* $\max(X) \geq \max(AX)$ *and* $\min(X) \leq \min(AX)$.

Proof. Let us consider $X = \begin{pmatrix} x_1 \\ x_2 \\ \ldots \\ x_n \end{pmatrix}$ and $A = \begin{pmatrix} a_{11}\ a_{12} \ldots a_{1n} \\ a_{21}\ a_{22} \ldots a_{2n} \\ \ldots\ \ldots\ \ldots\ \ldots \\ a_{n1}\ a_{n2} \ldots a_{nn} \end{pmatrix}$, where A is doubly stochastic. Let us suppose that $x_j = \max(X)$. For any i, $1 \leq i \leq n$,

$$x_j = a_{i1}x_j + a_{i2}x_j + \ldots + a_{in}x_j \geq a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n.$$

Therefore x_j is greater or equal than any component of AX. The second inequality is proved in the same way. □

Theorem 2.13. *For every natural positive n, a language $L_n = a_1^* a_2^* \dots a_n^*$ is recognizable by some PRA-C with alphabet $\{a_1, a_2, \dots, a_n\}$.*

Proof. We construct a PRA-C with $n+1$ states, q_0 being the initial state, corresponding to probability distribution vector $\begin{pmatrix} 1 \\ 0 \\ \dots \\ 0 \end{pmatrix}$. The transition function is determined by $(n+1) \times (n+1)$ matrices

$$V_{a_1} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \frac{1}{n} & \dots & \frac{1}{n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \frac{1}{n} & \dots & \frac{1}{n} \end{pmatrix}, V_{a_2} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & \dots & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & \dots & 0 \\ 0 & 0 & \frac{1}{n-1} & \dots & \frac{1}{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \frac{1}{n-1} & \dots & \frac{1}{n-1} \end{pmatrix}, \dots, V_{a_n} = \begin{pmatrix} \frac{1}{n} & \dots & \frac{1}{n} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \frac{1}{n} & \dots & \frac{1}{n} & 0 \\ 0 & \dots & 0 & 1 \end{pmatrix}.$$

The accepting states are $q_0 \dots q_{n-1}$, the only rejecting state is q_n. We prove, that the automaton recognizes the language L_n.

Case $\omega \in L_n$. Having read $\omega \in a_1^* \dots a_{k-1}^* a_k^+$, the automaton is in probability distribution $\begin{pmatrix} \frac{1}{k} \\ \dots \\ \frac{1}{k} \\ 0 \\ \dots \\ 0 \end{pmatrix}$. Therefore all $\omega \in L_n$ are accepted with probability 1.

Case $\omega \notin L_n$. Consider k such that $\omega = \omega_1 \sigma \omega_2$, $|\omega_1| = k$, $\omega_1 \in L_n$ and $\omega_1 \sigma \notin L_n$. Since all one-letter words are in L_n, $k > 0$. Let $a_t = [\omega]_k$ and $a_s = \sigma$. So we have $s < t$, $1 \le s \le n-1$, $2 \le t \le n$. Having read $\omega_1 \in a_1^* \dots a_{t-1}^* a_t^+$, the automaton is in distribution $\begin{pmatrix} \frac{1}{t} \\ \dots \\ \frac{1}{t} \\ 0 \\ \dots \\ 0 \end{pmatrix}$. After that, having read a_s, the automaton is in distribution

$$\begin{pmatrix} \frac{1}{s} & \dots & \frac{1}{s} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \frac{1}{s} & \dots & \frac{1}{s} & 0 & \dots & 0 \\ 0 & \dots & 0 & \frac{1}{n-s+1} & \dots & \frac{1}{n-s+1} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & \frac{1}{n-s+1} & \dots & \frac{1}{n-s+1} \end{pmatrix} \begin{pmatrix} \frac{1}{t} \\ \dots \\ \frac{1}{t} \\ 0 \\ \dots \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{t} \\ \dots \\ \frac{1}{t} \\ \frac{t-s}{t(n-s+1)} \\ \dots \\ \frac{t-s}{t(n-s+1)} \end{pmatrix} \begin{matrix} \Big\} s \\ \Big\} n-s+1 \end{matrix}.$$

So the word $\omega_1 a_s$ is accepted with probability $1 - \frac{t-s}{t(n-s+1)}$. By Lemma 2.12, since $\frac{t-s}{t(n-s+1)} < \frac{1}{t}$, reading the symbols succeeding $\omega_1 a_s$ does not increase accepting probability. Therefore, to find maximum accepting probability for words not in L_n, we have to maximize $1 - \frac{t-s}{t(n-s+1)}$, where $s < t$, $1 \le s \le n-1$, $2 \le t \le n$. Solving this problem, we get $t = k+1, s = k$ for $n = 2k$, and we get $t = k+1, s = k$ or $t = k+2, s = k+1$ for $n = 2k+1$. So the maximum accepting

probability is $1-\frac{1}{(k+1)^2}$, if $n=2k$, and it is $1-\frac{1}{(k+1)(k+2)}$, if $n=2k+1$. All in all, the automaton recognizes the language with interval $\left(1-\frac{1}{\left[\left(\frac{n}{2}\right)^2\right]+n+1},\ 1\right)$. (Actually, by Theorem 2.6, L_n can be recognized with probability $1-\varepsilon$). □

Corollary 2.14. *Quantum finite automata with mixed states (model of Nayak, [N 99]) recognize $L_n = a_1^* a_2^* \dots a_n^*$ with probability $1-\varepsilon$.*

Proof. This comes from the fact, that matrices $V_{a_1}, V_{a_2}, \dots, V_{a_n}$ from the proof of Theorem 2.13 (as well as tensor powers of those matrices) all have unitary prototypes (see Definition 5.1). □

Definition 2.15. *We say that a regular language is of type $(*)$ if the following is true for the minimal deterministic automaton recognizing this language: Exist three states q, q_1, q_2, exist words x, y such that*

1. $q_1 \neq q_2$;
2. $qx = q_1$, $qy = q_2$;
3. $\forall t \in (x,y)^* \ \exists t_1 \in (x,y)^* \ q_1 t t_1 = q_1$;
4. $\forall t \in (x,y)^* \ \exists t_2 \in (x,y)^* \ q_2 t t_2 = q_2$.

Definition 2.16. *We say that a regular language is of type $(*')$ if the following is true for the minimal deterministic automaton recognizing this language: Exist three states q, q_1, q_2, exist words x, y such that*

1. $q_1 \neq q_2$;
2. $qx = q_1$, $qy = q_2$;
3. $q_1 x = q_1$, $q_1 y = q_1$;
4. $q_2 x = q_2$, $q_2 y = q_2$.

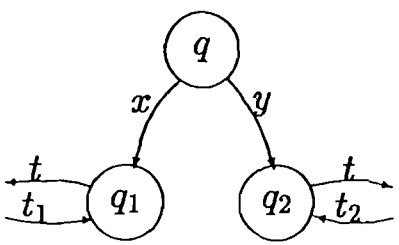

Fig. 1. Type $(*)$ construction

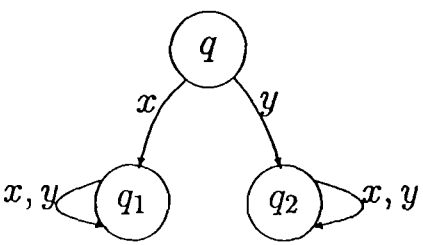

Fig. 2. Type $(*')$ construction

Definition 2.17. *We say that a regular language is of type* $(*'')$ *if the following is true for the minimal deterministic automaton recognizing this language: Exist two states* q_1, q_2, *exist words* x, y *such that*

1. $q_1 \neq q_2$;
2. $q_1 x = q_2$, $q_2 x = q_2$;
3. $q_2 y = q_1$.

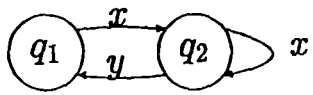

Fig. 3. Type $(*'')$ construction

Type $(*'')$ languages are exactly those languages that violate the partial order condition of [BP 99].

Lemma 2.18. *If* A *is a deterministic finite automaton with a set of states* Q *and alphabet* Σ, *then* $\forall q \in Q\ \forall x \in \Sigma^*\ \exists k > 0\ qx^k = qx^{2k}$.

Proof. We paraphrase a result from the theory of finite semigroups. Consider a state q and a word x. Since number of states is finite, $\exists m \geq 0\ \exists s \geq 1$ $\forall n\ qx^m = qx^m x^{sn}$. Take n_0, such that $sn_0 > m$. Note that $\forall t \geq 0\ qx^{m+t} = qx^{m+t}x^{sn_0}$. We take $t = sn_0 - m$, so $qx^{sn_0} = qx^{sn_0}x^{sn_0}$. Take $k = sn_0$. □

Lemma 2.19. *A regular language is of type* $(*)$ *iff it is of type* $(*')$ *or type* $(*'')$.

Proof. 1) If a language is of type $(*')$, it is of type $(*)$. Obvious.

2) If a language is of type $(*'')$, it is of type $(*)$. Consider a language of type $(*'')$ with states q_1'', q_2'' and words x'', y''. To build construction of type $(*)$, we take $q = q_1 = q_1''$, $q_2 = q_2''$, $x = x''y''$, $y = x''$. That forms transitions $qx = q_1$, $qy = q_2$, $q_1 x = q_1$, $q_1 y = q_2$, $q_2 x = q_1$, $q_2 y = q_2$. We have satisfied all the rules of $(*)$.

3) If a language is of type $(*)$, it is of type $(*')$ or $(*'')$. Consider a language whose minimal deterministic automaton has construction $(*)$. By Lemma 2.18,

$\exists s \exists a\ q_1 x^a = q_s$ and $q_s x^a = q_s$;
$\exists t \exists b\ q_1 y^b = q_t$ and $q_t y^b = q_t$;
$\exists u \exists c\ q_2 x^c = q_u$ and $q_u x^c = q_u$;
$\exists v \exists d\ q_2 y^d = q_v$ and $q_v y^d = q_v$.

If $q_1 \neq q_s$, by the 3rd rule of $(*)$, $\exists z\ q_s z = q_1$. Therefore the language is of type $(*'')$. If $q_2 \neq q_u$, by the 4th rule of $(*)$, $\exists z\ q_u z = q_2$, and the language is of type $(*'')$. Likewise, if $q_1 \neq q_t$ or $q_2 \neq q_v$, the language is of type $(*'')$.
If $q_1 = q_s = q_t$ and $q_2 = q_u = q_v$, we have $qx^a = q_1$, $qy^d = q_2$, $q_1 x^a = q_1 y^b = q_1$, $q_2 x^c = q_2 y^d = q_2$. We get the construction $(*')$ if we take $x' = x^{ac}$, $y' = y^{bd}$. □

We are going to prove that every language of type (*) is not recognizable by any PRA-C. For this purpose, we recall several definitions from the theory of finite Markov chains ([KS 76], etc.)

A Markov chain with n states can be determined by an $n \times n$ stochastic matrix A, i.e., matrix, where the sum of elements of every column in the matrix is 1. If $A_{i,j} = p > 0$, it means that a state q_i is accessible from a state q_j with a positive probability p in one step. Generally speaking, the matrix depends on the numbering of the states; if the states are renumbered, the matrix changes, as its rows and columns also need to be renumbered.

Definition 2.20. *A state q_j is accessible from q_i (denoted $q_i \to q_j$) if there is a positive probability to get from q_i to q_j (possibly in several steps).*

Definition 2.21. *States q_i and q_j communicate (denoted $q_i \leftrightarrow q_j$) if $q_i \to q_j$ and $q_j \to q_i$.*

Definition 2.22. *A state q is called ergodic if $\forall i\ q \to q_i \Rightarrow q_i \to q$. Otherwise the state is called transient.*

Definition 2.23. *A Markov chain without transient states is called irreducible if for all q_i, q_j $q_i \leftrightarrow q_j$. Otherwise the chain without transient states is called reducible.*

Definition 2.24. *The period of an ergodic state $q_i \in Q$ of a Markov chain with a matrix A is defined as $d(q_i) = \gcd\{n > 0 \mid (A^n)_{i,i} > 0\}$.*

Definition 2.25. *An ergodic state q_i is called aperiodic if $d(q_i) = 1$. Otherwise the ergodic state is called periodic.*

Definition 2.26. *A Markov chain without transient states is called aperiodic if all its states are aperiodic. Otherwise the chain without transient states is called periodic.*

Definition 2.27. *A probability distribution X of a Markov chain with a matrix A is called stationary, if $AX = X$.*

Definition 2.28. *A Markov chain is called doubly stochastic, if its transition matrix is a doubly stochastic matrix.*

We recall the following theorem from the theory of finite Markov chains:

Theorem 2.29. *If a Markov chain with a matrix A is irreducible and aperiodic, then*
a) it has a unique stationary distribution Z;
b) $\lim_{n\to\infty} A^n = (Z, \ldots, Z)$;
c) $\forall X \lim_{n\to\infty} A^n X = Z$.

Corollary 2.30. *If a doubly stochastic Markov chain with an $m \times m$ matrix A is irreducible and aperiodic,*

a) $\lim_{n\to\infty} A^n = \begin{pmatrix} \frac{1}{m} & \cdots & \frac{1}{m} \\ \cdots & \cdots & \cdots \\ \frac{1}{m} & \cdots & \frac{1}{m} \end{pmatrix}$;

b) $\forall X \lim_{n\to\infty} A^n X = \begin{pmatrix} \frac{1}{m} \\ \cdots \\ \frac{1}{m} \end{pmatrix}$.

Proof. By Theorem 2.29. □

Lemma 2.31. *If M is a doubly stochastic Markov chain with a matrix A, then* $\forall q\ q \to q$.

Proof. Assume existence of q_0 such that q_0 is not accessible from itself. Let $Q_{q_0} = \{q_i \mid q_0 \to q_i\} = \{q_1, \ldots, q_k\}$. Q_{q_0} is not empty set. Consider those rows and columns of A, which are indexed by states in Q_{q_0}. These rows and columns form a submatrix A'. Each column j of A' must include all non-zero elements of the corresponding column of A as those states are accessible from the state q_j, hence also from q_0 and are in Q_{q_0}. Therefore $\forall j$, $1 \le j \le k$, $\sum_{i=1}^{k} A'_{i,j} = 1$ and $\sum_{1\le i,j\le k} A'_{i,j} = k$. On the other hand, since $q_0 \notin Q_{q_0}$, a row of A' indexed by a state accessible in one step from q_0 does not include all nonzero elements. Since A is doubly stochastic, $\exists i, 1 \le i \le k$, $\sum_{j=1}^{k} A'_{i,j} < 1$ and $\sum_{1\le i,j\le k} A'_{i,j} < k$. This is a contradiction. □

Corollary 2.32. *Suppose A is a doubly stochastic matrix. Then exists $k > 0$, such that* $\forall i\ (A^k)_{i,i} > 0$.

Proof. Consider an $m \times m$ doubly stochastic matrix A. By Lemma 2.31, $\forall i$ $\exists n_i > 0\ (A^{n_i})_{i,i} > 0$. Take $n = \prod_{s=1}^{m} n_s$. For every i, $(A^n)_{i,i} > 0$. □

Lemma 2.33. *If M is a doubly stochastic Markov chain with a matrix A, then* $\forall q_a, q_b\ A_{b,a} > 0 \Rightarrow q_b \to q_a$.

Proof. $A_{b,a} > 0$ means that q_b is accessible from q_a in one step. We have to prove, that $q_b \to q_a$. Assume from the contrary, that q_a is not accessible from q_b. Let $Q_{q_b} = \{q_i \mid q_b \to q_i\} = \{q_1, q_2, \ldots, q_k\}$. By Lemma 2.31, $q_b \in Q_{q_b}$. As in proof of Lemma 2.31, consider a matrix A', which is a submatrix of A and whose rows and columns are indexed by states in Q_{q_b}. Each column j has to include all nonzero elements of the corresponding column of A. Therefore $\forall j$, $1 \le j \le k$, $\sum_{i=1}^{k} A'_{i,j} = 1$ and $\sum_{1\le i,j\le k} A'_{i,j} = k$. On the other hand, $A_{b,a} > 0$ and $q_a \notin Q_{q_b}$, therefore a row of A' indexed by q_b does not include all nonzero elements. Since A is doubly stochastic, $\sum_{j=1}^{k} A'_{b,j} < 1$ and $\sum_{1\le i,j\le k} A'_{i,j} < k$. This is a contradiction. □

Corollary 2.34. *If M is a doubly stochastic Markov chain and $q_a \to q_b$, then $q_a \leftrightarrow q_b$.*

Proof. If $q_a \to q_b$ then exists a sequence $q_{i_1}, q_{i_2}, \ldots, q_{i_k}$, such that $A_{i_1,a} > 0$, $A_{i_2,i_1} > 0, \ldots, A_{i_k,i_{k-1}} > 0$, $A_{b,i_k} > 0$. By Lemma 2.33, we get $q_b \to q_{i_k}$, $q_{i_k} \to q_{i_{k-1}}, \ldots, q_{i_2} \to q_{i_1}$, $q_{i_1} \to q_a$. Therefore $q_b \to q_a$. □

By Corollary 2.34, every doubly stochastic Markov chain does not have transient states, so it is either periodic or aperiodic, either reducible or irreducible.

Lemma 2.35. *If a regular language is of type $(*')$, it is not recognizable by any PRA-C.*

Proof. Assume from the contrary, that A is a PRA-C automaton which recognizes a language $L \subset \Sigma^*$ of type $(*')$.

Since L is of type $(*')$, it is recognized by a deterministic automaton D which has three states q, q_1, q_2 such that $q_1 \neq q_2$, $qx = q_1$, $qy = q_2$, $q_1x = q_1$, $q_1y = q_1$, $q_2x = q_2$, $q_2y = q_2$, where $x, y \in \Sigma^*$. Furthermore, exists $\omega \in \Sigma^*$ such that $q_0\omega = q$, where q_0 is an initial state of D, and exists a word $z \in \Sigma^*$, such that $q_1z = q_{acc}$ if and only if $q_2z = q_{rej}$, where q_{acc} is an accepting state and q_{rej} is a rejecting state of D. Without loss of generality we assume that $q_1z = q_{acc}$ and $q_2z = q_{rej}$.

The transition function of the automaton A is determined by doubly stochastic matrices $V_{\sigma_1}, \ldots, V_{\sigma_n}$. The words from the construction $(*')$ are $x = \sigma_{i_1} \ldots \sigma_{i_k}$ and $y = \sigma_{j_1} \ldots \sigma_{j_s}$. The transitions induced by words x and y are determined by doubly stochastic matrices $X = V_{\sigma_{i_k}} \ldots V_{\sigma_{i_1}}$ and $Y = V_{\sigma_{j_s}} \ldots V_{\sigma_{j_1}}$. Similarly, the transitions induced by words ω and z are determined by doubly stochastic matrices W and Z. By Corollary 2.32, exists $K > 0$, such that

$$\forall i \ (X^K)_{i,i} > 0 \text{ and } (Y^K)_{i,i} > 0. \tag{9}$$

Consider a relation between the states of the automaton defined as $R = \{(q_i, q_j) \mid q_i \xrightarrow{(x^K, y^K)^*} q_j\}$. By (9), this relation is reflexive.

Suppose exists a word $\xi = \xi_1\xi_2 \ldots \xi_k$, $\xi_s \in \{x^K, y^K\}$, such that $q \xrightarrow{\xi} q'$. This means that $q \xrightarrow{\xi_1} q_{i_1}$, $q_{i_1} \xrightarrow{\xi_2} q_{i_2}, \ldots, q_{i_{k-1}} \xrightarrow{\xi_k} q'$. By Corollary 2.34, since both X^K and Y^K are doubly stochastic, $\exists \xi'_k \ldots \xi'_1$, $\xi'_s \in \{(x^K)^*, (y^K)^*\}$, such that $q' \xrightarrow{\xi'_k} q_{i_{k-1}}, \ldots, q_{i_2} \xrightarrow{\xi'_2} q_{i_1}$, $q_{i_1} \xrightarrow{\xi'_1} q$, therefore $q' \xrightarrow{\xi'} q$, where $\xi' \in (x^K, y^K)^*$. So the relation R is symmetric.

Surely R is transitive. Therefore all states of A may be partitioned into equivalence classes $[q_0], [q_{i_1}], \ldots, [q_{i_n}]$. Let us renumber the states of A in such a way, that states from one equivalence class have consecutive numbers. First come the states in $[q_0]$, then in $[q_{i_1}]$, etc.

Consider the word $x^K y^K$. The transition induced by this word is determined by a doubly stochastic matrix $C = Y^K X^K$. We prove the following proposition. States q_a and q_b are in one equivalence class if and only if $q_a \to q_b$ with matrix

C. Suppose $q_a \to q_b$. Then $(q_a, q_b) \in R$, and q_a, q_b are in one equivalence class. Suppose q_a, q_b are in one equivalence class. Then

$$q_a \xrightarrow{\xi_1} q_{i_1}, q_{i_1} \xrightarrow{\xi_2} q_{i_2}, \ldots, q_{i_{k-1}} \xrightarrow{\xi_k} q_b, \text{ where } \xi_s \in \{x^K, y^K\}. \tag{10}$$

By (9), $q_i \xrightarrow{x^K} q_i$ and $q_j \xrightarrow{y^K} q_j$. Therefore, if $q_i \xrightarrow{x^K} q_j$, then $q_i \xrightarrow{x^K y^K} q_j$, and again, if $q_i \xrightarrow{y^K} q_j$, then $q_i \xrightarrow{x^K y^K} q_j$. That transforms (10) to

$$q_a \xrightarrow{(x^K y^K)^t} q_b, \text{ where } t > 0. \tag{11}$$

We have proved the proposition.

By the proved proposition, due to the renumbering of states, matrix C is a block diagonal matrix, where each block corresponds to an equivalence class of the relation R. Let us identify these blocks as $C_0, C_1, \ldots, C_n$. By (9), a Markov chain with matrix C is aperiodic. Therefore each block C_r corresponds to an aperiodic irreducible doubly stochastic Markov chain with states $[q_{i_r}]$. By Corollary 2.30, $\lim_{m\to\infty} C^m = J$, J is a block diagonal matrix, where for each $(p \times p)$ block C_r $(C_r)_{i,j} = \frac{1}{p}$. Relation $q_i \xrightarrow{(y^K)^*} q_j$ is a subrelation of R, therefore Y^K is a block diagonal matrix with the same block ordering and sizes as C and J. (This does not eliminate possibility that some block of Y^K is constituted of smaller blocks, however.) Therefore $JY^K = J$, and $\lim_{m\to\infty} Z(Y^K X^K)^m W = \lim_{m\to\infty} Z(Y^K X^K)^m Y^K W = ZJW$. So

$$\forall \varepsilon > 0 \; \exists m \; \left\| \left(Z(Y^K X^K)^m W - Z(Y^K X^K)^m Y^K W \right) Q_0 \right\| < \varepsilon. \tag{12}$$

However, by construction $(*')$, $\forall k \; \forall m \; \omega(x^k y^k)^m z \in L$ and $\omega y^k (x^k y^k)^m z \notin L$. This requires existence of $\varepsilon > 0$, such that

$$\forall m \; \left\| \left(Z(Y^K X^K)^m W - Z(Y^K X^K)^m Y^K W \right) Q_0 \right\| > \varepsilon. \tag{13}$$

This is a contradiction. □

Lemma 2.36. *If a regular language is of type $(*'')$, it is not recognizable by any PRA-C.*

Proof. Proof is nearly identical to that of Lemma 2.35. Consider a PRA-C which recognizes the language L of type $(*'')$. We prove that for words x, y exists constant K, such that for every ε exists m, such that for two words $\xi_1 = \omega(x^K(xy)^K)^m z$ and $\xi_2 = \omega(x^K(xy)^K)^m x^K z$, $|p_{\xi_1} - p_{\xi_2}| < \varepsilon$. We can choose z, such that $\xi_1 \in L$ iff $\xi_2 \notin L$. □

Theorem 2.37. *If a regular language is of type $(*)$, it is not recognizable by any PRA-C.*

Proof. By Lemmas 2.19, 2.35, 2.36. □

We proved (Lemma 2.19) that the construction of type (*) is a generalization the construction proposed by [BP 99]. Also it can be easily noticed, that the type (*) construction is a generalization of construction proposed by [AKV 00]. (Constructions of [BP 99] and [AKV 00] characterize languages, not recognized by measure-many quantum finite automata of [KW 97].)

Corollary 2.38. *Languages (a,b)*a and a(a,b)* are not recognized by PRA-C.*

Proof. Both languages are of type (*). □

Corollary 2.39. *Class of languages recognizable by PRA-C is not closed under homomorphisms.*

Proof. Consider a homomorphism $a \to a$, $b \to b$, $c \to a$. Similarly as in Theorem 2.13, the language (a,b)*cc* is recognizable by a PRA-C. (Take $n = 2$, $V_a = V_{a_1}$, $V_b = V_{a_1}$, $V_c = V_{a_2}$ from Theorem 2.13, $Q_F = \{q_1\}$) However, by Corollary 2.38 the language (a,b)*aa*=(a,b)*a is not recognizable. □

3 1-way Probabilistic Reversible DH-Automata

Definition 3.1. *The definition differs from one for PRA-C (Definition 2.1) by the following: languages are recognized according to Definition 1.4.*

It is easy to see that the class of languages recognized by PRA-C is a proper subclass of languages recognized by PRA-DH. For example, the language a(a,b)* is recognizable by PRA-DH. However, the following theorem holds:

Theorem 3.2. *Language (a,b)*a is not recognized by PRA-DH.*

Proof. Assume from the contrary that such automaton exists. While reading any sequence of a and b, this automaton can halt only with some probability p strictly less then 1, so accepting and rejecting probabilities may differ only by 1-p, because any word belonging to the language is not dependent on any prefix. Therefore for each $\varepsilon > 0$ we can find that after reading of a prefix of certain length, the total probability to halt while continue reading the word is less then ε. In this case we can apply similar techniques as in the proof of Lemma 2.35, such that for words x, y exists constant K, such that for every ε exists s, such that for two words $\xi_1 = \omega(x^K(xy)^K)^s z$ and $\xi_2 = \omega(x^K(xy)^K)^s x^K z$, $|p_{\xi_1} - p_{\xi_2}| < \varepsilon$. □

4 Alternative Approach to Finite Reversible Automata and 1.5-way Probabilistic Reversible Automata

Let us consider automaton $A' = (Q, \Sigma, q_0, Q_F, \delta')$ that can be obtained from a probabilistic automaton $A = (Q, \Sigma, q_0, Q_F, \delta)$ by specifying
$\delta'(q, \sigma, q') = \delta(q', \sigma, q)$ for all q', σ and q.

If A' is a valid probabilistic automaton then we can call A and A' probabilistic reversible automata.

Definition 4.1. *An automaton of some type is called weakly reversible if the reverse of its transition function corresponds to the transition function of a valid automaton of the same type.*

Note: in case of deterministic automaton where $\delta : Q \times \Gamma \times Q \longrightarrow \{0,1\}$ this property means that A' is still deterministic automaton, not nondeterministic.

In case of one-way automata it is easy to check that this definition is equivalent to the one in Section 2.

We give an example that illustrates that in case of 1.5-way automata these definitions are different.

Definition 4.2. *1.5-way probabilistic weakly reversible C-automaton $A = (Q, \Sigma, q_0, Q_F, \delta)$ is specified by Q, Σ, q_0, Q_F defined as in 1-way PRA-C Definition 2.1, and a transition function*

$$\delta : Q \times \Gamma \times Q \times D \longrightarrow \mathbb{R}_{[0,1]},$$

where Γ defined as in 1-way PRA-C definition and $D = \{0,1\}$ denotes whether automaton stays on the same position or moves one letter ahead on the input tape. Furthermore, transition function satisfies the following requirements:

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \sum_{q \in Q, d \in D} \delta(q_1, \sigma_1, q, d) = 1 \tag{14}$$

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \sum_{q \in Q, d \in D} \delta(q, \sigma_1, q_1, d) = 1 \tag{15}$$

Definition 4.3. *1.5-way probabilistic reversible C-automaton $A = (Q, \Sigma, q_0, Q_F, \delta)$ is specified by Q, Σ, q_0, Q_F defined as in 1-way PRA-C Definition 2.1, and a transition function*

$$\delta : Q \times \Gamma \times Q \times D \longrightarrow \mathbb{R}_{[0,1]},$$

where Γ defined as in 1-way PRA-C definition and $D = \{0,1\}$ denotes whether automaton stays on the same position or moves one letter ahead on the input tape. Furthermore, transition function satisfies the following requirements:

$$\forall (q_1, \sigma_1) \in Q \times \Gamma \sum_{q \in Q, d \in D} \delta(q_1, \sigma_1, q, d) = 1 \tag{16}$$

$$\forall (q_1, \sigma_1, \sigma_2) \in Q \times \Gamma^2 \sum_{q \in Q} \delta(q, \sigma_1, q_1, 0) + \sum_{q \in Q, \sigma \in \Gamma} \delta(q, \sigma_2, q_1, 1) = 1 \tag{17}$$

Theorem 4.4. *Language (a,b)*a is recognizable by 1.5-way weakly reversible PRA-C.*

Proof. The $Q = \{q_0, q_1\}$, $Q_F = \{q_1\}$, δ is defined as follows

$\delta(q_0, a, q_0, 0) = \frac{1}{2}\ \delta(q_0, a, q_1, 1) = \frac{1}{2}\ \delta(q_1, a, q_0, 0) = \frac{1}{2}\ \delta(q_1, a, q_1, 1) = \frac{1}{2}$
$\delta(q_0, b, q_0, 1) = \frac{1}{2}\ \delta(q_0, b, q_1, 0) = \frac{1}{2}\ \delta(q_1, b, q_0, 1) = \frac{1}{2}\ \delta(q_1, b, q_1, 0) = \frac{1}{2}$
$\delta(q_0, \$, q_0, 1) = 1\ \delta(q_1, \$, q_1, 1) = 1$

It is easy to check that such automaton moves ahead according to the transition of the following deterministic automaton

$\delta(q_0, a, q_1, 1) = 1\ \delta(q_1, a, q_1, 1) = 1$
$\delta(q_0, b, q_0, 1) = 1\ \delta(q_1, b, q_0, 1) = 1$
$\delta(q_0, \$, q_0, 1) = 1\ \delta(q_1, \$, q_1, 1) = 1$

So the probability of wrong answer is 0. The probability to be at the m-th position of the input tape after n steps of calculation for $m \leq n$ is C_n^m. Therefore it is necessary no more than $O(n * \log(p))$ steps to reach the end of the word of length n (and so obtain correct answer) with probability $1 - \frac{1}{p}$. □

5 A Classification of Reversible Automata

We propose the following classification for finite 1-way reversible automata:

	C-automata	DH-automata
Deterministic Automata	Permutation Automata [HS 66,T 68] (DRA-C)	Reversible Finite Automata [AF 98] (DRA-DH)
Quantum Automata with Pure States	Measure-Once Quantum Finite Automata [MC 97] (QRA-P-C)	Measure-Many Quantum Finite Automata [KW 97] (QRA-P-DH)
Probabilistic Automata	Probabilistic Reversible C-automata (PRA-C)	Probabilistic Reversible DH-automata (PRA-DH)
Quantum Finite Automata with Mixed States	not considered yet (QRA-M-C)	Enhanced Quantum Finite Automata [N 99] (QRA-M-DH)

Language class problems are solved for DRA-C, DRA-DH, QRA-P-C, for the rest types they are still open. Every type of DH-automata may simulate the corresponding type of C-automata.

Generally, language classes recognized by C-automata are closed under boolean operations (though this is open for QRA-M-C), while DH-automata are not (though this is open for QRA-M-DH and possibly for PRA-DH).

Definition 5.1. *We say that a unitary matrix U is a* prototype *for a doubly stochastic matrix S, if* $\forall i, j\ |U_{i,j}|^2 = S_{i,j}$.

Not every doubly stochastic matrix has a unitary prototype. Such matrix is, for example, $\begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$.

In Introduction, we demonstrated some relation between PRA-C and QRA-M-DH (and hence, QRA-M-C). However, due to the example above, we do not know exactly, whether every PRA-C can be simulated by QRA-M-C, or whether every PRA-DH can be simulated by QRA-M-DH.

Theorem 5.2. *If all matrices of a PRA-C have unitary prototypes, then the PRA-C may be simulated by a QRA-M-C and by a QRA-M-DH.*

Proof. Trivial. □

Theorem 5.3. *If all matrices of a PRA-DH have unitary prototypes, then the PRA-DH may be simulated by a QRA-M-DH.*

Proof. Trivial. □

References

[ABFK 99] A. Ambainis, R. Bonner, R. Freivalds, A. Ķikusts. Probabilities to Accept Languages by Quantum Finite Automata. *COCOON 1999, Lecture Notes in Computer Science*, 1999, Vol. 1627, pp. 174-183.
http://arxiv.org/abs/quant-ph/9904066

[AF 98] A. Ambainis, R. Freivalds. 1-Way Quantum Finite Automata: Strengths, Weaknesses and Generalizations. *Proc. 39th FOCS*, 1998, pp. 332-341.
http://arxiv.org/abs/quant-ph/9802062

[AKV 00] A. Ambainis, A. Ķikusts, M. Valdats. On the Class of Languages Recognizable by 1-Way Quantum Finite Automata. *STACS 2001, Lecture Notes in Computer Science*, 2001, Vol. 2010, pp. 75-86.
http://arxiv.org/abs/quant-ph/0009004

[BP 99] A. Brodsky, N. Pippenger. Characterizations of 1-Way Quantum Finite Automata.
http://arxiv.org/abs/quant-ph/9903014

[GS 97] C. M. Grinstead, J. L. Snell. Introduction to Probability. *American Mathematical Society*, 1997.
http://www.dartmouth.edu/~chance/teaching_aids/articles.html

[HW 79] G. H. Hardy, E. M. Wright. An Introduction to the Theory of Numbers. Fifth Edition. *Oxford University Press*, 1979.

[HS 66] J. Hartmanis, R. E. Stearns. Algebraic Structure Theory of Sequential Machines. *Prentice Hall*, 1966.

[KS 76] J. G. Kemeny, J. L. Snell. Finite Markov Chains. *Springer Verlag*, 1976.

[KW 97] A. Kondacs, J. Watrous. On The Power of Quantum Finite State Automata. *Proc. 38th FOCS*, 1997, pp. 66-75.

[MC 97] C. Moore, J. P. Crutchfield. Quantum Automata and Quantum Grammars. *Theoretical Computer Science*, 2000, Vol. 237(1-2), pp. 275-306.
http://arxiv.org/abs/quant-ph/9707031

[N 99] A. Nayak. Optimal Lower Bounds for Quantum Automata and Random Access Codes. *Proc. 40th FOCS*, 1999, pp. 369-377.
http://arxiv.org/abs/quant-ph/9904093

[R 63] M. O. Rabin. Probabilistic Automata. *Information and Control*, 1963, Vol. 6(3), pp. 230-245.

[T 68] G. Thierrin. Permutation Automata. *Mathematical Systems Theory*, 1968, Vol. 2(1), pp. 83-90.

A End-Marker Theorems for PRA-C Automata

We denote a PRA-C with both end-markers as #,$-PRA-C. We denote a PRA-C with left end-marker only as #-PRA-C.

Theorem A.1. *Let A be a #,$-PRA-C, which recognizes a language L. There exists a #-PRA-C which recognizes the same language.*

Proof. Suppose $A = (Q, \Sigma, q_0, Q_F, \delta)$, where $|Q| = n$. A recognizes L with interval (p_1, p_2). We construct the following automaton $A' = (Q', \Sigma, q_{0,0}, Q'_F, \delta')$ with mn states. Informally, A' equiprobably simulates m copies of the automaton A.

$Q' = \{q_{0,0}, \ldots, q_{0,m-1}, q_{1,0}, \ldots, q_{1,m-1}, \ldots, q_{n-1,0}, \ldots, q_{n-1,m-1}\}$.

If $\sigma \neq \#$, $\delta'(q_{i,k}, \sigma, q_{j,l}) = \begin{cases} \delta(q_i, \sigma, q_j), \text{ if } k = l \\ 0, \text{ if } k \neq l. \end{cases}$

Otherwise, $\delta'(q_{0,0}, \#, q_{j,l}) = \frac{1}{m}\delta(q_0, \#, q_j)$, and if $q_{i,k} \neq q_{0,0}$, $\delta'(q_{i,k}, \#, q) = \frac{1-\delta'(q_{0,0}, \#, q)}{mn-1}$. Function δ' satisfies the requirements (1) and (2) of Definition 2.1.

We define Q'_F as follows. A state $q_{i,k} \in Q'_F$ if and only if $0 \leq k < mp(q_i)$, where $p(q_i) \stackrel{\text{def}}{=} \sum_{q \in Q_F} \delta(q_i, \$, q)$.

Suppose $\#\omega\$$ is an input word. Having read $\#\omega$, A is in superposition $\sum_{i=0}^{n-1} a_i^\omega q_i$. After A has read \$, $\#\omega\$$ is accepted with probability $p_\omega = \sum_{i=0}^{n-1} a_i^\omega p(q_i)$.

On the other hand, having read $\#\omega$, A' is in superposition $\frac{1}{m} \sum_{j=0}^{m-1} \sum_{i=0}^{n-1} a_i^\omega q_{i,j}$.

So the input word $\#\omega$ is accepted with probability $p'_\omega = \frac{1}{m} \sum_{i=0}^{n-1} a_i^\omega \lceil mp(q_i) \rceil$.

Consider $\omega \in L$. Then $p'_\omega = \frac{1}{m} \sum_{i=0}^{n-1} a_i^\omega \lceil mp(q_i) \rceil \geq \sum_{i=0}^{n-1} a_i^\omega p(q_i) = p_\omega \geq p_2$.

Consider $\xi \notin L$. Then $p'_\xi = \frac{1}{m} \sum_{i=0}^{n-1} a_i^\xi \lceil mp(q_i) \rceil < \sum_{i=0}^{n-1} a_i^\xi p(q_i) + \frac{1}{m} \sum_{i=0}^{n-1} a_i^\xi = p_\xi + \frac{1}{m} \leq p_1 + \frac{1}{m}$.

Therefore A' recognizes L with bounded error, provided $m > \frac{1}{p_2 - p_1}$. □

Now we are going to prove that PRA-C without end-markers recognize the same languages as #-PRA-C automata.

If A is a #-PRA-C, then, having read the left end-marker #, the automaton simulates some other automata $A_0, A_1, \ldots, A_{m-1}$ with positive probabilities $p_0, \ldots, p_{m-1}$, respectively. $A_0, A_1, \ldots, A_{m-1}$ are automata without end-markers. By $p_{i,\omega}$, $0 \leq i < m$, we denote the probability that the automaton A_i accepts the word ω.

We prove the following lemma first.

Lemma A.2. *Suppose A' is a #-PRA-C which recognizes a language L with interval (a_1, a_2). Then for every ε, $0 < \varepsilon < 1$, exists a #-PRA-C A which recognizes L with interval (a_1, a_2), such that*

a) if $\omega \in L$, $p_{0,\omega} + p_{1,\omega} + \ldots + p_{n-1,\omega} > \frac{a_2 n}{1+\varepsilon}$

b) if $\omega \notin L$, $p_{0,\omega} + p_{1,\omega} + \ldots + p_{n-1,\omega} < \frac{a_1 n}{1-\varepsilon}$.

Here n ***is the number of automata without end-markers, being simulated by*** A, ***and*** $p_{i,\omega}$ ***is the probability that*** i***-th simulated automaton*** A_i ***accepts*** ω.

Proof. Suppose a #-PRA-C A' recognizes a language L with interval (a_1, a_2). Having read the symbol #, A' simulates automata $A'_0, \ldots, A'_{m-1}$ with probabilities $p'_0, \ldots, p'_{m-1}$, respectively. We choose ε, $0 < \varepsilon < 1$.

By Dirichlet's principle ([HW 79], p. 170), $\forall \varphi > 0$ exists $n \in \mathbb{N}^+$ such that $\forall i$ $p'_i n$ differs from some positive integer by less than φ.

Let $0 < \varphi < \min\left(\frac{1}{m}, \varepsilon\right)$. Let g_i be the nearest integer of $p'_i n$. So $|p'_i n - g_i| < \varphi$ and $\left|\frac{p'_i}{g_i} - \frac{1}{n}\right| < \frac{\varphi}{n g_i} \leq \frac{\varphi}{n}$. Since $|p'_i n - g_i| < \varphi$, we have $\left|n - \sum_{i=0}^{m-1} g_i\right| < \varphi m < 1$. Therefore, since $g_i \in \mathbb{N}^+$, $\sum_{i=0}^{m-1} g_i = n$.

Now we construct the #-PRA-C A, which satisfies the properties expressed in Lemma A.2. For every i, we make g_i copies of A'_i. Having read #, for every i A simulates each copy of A'_i with probability $\frac{p'_i}{g_i}$. The construction of $V_\#$ is equivalent to that used in the proof of Lemma 2.7. Therefore A is characterized by doubly stochastic matrices. A recognizes L with the same interval as A', i.e., (a_1, a_2).

Using new notations, A simulates n automata $A_0, A_1, \ldots, A_{n-1}$ with probabilities $p_0, p_1, \ldots, p_{n-1}$, respectively. Note that $\forall i$ $\left|p_i - \frac{1}{n}\right| < \frac{\varphi}{n}$. Let $p_{i,\omega}$ be the probability that A_i accepts the word ω.

Consider $\omega \in L$. We have $p_0 p_{0,\omega} + p_1 p_{1,\omega} + \ldots + p_{n-1} p_{n-1,\omega} \geq a_2$. Since $p_i < \frac{1+\varphi}{n}$, $\frac{1+\varphi}{n}(p_{0,\omega} + p_{1,\omega} + \ldots + p_{n-1,\omega}) > a_2$. Hence

$$p_{0,\omega} + p_{1,\omega} + \ldots + p_{n-1,\omega} > \frac{a_2 n}{1+\varphi} > \frac{a_2 n}{1+\varepsilon}.$$

Consider $\xi \notin L$. We have $p_0 p_{0,\xi} + p_1 p_{1,\xi} + \ldots + p_{n-1} p_{n-1,\xi} \leq a_1$. Since $p_i > \frac{1-\varphi}{n}$, $\frac{1-\varphi}{n}(p_{0,\xi} + p_{1,\xi} + \ldots + p_{n-1,\xi}) < a_1$. Hence

$$p_{0,\xi} + p_{1,\xi} + \ldots + p_{n-1,\xi} < \frac{a_1 n}{1-\varphi} < \frac{a_1 n}{1-\varepsilon}.$$

□

Theorem A.3. *Let* A *be a #-PRA-C, which recognizes a language* L. *There exists a PRA-C without end-markers, which recognizes the same language.*

Proof. Consider a #-PRA-C which recognizes a language L with interval (a_1, a_2). Using Lemma A.2, we choose ε, $0 < \varepsilon < \frac{a_2 - a_1}{a_2 + a_1}$, and construct an automaton A' which recognizes L with interval (a_1, a_2), with the following properties.

Having read #, A' simulates $A'_0, \ldots, A'_{m-1}$ with probabilities $p'_0, \ldots, p'_{m-1}$, respectively. $A'_0, \ldots, A'_{m-1}$ are automata without end-markers. A'_i accepts ω with probability $p'_{i,\omega}$. If $\omega \in L$, $p'_{0,\omega} + p'_{1,\omega} + \ldots + p'_{m-1,\omega} > \frac{a_2 m}{1+\varepsilon}$. Otherwise, if $\omega \notin L$, $p'_{0,\omega} + p'_{1,\omega} + \ldots + p'_{m-1,\omega} < \frac{a_1 m}{1-\varepsilon}$.

That also implies that for every $n = km$, $k \in \mathbb{N}^+$, we are able to construct a #-PRA-C A which recognizes L with interval (a_1, a_2), such that

a) if $\omega \in L$, $p_{0,\omega} + p_{1,\omega} + \ldots + p_{n-1,\omega} > \frac{a_2 n}{1+\epsilon}$;
b) if $\omega \notin L$, $p_{0,\omega} + p_{1,\omega} + \ldots + p_{n-1,\omega} < \frac{a_1 n}{1-\epsilon}$.

A simulates $A_0, \ldots, A_{n-1}$. Let us consider the system $F_n = (A_0, \ldots, A_{n-1})$. Let $\delta = \frac{1}{2}(a_1 + a_2)$. Since $\epsilon < \frac{a_2 - a_1}{a_2 + a_1}$, $\frac{a_2}{1+\epsilon} > \delta$ and $\frac{a_1}{1-\epsilon} < \delta$. As in the proof of Theorem 2.6, we define that the system accepts a word, if more than $n\delta$ automata in the system accept the word.

Let us take η_0, such that $0 < \eta_0 < \frac{a_2}{1+\epsilon} - \delta < \delta - \frac{a_1}{1-\epsilon}$.

Consider $\omega \in L$. We have that $\sum_{i=0}^{n-1} p_{i,\omega} > \frac{a_2 n}{1+\epsilon} > n\delta$. As a result of reading ω, μ_n^ω automata in the system accept the word, and the rest reject it. The system has accepted the word, if $\frac{\mu_n^\omega}{n} > \delta$. Since $0 < \eta_0 < \frac{a_2}{1+\epsilon} - \delta < \frac{1}{n}\sum_{i=0}^{n-1} p_{i,\omega} - \delta$, we have

$$P\left\{\frac{\mu_n^\omega}{n} > \delta\right\} \geq P\left\{\left|\frac{\mu_n^\omega}{n} - \frac{1}{n}\sum_{i=0}^{n-1} p_{i,\omega}\right| < \eta_0\right\}. \tag{18}$$

If we look on $\frac{\mu_n^\omega}{n}$ as a random variable X, $E(X) = \frac{1}{n}\sum_{i=0}^{n-1} p_{i,\omega}$ and variance $V(X) = \frac{1}{n^2}\sum_{i=0}^{n-1} p_{i,\omega}(1 - p_{i,\omega})$, therefore Chebyshev's inequality yields the following:

$$P\left\{\left|\frac{\mu_n^\omega}{n} - \frac{1}{n}\sum_{i=0}^{n-1} p_{i,\omega}\right| \geq \eta_0\right\} \leq \frac{1}{n^2\eta_0^2}\sum_{i=0}^{n-1} p_{i,\omega}(1 - p_{i,\omega}) \leq \frac{1}{4n\eta_0^2}.$$

That is equivalent to $P\left\{\left|\frac{\mu_n^\omega}{n} - \frac{1}{n}\sum_{i=0}^{n-1} p_{i,\omega}\right| < \eta_0\right\} \geq 1 - \frac{1}{4n\eta_0^2}$. So, taking into account (18),

$$P\left\{\frac{\mu_n^\omega}{n} > \delta\right\} \geq 1 - \frac{1}{4n\eta_0^2}. \tag{19}$$

On the other hand, consider $\xi \notin L$. So $\sum_{i=0}^{n-1} p_{i,\xi} < \frac{a_1 n}{1-\epsilon} < n\delta$. Again, since $0 < \eta_0 < \delta - \frac{a_1}{1-\epsilon} < \delta - \frac{1}{n}\sum_{i=0}^{n-1} p_{i,\xi}$,

$$P\left\{\frac{\mu_n^\xi}{n} > \delta\right\} \leq P\left\{\left|\frac{\mu_n^\xi}{n} - \frac{1}{n}\sum_{i=0}^{n-1} p_{i,\xi}\right| \geq \eta_0\right\} \leq \frac{1}{4n\eta_0^2}. \tag{20}$$

The constant η_0 does not depend on n and n may be chosen sufficiently large. Therefore, by (19) and (20), the system F_n recognizes L with bounded error, if $n > \frac{1}{2\eta_0^2}$.

Following a way identical to that used in the proof of Theorem 2.6, it is possible to construct a single PRA-C without end-markers, which simulates the system F_n and therefore recognizes the language L. □

Richard Bonner and Rūsiņš Freivalds (Eds.)

Quantum Computation and Learning

International Workshop
27-29 May 2000
Sundbyholms Slott, Sweden

Proceedings

Department of Mathematics and Physics
Mälardalen University, Eskilstuna-Västerås, Sweden

Institute of Mathematics and Computer Science
University of Latvia, Riga, Latvia

Workshop sponsored by The Swedish Institute

Contents

Quantum Computation

Learning

On Quantum Pushdown Automata

Marats Golovkins

Institute of Mathematics and Computer Science, University of Latvia, Raiņa bulv. 29, Riga, Latvia*
marats@cclu.lv

Abstract. Quantum finite automata, as well as quantum pushdown automata were first introduced by C. Moore, J. P. Crutchfield [MC 97]. In this paper we introduce the notion of quantum pushdown automata in a non-equivalent way, including unitarity criteria, by using the definition of quantum finite automata of [KW 97]. It is established that the unitarity criteria of quantum pushdown automata are not equivalent to the corresponding unitarity criteria of quantum Turing machines [BV 97]. Finally we present some simple languages recognized by quantum pushdown automata, two of them is not recognizable by deterministic pushdown automata and one seems to be not recognizable by probabilistic pushdown automata as well.

1 Introduction

Nobel prize winner physicist R. Feynman asked in 1982, what effects may have the principles of quantum mechanics on computation [Fe 82]. He gave arguments that it may require exponential time to simulate quantum mechanical processes on classical computers. This served as a basis to the opinion that quantum computers may have advantages versus classical ones. It was in 1985, when D. Deutsch introduced the notion of quantum Turing machine [De 85] and proved that quantum Turing machines compute the same recursive functions as classical deterministic Turing machines do. P. Shor discovered that by use of quantum algorithms it is possible to factorize large integers and compute discrete logarithms in a polynomial time [Sh 94], what resulted into additional interest in quantum computing and attempts to create quantum computers. First steps have been made to this direction, and first quantum computers which memory is limited by a few quantum bits have been constructed [KLMT 99]. To make quantum computers with larger memory feasible, one of the problems is to minimize error possibilities in quantum bits. Quantum error correction methods are developed [CRSS 98] which would enable quantum computers with larger quantum memory.

Quantum mechanics differs from the classical physics substantially. It is enough to mention *Heisenberg's uncertainty principle*, which states that it is

* Research partially supported by the Latvian Council of Science, grant 96-0282; European Comission, contract IST-1999-11234; Swedish Institute, project ML2000

impossible to get information about different parameters of quantum particle simultaneously precisely. Another well known distinction is the impossibility to observe quantum object without changing it.

Fundamental concept of quantum information theory is *quantum bit*. Classical information theory is based on classical bit, which has two states 0 and 1. The next step is *probabilistic bit*, which can be 0 with probability α and 1 with probability β, where $\alpha + \beta = 1$. Quantum bit or *qbit* is similar to probabilistic bit with the difference that α and β are complex numbers with the property $|\alpha|^2 + |\beta|^2 = 1$. It is common to denote qbit as $\alpha|0\rangle + \beta|1\rangle$. As a result of *measurement*, we get 0 with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$.

Every computation done on qbits is accomplished by means of unitary operators. Informally, every unitary operator can be interpreted as a revolution in complex space. Therefore one of the basic properties of unitary operators is that every quantum computing process not disturbed by measurements is reversible. Unitarity is rather hard requirement which complicates programming of quantum devices. The following features of quantum computers are most important:

1. Information is represented by qbits.
2. Any step of computation can be represented as a unitary operation, therefore computation is reversible.
3. Quantum information cannot be copied.
4. Quantum parallelism; quantum computer can compute several paths simultaneously, however as a result of measurement it is possible to get the results of only one computation path.

Quantum finite automata [KW 97] are considered to be the most elementary model of quantum device. We shall denote this as QFA further in the paper. Quantum pushdown automata were first defined in [MC 97], however the authors do not mention clear criteria to ensure unitarity in the sense of respective definition. The definition in [MC 97] is not equivalent to that given in this paper.

The following notations will be used further in the paper:
z^* is the complex conjugate of a complex number z.
U^* is the Hermitian conjugate of a matrix U.
I is the identity matrix.
ε is empty word.

Definition 1.1. *Matrix U is called* unitary, *if $UU^* = U^*U = I$.*

If U is a finite matrix, then $UU^* = I$ iff $U^*U = I$. However this is not true for infinite matrices:

Example 1.1.

$$U = \begin{pmatrix} \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & \dots \\ \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & 0 & 1 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

Here $U^*U = I$ but $UU^* \neq I$.

Lemma 1.1. *If infinite matrices A, B, C have finite number of nonzero elements in each row and column, then their multiplication is associative: $(AB)C = A(BC)$.*

Proof. The element of matrix $(AB)C$ in i-th row and j-th column is $k_{ij} = \sum_{s=1}^{\infty}\sum_{r=1}^{\infty} a_{ir}b_{rs}c_{sj}$. The element of matrix $A(BC)$ in the same row and column is $l_{ij} = \sum_{r=1}^{\infty}\sum_{s=1}^{\infty} a_{ir}b_{rs}c_{sj}$. As in the each row and column of matrices A, B, C there is a finite number of nonzero elements, it is also finite in the given series. Therefore the elements of the series can be rearranged, and $k_{ij} = l_{ij}$. □

As noted further in the paper infinite matrices with finite number of nonzero elements in each row and column describe the work of pushdown automata. In further theorems there are stated some properties of such matrices.

Lemma 1.2. *If $U^*U = I$, then the norm of any row in the matrix U does not exceed* 1.

Proof. Let us consider the matrix $S = UU^*$. The element of this matrix $s_{ij} = \langle r_j|r_i\rangle$, where r_i is i-th row of the matrix U. Let us consider the matrix $T = S^2$. The diagonal element of this matrix is

$$t_{ii} = \sum_{k=1}^{\infty} s_{ik}s_{ki} = \sum_{k=1}^{\infty}\langle r_k|r_i\rangle\langle r_i|r_k\rangle = \sum_{k=1}^{\infty}|\langle r_k|r_i\rangle|^2.$$

On the other hand, taking into account Theorem 1.1, we get that

$$T = S^2 = (UU^*)(UU^*) = U(U^*U)U^* = UU^* = S.$$

Therefore $t_{ii} = s_{ii} = \langle r_i|r_i\rangle$. It means that

$$\sum_{k=1}^{\infty}|\langle r_k|r_i\rangle|^2 = \langle r_i|r_i\rangle. \tag{1}$$

This implies that every element of series (1) does not exceed $\langle r_i|r_i\rangle$. Hence $|\langle r_i|r_i\rangle|^2 = \langle r_i|r_i\rangle^2 \leq \langle r_i|r_i\rangle$. The last inequality implies that $0 \leq \langle r_i|r_i\rangle \leq 1$. Therefore $|r_i| \leq 1$. □

Lemma 1.3. *Let us assume that $U^*U = I$. Then the rows of the matrix U are orthogonal iff every row of the matrix has norm* 0 *or* 1.

Proof. Let us assume that the rows of the matrix U are orthogonal. Let us consider equation (1) from the proof of Theorem 1.2, i.e., $\sum_{k=1}^{\infty}|\langle r_k|r_i\rangle|^2 = \langle r_i|r_i\rangle$.

As the rows of the matrix U are orthogonal, $\sum_{k=1}^{\infty} |\langle r_k|r_i\rangle|^2 = |\langle r_i|r_i\rangle|^2$. Hence $\langle r_i|r_i\rangle^2 = \langle r_i|r_i\rangle$, i.e., $\langle r_i|r_i\rangle = 0$ or $\langle r_i|r_i\rangle = 1$. Therefore $|r_i| = 0$ or $|r_i| = 1$.

Let as assume that every row of the matrix has norm 0 or 1. Then $\langle r_i|r_i\rangle^2 = \langle r_i|r_i\rangle$ and in compliance with the equation (1), $\sum_{k\in\mathbb{N}^+\setminus\{i\}} |\langle r_k|r_i\rangle|^2 = 0$. This implies that $\forall k \neq i\ |\langle r_k|r_i\rangle| = 0$. Hence the rows of the matrix are orthogonal. □

Lemma 1.4. *The matrix U is unitary iff $U^*U = I$ and its rows are normalized.*

Proof. Let us assume that the matrix U is unitary. Then in compliance with Definition 1.1, $U^*U = I$ and $UU^* = I$, i.e, the rows of the matrix are orthonormal.

Let us assume that $U^*U = I$ and the rows of the matrix are normalized. Then in compliance with Theorem 1.3 the rows of the matrix are orthogonal. Hence $UU^* = I$ and the matrix is unitary. □

2 Quantum pushdown automata

Definition 2.1. *A quantum pushdown automaton (QPA)*
$A = (Q, \Sigma, T, q_0, Q_a, Q_r, \delta)$ is specified by a finite set of states Q, a finite input alphabet Σ and a stack alphabet T, an initial state $q_0 \in Q$, sets $Q_a \subset Q$, $Q_r \subset Q$ of accepting and rejecting states, respectively, with $Q_a \cap Q_r = \emptyset$, and a transition function

$$\delta : Q \times \Gamma \times \Delta \times Q \times \{\downarrow, \rightarrow\} \times \Delta^* \longrightarrow \mathbb{C}_{[0,1]},$$

where $\Gamma = \Sigma \cup \{\#, \$\}$ is the input tape alphabet of A and $\#, \$$ are end-markers not in Σ, $\Delta = T \cup \{Z_0\}$ is the working stack alphabet of A and $Z_0 \notin T$ is the stack base symbol; $\{\downarrow, \rightarrow\}$ is the set of directions of input tape head. The automaton must satisfy conditions of well-formedness, *which will be expressed below. Furthermore, the transition function is restricted to a following requirement:*

If $\delta(q, \alpha, \beta, q', d, \omega) \neq 0$, then

1. *$|\omega| \leq 2$;*
2. *if $|\omega| = 2$, then $\omega_1 = \beta$;*
3. *if $\beta = Z_0$, then $\omega \in Z_0 T^*$;*
4. *if $\beta \neq Z_0$, then $\omega \in T^*$.*

Definition 2.1 utilizes that of classical pushdown automata from [Gu 89].

Well-formedness conditions 2.1.

1. Local probability condition.

$$\forall (q_1, \sigma_1, \tau_1) \in Q \times \Gamma \times \Delta \quad \sum_{(q,d,\omega)\in Q\times\{\downarrow,\rightarrow\}\times\Delta^*} |\delta(q_1, \sigma_1, \tau_1, q, d, \omega)|^2 = 1. \tag{2}$$

2. Orthogonality of column vectors condition.

$$\text{For all triples } (q_1,\sigma_1,\tau_1)\neq(q_2,\sigma_1,\tau_2) \text{ in } Q\times\Gamma\times\Delta$$
$$\sum_{(q,d,\omega)\in Q\times\{\downarrow,\rightarrow\}\times\Delta^*}\delta^*(q_1,\sigma_1,\tau_1,q,d,\omega)\delta(q_2,\sigma_1,\tau_2,q,d,\omega)=0. \quad (3)$$

3. Row vectors norm condition.

$$\forall(q_1,\sigma_1,\sigma_2,\tau_1,\tau_2)\in Q\times\Gamma^2\times\Delta^2$$
$$\sum_{(q,\tau,\omega)\in Q\times\Delta\times\{\varepsilon,\tau_2,\tau_1\tau_2\}}|\delta(q,\sigma_1,\tau,q_1,\rightarrow,\omega)|^2+|\delta(q,\sigma_2,\tau,q_1,\downarrow,\omega)|^2=1. \quad (4)$$

4. Separability condition I.

$$\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_1,\tau_2)\in Q\times\Gamma\times\Delta,\ \forall\tau_3\in\Delta$$
$$a)\ \sum_{(q,d,\tau)\in Q\times\{\downarrow,\rightarrow\}\times\Delta}\delta^*(q_1,\sigma_1,\tau_1,q,d,\tau)\delta(q_2,\sigma_1,\tau_2,q,d,\tau_3\tau)+$$
$$+\sum_{(q,d)\in Q\times\{\downarrow,\rightarrow\}}\delta^*(q_1,\sigma_1,\tau_1,q,d,\varepsilon)\delta(q_2,\sigma_1,\tau_2,q,d,\tau_3)=0; \quad (5)$$
$$b)\ \sum_{(q,d)\in Q\times\{\downarrow,\rightarrow\}}\delta^*(q_1,\sigma_1,\tau_1,q,d,\varepsilon)\delta(q_2,\sigma_1,\tau_2,q,d,\tau_2\tau_3)=0. \quad (6)$$

5. Separability condition II.

$$\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_2,\tau_2)\in Q\times\Gamma\times\Delta$$
$$\sum_{(q,\omega)\in Q\times\Delta^*}\delta^*(q_1,\sigma_1,\tau_1,q,\downarrow,\omega)\delta(q_2,\sigma_2,\tau_2,q,\rightarrow,\omega)=0. \quad (7)$$

6. Separability condition III.

$$\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_2,\tau_2)\in Q\times\Gamma\times\Delta,\ \forall\tau_3\in\Delta,\ \forall d_1,d_2\in\{\downarrow,\rightarrow\},\ d_1\neq d_2$$
$$a)\ \sum_{(q,\tau)\in Q\times\Delta}\delta^*(q_1,\sigma_1,\tau_1,q,d_1,\tau)\delta(q_2,\sigma_2,\tau_2,q,d_2,\tau_3\tau)+$$
$$+\sum_{q\in Q}\delta^*(q_1,\sigma_1,\tau_1,q,d_1,\varepsilon)\delta(q_2,\sigma_2,\tau_2,q,d_2,\tau_3)=0; \quad (8)$$
$$b)\ \sum_{q\in Q}\delta^*(q_1,\sigma_1,\tau_1,q,d_1,\varepsilon)\delta(q_2,\sigma_2,\tau_2,q,d_2,\tau_2\tau_3)=0. \quad (9)$$

Let us assume that an automaton is in a state q, its input tape head is above a symbol α and the stack head is above a symbol β. Then the automaton undertakes following actions with an amplitude $\delta(q,\alpha,\beta,q',d,\omega)$:

1. goes into the state q';
2. if $d=\text{`}\rightarrow\text{'}$, moves the input tape head one cell forwards;

3. takes out of the stack the symbol β (deletes it and moves the stack head one cell backwards);
4. starting with the first empty cell, puts into the stack the string ω, moving the stack head $|\omega|$ cells forwards.

Definition 2.2. *The* configuration *of a pushdown automaton is a pair* $|c\rangle = |\nu_i q_j \nu_k, \omega_l\rangle$, *where the automaton is in a state* $q_j \in Q$, $\nu_i\nu_k \in \#\Sigma^*\$$ *is a finite word on the input tape,* $\omega_l \in Z_0 T^*$ *is a finite word on the stack tape, the input tape head is above the first symbol of the word* ν_k *and the stack head is above the last symbol of the word* ω_l.

We shall denote by C the set of all configurations of a pushdown automaton. The set C is countably infinite. Every configuration $|c\rangle$ denotes a basis vector in the space $H_A = l_2(C)$. Therefore a global state of A in the space H_A has a form $|\psi\rangle = \sum\limits_{c\in C} \alpha_c|c\rangle$, where $\sum\limits_{c\in C} |\alpha_c|^2 = 1$ and $\alpha_c \in \mathbb{C}$ denotes the amplitude of a configuration $|c\rangle$. If an automaton is in its global state (superposition) $|\psi\rangle$, then its further step is equivalent to the application of a linear operator (evolution) U_A over the space H_A.

Definition 2.3. *A linear operator* U_A *is defined as follows:*

$$U_A|\psi\rangle = \sum_{c\in C} \alpha_c U_A|c\rangle.$$

If a configuration $c = |\nu_i q_j \sigma \nu_k, \omega_l \tau\rangle$, *then*

$$U_A|c\rangle = \sum_{(q,d,\omega)\in Q\times\{\downarrow,\rightarrow\}\times\Delta^*} \delta(q_j,\sigma,\tau,q,d,\omega)|f(|c\rangle,d,q),\omega_l\omega\rangle,$$

where

$$f(|\nu_i q_j \sigma \nu_k, \omega_l \tau\rangle, d, q) = \begin{cases} \nu_i q \sigma \nu_k, \text{ if } d = \text{`}\downarrow\text{'} \\ \nu_i \sigma q \nu_k, \text{ if } d = \text{`}\rightarrow\text{'}. \end{cases}$$

Remark 2.1. Although a QPA evolution operator matrix is infinite, it has a finite number of nonzero elements in each row and column, as it is possible to reach only a finite number of other configurations from a given configuration within one step, all the same, within one step the given configuration is reachable only from a finite number of different configurations.

Lemma 2.1. *The columns system of a QPA evolution matrix is normalized iff the condition (2), i.e., local probability condition, is satisfied.*

Lemma 2.2. *The columns system of a QPA evolution matrix is orthogonal iff the conditions (3,5,6,7,8,9), i.e., orthogonality of column vectors and separability conditions, are satisfied.*

Lemma 2.3. *The rows system of a QPA evolution matrix is normalized iff the condition (4), i.e., row vectors norm condition, is satisfied.*

Theorem 2.1. *Well-formedness conditions 2.1 are satisfied iff the evolution operator U_A is unitary.*

Proof. Lemmas 2.1, 2.2, 2.3 imply that Well-formedness conditions 2.1 are satisfied iff the columns of the evolution matrix are orthonormal and rows are normalized. In compliance with Theorem 1.4, columns are orthonormal and rows are normalized iff the matrix is unitary. □

Remark 2.2. Well-formedness conditions 2.1 contain the requirement that rows system has to be normalized, which is not necessary in the case of quantum Turing machine [BV 97]. Here is taken into account the fact that the evolution of QPA can violate the unitarity requirement if the row vectors norm condition is omitted.

Example 2.1. A QPA, which evolution matrix columns are orthonormal, however the evolution is not unitary.

$$Q = \{q\},\ \Sigma = \{1\},\ T = \{1\}.$$
$$\delta(q, \#, Z_0, q, \rightarrow, Z_0 1) = 1, \qquad \delta(q, \#, 1, q, \rightarrow, 11) = 1,$$
$$\delta(q, 1, Z_0, q, \rightarrow, Z_0 1) = 1, \qquad \delta(q, 1, 1, q, \rightarrow, 11) = 1,$$
$$\delta(q, \$, Z_0, q, \rightarrow, Z_0 1) = 1, \qquad \delta(q, \$, 1, q, \rightarrow, 11) = 1,$$

other values of arguments yield $\delta = 0$.

By Well-formedness conditions 2.1, the columns of the evolution matrix are orthonormal, but the matrix is not unitary, because the norm of the rows specified by the configurations $|\omega, Z_0\rangle$ is 0.

Even in a case of trivial QPA, it is a cumbersome task to check all the conditions of well-formedness 2.1. It is possible to relax the conditions slightly by introducing a notion of *simplified* QPA.

Definition 2.4. *We shall say that a QPA is* simplified, *if there exists a function $D : Q \longrightarrow \{\downarrow, \rightarrow\}$, and $\delta(q_1, \sigma, \tau, q, d, \omega) = 0$, if $D(q) \neq d$. Therefore the transition function of a simplified QPA is*

$$\varphi(q_1, \sigma, \tau, q, \omega) = \delta(q_1, \sigma, \tau, q, D(q), \omega).$$

Taking into account Definition 2.4, following well-formedness conditions correspond to simplified QPA:

Well-formedness conditions 2.2.

1. Local probability condition.

$$\forall (q_1, \sigma_1, \tau_1) \in Q \times \Gamma \times \Delta$$
$$\sum_{(q,\omega) \in Q \times \Delta^*} |\varphi(q_1, \sigma_1, \tau_1, q, \omega)|^2 = 1. \tag{10}$$

2. Orthogonality of column vectors condition.

$$\text{For all triples } (q_1,\sigma_1,\tau_1) \neq (q_2,\sigma_1,\tau_2) \text{ in } Q\times\Gamma\times\Delta$$
$$\sum_{(q,\omega)\in Q\times\Delta^*} \varphi^*(q_1,\sigma_1,\tau_1,q,\omega)\varphi(q_2,\sigma_1,\tau_2,q,\omega) = 0. \tag{11}$$

3. Row vectors norm condition.

$$\forall(q_1,\sigma_1,\tau_1,\tau_2)\in Q\times\Gamma\times\Delta^2$$
$$\sum_{(q,\tau,\omega)\in Q\times\Delta\times\{\varepsilon,\tau_2,\tau_1\tau_2\}} |\varphi(q,\sigma_1,\tau,q_1,\omega)|^2 = 1. \tag{12}$$

4. Separability condition.

$$\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_1,\tau_2)\in Q\times\Gamma\times\Delta,\ \forall\tau_3\in\Delta$$
$$a)\ \sum_{(q,\tau)\in Q\times\Delta} \varphi^*(q_1,\sigma_1,\tau_1,q,\tau)\varphi(q_2,\sigma_1,\tau_2,q,\tau_3\tau) +$$
$$+\sum_{q\in Q}\varphi^*(q_1,\sigma_1,\tau_1,q,\varepsilon)\varphi(q_2,\sigma_1,\tau_2,q,\tau_3) = 0; \tag{13}$$
$$b)\ \sum_{q\in Q}\varphi^*(q_1,\sigma_1,\tau_1,q,\varepsilon)\varphi(q_2,\sigma_1,\tau_2,q,\tau_2\tau_3) = 0. \tag{14}$$

Theorem 2.2. *The evolution of a simplified QPA is unitary iff Well-formedness conditions 2.2 are satisfied.*

Proof. By Theorem 2.1 and Definition 2.4. □

3 Language recognition

Language recognition for QPA is defined as follows. For a QPA $A = (Q,\Sigma,T,q_0,Q_a,Q_r,\delta)$ we define $C_a = \{|\nu_i q\nu_k,\omega_l\rangle \in C \mid q\in Q_a\}$, $C_r = \{|\nu_i q\nu_k,\omega_l\rangle \in C \mid q\in Q_r\}$, $C_n = C\setminus(C_a\cup C_r)$. E_a, E_r, E_n are subspaces of H_A spanned by C_a, C_r, C_n respectively. We use the observable $\mathcal{O}$ that corresponds to the orthogonal decomposition $H_A = E_a\oplus E_r\oplus E_n$. The outcome of each observation is either "accept" or "reject" or "non-halting".

The language recognition is now defined as follows: For an $x\in\Sigma^*$ we consider as an input $\#x\$$, and assume that the computation starts with A being in the configuration $|q_0\#x\$, Z_0\rangle$. Each computation step consists of two parts. At first the linear operator U_A is applied to the current global state and then the resulting superposition is observed using the observable $\mathcal{O}$ as defined above. If the global state before the observation is $\sum_{c\in C}\alpha_c|c\rangle$, then the probability that the resulting superposition is projected onto the subspace E_i, $i\in\{a,r,n\}$, is $\sum_{c\in C_i}|\alpha_c|^2$. The computation continues until the result of an observation is "accept" or "reject".

Definition 3.1. *We shall say that an automaton is a deterministic reversible pushdown automaton (RPA), if it is a simple QPA with* $\varphi(q_1, \sigma, \tau, q, \omega) \in \{0, 1\}$.

Needless to say, if any language is recognized by a RPA, it is recognized with probability equal to 1.

First, let us consider a language $L_1 = (0,1)^*1$, for which we know that it is not recognizable by QFA [KW 97].

Lemma 3.1. *Language* L_1 *is recognizable by a RPA.*

Proof. Let us consider a deterministic automaton with two states q_0, q_1 and the following transitions: $\delta(q_0, 0) = q_0$, $\delta(q_0, 1) = q_1$, $\delta(q_1, 0) = q_0$, $\delta(q_1, 1) = q_1$.
It is possible to transform this automaton to a RPA, which satisfies the corresponding well-formedness conditions:
$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$, $Q_a = \{q_5\}$, $Q_r = \{q_4\}$, $\Sigma = \{0, 1\}$, $T = \{0, 1\}$.
The states q_0, q_1 have the same semantics as in the deterministic prototype, the only difference is in case input tape symbols 0 or 1 is read, when each transition starting in the state q_0, automaton pushes 0 into stack, whereas in the state q_1 pushes 1. After reaching the endmarking symbol \$, depending on its current state, the automaton goes to the state q_5 or q_6.
Finally, we have to add two more states q_2, q_3 to our RPA, to ensure its unitarity.
Values of the transition function follow:

$$\begin{array}{ll}
\forall \tau \in \Delta\ \forall q \in Q\ \forall \sigma \in \Sigma; & \\
\varphi(q, \#, \tau, q, \tau) = 1, & \\
\varphi(q_0, 0, \tau, q_0, \tau 0) = 1, & \varphi(q_1, 0, \tau, q_0, \tau 1) = 1,\ D(q_0) = \rightarrow, \\
\varphi(q_0, 1, \tau, q_1, \tau 0) = 1, & \varphi(q_1, 1, \tau, q_1, \tau 1) = 1,\ D(q_1) = \rightarrow, \\
\varphi(q_0, \$, \tau, q_4, \tau) = 1, & \varphi(q_1, \$, \tau, q_5, \tau) = 1,\ D(q_2) = \downarrow, \\
\varphi(q_2, 1, \tau, q_0, \tau) = 1, & \varphi(q_3, 0, \tau, q_1, \tau) = 1,\ D(q_3) = \downarrow, \\
\varphi(q_2, \$, \tau, q_2, \tau) = 1, & \varphi(q_3, \$, \tau, q_3, \tau) = 1,\ D(q_4) = \downarrow, \\
\varphi(q_4, \sigma, \tau, q_4, \tau) = 1, & \varphi(q_5, \sigma, \tau, q_5, \tau) = 1,\ D(q_5) = \downarrow, \\
\varphi(q_2, 0, Z, q_0, Z) = 1, & \varphi(q_3, 1, Z, q_1, Z) = 1,\ D(q_6) = \downarrow, \\
\varphi(q_2, 0, 0, q_2, \varepsilon) = 1, & \varphi(q_2, 0, 1, q_3, \varepsilon) = 1, \\
\varphi(q_3, 1, 0, q_2, \varepsilon) = 1, & \varphi(q_3, 1, 1, q_3, \varepsilon) = 1, \\
\varphi(q_4, \$, \tau, q_0, \tau) = 1, & \varphi(q_5, \$, \tau, q_1, \tau) = 1,
\end{array}$$

other values of arguments yield $\delta = 0$.
Let us note that states q_2, q_3 are not reachable from the initial state q_0, however they are necessary to make the automaton unitary. □

Lemma 3.2. *Language* $L_2 = a^*b^*$ *is recognizable by a RPA.*

Proof. Proof is similar to that of Lemma 3.1. □

It seems that the same technique as in the proof of Lemma 3.1 can be used to construct reversible pushdown automata for every regular language by use of deterministic finite automata as a prototype. The number of states of corresponding RPA would be twice as much as in deterministic finite automaton.

Let us consider a language which is not regular, namely,

$$L_3 = \{\omega \in (a,b)^* |\ |\omega|_a = |\omega|_b\},$$

where $|\omega|_i$ denotes the number of occurrences of the symbol i in the word ω.

Lemma 3.3. *Language L_3 is recognizable by a RPA.*

Proof. Our RPA has four states q_0, q_1, q_2, q_3, where q_2 is an accepting state, whereas q_3 - rejecting one. Stack alphabet T consists of two symbols $1, 2$. Stack filled with 1's means that the processed part of the word ω has more occurrences of a's than b's, whereas 2's means that there are more b's than a's. Furthermore, length of the stack word is equal to the difference of number of a's and b's. Empty stack denotes that the number of a's and b's is equal.

Values of the transition function follow:

$$\begin{array}{lll}
\forall q \in Q\ \forall \tau \in \Delta; & & \\
\varphi(q,\#,\tau,q,\tau) = 1, & \varphi(q_0,a,Z,q_0,Z1) = 1, & D(q_0) = \rightarrow, \\
\varphi(q_0,b,Z,q_0,Z2) = 1, & \varphi(q_0,\$,Z,q_2,Z1) = 1, & D(q_1) = \downarrow, \\
\varphi(q_0,a,1,q_0,11) = 1, & \varphi(q_0,b,1,q_1,\varepsilon) = 1, & D(q_2) = \downarrow, \\
\varphi(q_0,\$,1,q_3,1) = 1, & \varphi(q_0,a,2,q_1,\varepsilon) = 1, & D(q_3) = \downarrow, \\
\varphi(q_0,b,2,q_0,22) = 1, & \varphi(q_0,\$,2,q_3,2) = 1, & \varphi(q_1,a,Z,q_0,Z) = 1, \\
\varphi(q_1,b,Z,q_0,Z) = 1, & \varphi(q_1,\$,\tau,q_1,\tau) = 1, & \varphi(q_1,a,1,q_3,12) = 1, \\
\varphi(q_1,b,1,q_0,1) = 1, & \varphi(q_1,a,2,q_0,2) = 1, & \varphi(q_1,b,2,q_3,21) = 1, \\
\varphi(q_2,a,Z,q_3,Z2) = 1, & \varphi(q_2,b,Z,q_3,Z1) = 1, & \varphi(q_2,\$,Z,q_0,Z) = 1, \\
\varphi(q_2,a,1,q_2,\varepsilon) = 1, & \varphi(q_2,b,1,q_0,12) = 1, & \varphi(q_2,\$,1,q_0,1) = 1, \\
\varphi(q_2,a,2,q_0,21) = 1, & \varphi(q_2,b,2,q_2,\varepsilon) = 1, & \varphi(q_2,\$,2,q_0,2) = 1, \\
\forall \sigma \in \{a,b,\$\} & \varphi(q_3,\sigma,Z,q_3,Z) = 1, & \\
\varphi(q_3,a,1,q_3,1) = 1, & \varphi(q_3,b,1,q_3,11) = 1, & \varphi(q_3,\$,1,q_2,1) = 1, \\
\varphi(q_3,a,2,q_3,22) = 1, & \varphi(q_3,b,2,q_3,2) = 1, & \varphi(q_3,\$,2,q_2,2) = 1,
\end{array}$$

other values of arguments yield 0.

□

It is doubtful whether language L_3 can be recognized with probability 1 by QPA with stack alphabet T containing only one symbol, i.e, by quantum finite one counter automata [Kr 99].

Let us consider language which is not recognizable by any deterministic pushdown automaton:

Theorem 3.1. *Language $L_4 = \{\omega \in (a,b,c)^* |\ |\omega|_a = |\omega|_b = |\omega|_c\}$ is recognizable by a QPA with probability $\frac{2}{3}$.*

Proof. Sketch of proof. The automaton takes three equiprobable actions, during the first action it compares $|\omega|_a$ to $|\omega|_b$, whereas during the second action $|\omega|_b$ to $|\omega|_c$ is compared. Input word is rejected if the third action is chosen. Acceptance probability totals $\frac{2}{3}$.

□

Theorem 3.2. *Language* $L_5 = \{\omega \in (a,b,c)^* |\ |\omega|_a = |\omega|_b \ xor\ |\omega|_a = |\omega|_c\}$ *is recognizable by a QPA with probability* $\frac{4}{7}$.

Proof. Sketch of proof. The automaton starts the following actions with the following amplitudes:

a) with an amplitude $\sqrt{\frac{2}{7}}$ compares $|\omega|_a$ to $|\omega|_b$.

b) with an amplitude $-\sqrt{\frac{2}{7}}$ compares $|\omega|_a$ to $|\omega|_c$.

c) with an amplitude $\sqrt{\frac{3}{7}}$ accepts the input. If exactly one comparison gives positive answer, input is accepted with probability $\frac{4}{7}$. If both comparisons gives positive answer, amplitudes, which are chosen to be opposite, annihilate and the input is accepted with probability $\frac{3}{7}$.

□

Language L_5 cannot be recognized by deterministic pushdown automata. It even seems that this language is not recognizable by probabilistic pushdown automata either. In this case this result would be similar to that of [AFGK 99], where the properties of quantum finite multitape automata are considered.

References

[AFGK 99] Andris Ambainis, Rūsiņš Freivalds, Marats Golovkins, Marek Karpinski. Quantum finite multitape automata. *Lecture Notes in Computer Science*, 1999, Vol. 1725, pp. 340-348.

[BV 97] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26:1411-1473, 1997.

[CRSS 98] A. Robert Calderbank, Eric M. Rains, Peter W. Shor, Neil J. A. Sloane. Quantum error correction via codes over GF(4). *IEEE Transactions on Information Theory*, 1998, vol. 44, p. 1369-1387.

[De 85] David Deutsch. Quantum Theory, the Church-Turing principle and the universal quantum computer. *Proc. Royal Society London, A400*, 1985. p. 96-117.

[Fe 82] Richard Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 1982, vol. 21, No 6/7, p. 467-488.

[Gu 89] E. Gurari. An introduction to the theory of computation. *Computer Science Press*, 1989.

[KLMT 99] E. Knill, R. Laflamme, R. Martinez, C.-H. Tseng. A cat-state benchmark on a seven bit quantum computer. http://xxx.lanl.gov/abs/quant-ph/9908051

[KW 97] Attila Kondacs and John Watrous. On the power of quantum finite state automata. In *Proc. 38th FOCS*, 1997, p. 66–75.

[Kr 99] Maksim Kravtsev. Quantum finite one-counter automata. *Lecture Notes in Computer Science*, 1999, Vol. 1725, pp. 431-440.

[MC 97] Christopher Moore, James P. Crutchfield. Quantum automata and quantum grammars. http://xxx.lanl.gov/abs/quant-ph/9707031

[Sh 94] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. *Proc. 35th FOCS*, 1994, p. 124-134.

Richard Bonner and Rūsiņš Freivalds (Eds.)

Quantum Computation and Learning

International Workshop
Riga, Latvia, 11-13 September 1999

Proceedings

Institute of Mathematics and Computer Science
University of Latvia, Riga, Latvia

Department of Mathematics and Physics
Mälardalen University, Västerås, Sweden

Workshop sponsored by The Swedish Institute

Contents

Quantum Computation

Learning

Quantum Finite Multitape Automata vs Probabilistic Finite Multitape Automata

Andris Ambainis,[1] Richard Bonner,[2] Rūsiņš Freivalds,[3] Marats Golovkins,[3] and Marek Karpinski[4]

[1] Computer Science Division, University of California, Berkeley, CA 94720-2320[†]
[2] Department of Mathematics and Physics, Mälardalens University
[3] Institute of Mathematics and Computer Science, University of Latvia, Raiņa bulv. 29, Riga, Latvia[§]
[4] Department of Computer Science, University of Bonn, 53117, Bonn, Germany[¶]
ambainis@cs.berkeley.edu, richard.bonner@mdh.se, rusins@cclu.lv, marats@cclu.lv, marek@cs.bonn.edu

Abstract. Quantum finite automata were introduced by C. Moore, J. P. Crutchfield [4], and by A. Kondacs and J. Watrous [3]. This notion is not a generalization of the deterministic finite automata. Moreover, in [3] it was proved that not all regular languages can be recognized by quantum finite automata. A. Ambainis and R. Freivalds [1] proved that for some languages quantum finite automata may be exponentially more concise rather than both deterministic and probabilistic finite automata. In this paper we introduce the notion of quantum finite multitape automata and prove that there is a language recognized by a quantum finite automaton but not by deterministic or probabilistic finite automata. This is the first result on a problem which can be solved by a quantum computer but not by a deterministic or probabilistic computer. Additionally we discover unexpected probabilistic automata recognizing complicated languages.

1 Introduction

The basic model, i.e., quantum finite automata (QFA), were introduced twice. First this was done by C. Moore and J.P.Crutchfield [4]. Later in a different and non-equivalent way these automata were introduced by A. Kondacs and J. Watrous [3].

The first definition just mimics the definition of 1-way finite probabilistic only substituting *stochastic* matrices by *unitary* ones. To define quantum finite multitape automata we generalize a more elaborated definition [3].

We are using these notations in the following definition:

[†] Supported by Berkeley Fellowship for Graduate Studies.
[§] Research supported by Grant No.96.0282 from the Latvian Council of Science
[¶] Research partially supported by the International Computer Science Institute, Berkeley, California, by the DFG grant KA 673/4-1, and by the ESPRIT BR Grants 7079 and ECUS030

z^* is the complex conjugate of a complex number z.
$M =_{def} \{1, 2, \ldots, m\}$.
The k-th component of an arbitrary vector s will be defined as s^k.
We shall understand by I an arbitrary element from the set $P(M) \setminus \{\emptyset\}$.

$R_I =_{def} A_1 \times A_2 \times \ldots \times A_m$, where $A_i = \begin{cases} \{\downarrow, \rightarrow\}, & if\ i \notin I \\ \{\text{"}nothing\text{"}\}, & if\ i \in I. \end{cases}$

$T_I =_{def} B_1 \times B_2 \times \ldots \times B_m$, where $B_i = \begin{cases} \{\downarrow, \rightarrow\}, & if\ i \in I \\ \{\text{"}nothing\text{"}\}, & if\ i \notin I. \end{cases}$

The function $R_i \times T_i \xrightarrow{d_I} \{\downarrow, \rightarrow\}^m$ is defined as follows:

$d_I(r, t) =_{def} (d_I^1(r, t), d_I^2(r, t), \ldots, d_I^m(r, t))$, where $d_I^i(r, t) = \begin{cases} r^i, & if\ i \notin I \\ t^i, & if\ i \in I. \end{cases}$

Definition 1. A quantum finite multitape automaton (QFMA) $A = (Q; \Sigma; \delta; q_0; Q_a; Q_r)$ is specified by the finite input alphabet Σ, the finite set of states Q, the initial state $q_0 \in Q$, the sets $Q_a \subset Q$, $Q_r \subset Q$ of accepting and rejecting states, respectively, with $Q_a \cap Q_r = \emptyset$, and the transition function

$$\delta : Q \times \Gamma^m \times Q \times \{\downarrow, \rightarrow\}^m \longrightarrow \mathbb{C}_{[0,1]},$$

where m is the number of input tapes, $\Gamma = \Sigma \cup \{\#, \$\}$ is the tape alphabet of A and $\#, \$$ are end-markers not in Σ, which satisfies the following conditions (of well-formedness):

1. Local probability condition

$$\forall (q_1, \sigma) \in Q \times \Gamma^m \sum_{(q, d) \in Q \times \{\downarrow, \rightarrow\}^m} |\delta(q_1, \sigma, q, d)|^2 = 1.$$

2. Orthogonality of column vectors condition.

$$\forall q_1, q_2 \in Q, q_1 \neq q_2, \forall \sigma \in \Gamma^m \sum_{(q, d) \in Q \times \{\downarrow, \rightarrow\}^m} \delta^*(q_1, \sigma, q, d)\delta(q_2, \sigma, q, d) = 0.$$

3. Separability condition.

$$\forall I \in P(M) \setminus \{\emptyset\}\ \forall q_1, q_2 \in Q$$

$$\forall \sigma_1, \sigma_2 \in \Gamma^m,\ where\ \forall i \notin I\ \sigma_1^i = \sigma_2^i$$

$$\forall t_1, t_2 \in T_I,\ where\ \forall j \in I\ t_1^j \neq t_2^j$$

$$\sum_{(q, r) \in Q \times R_I} \delta^*(q_1, \sigma_1, q, d_I(r, t_1))\delta(q_2, \sigma_2, q, d_I(r, t_2)) = 0.$$

States from $Q_a \cup Q_r$ are called halting states and states from $Q_{non} = Q \setminus (Q_a \cup Q_r)$ are called non-halting states.

To process an input word vector $x \in (\Sigma^*)^m$ by A it is assumed that the input is written on every tape k with the end-markers in the form $w_x^k = \#x^k\$$ and that every such a tape, of length $|x^k| + 2$, is circular, i. e., the symbol to the right of $\$$ is $\#$.

For the fixed input word vector x we can define $n \in \mathbb{N}^m$ to be an integer vector which determines the length of input word on every tape. So for every n we can define C_n to be the set of all possible configurations of A where $|x^i| = n^i$. $|C_n| = |Q| \prod_{i=1}^{m} (n^i + 2)$. Every such a configuration is uniquely determined by a pair $|q, s\rangle$, where $q \in Q$ and $0 \leq s^i \leq |x^i| + 1$ specifies the position of head on the i-th tape.

Every computation of A on an input x, $|x^i| = n^i$, is specified by a unitary evolution in the Hilbert space $H_{A,n} = l_2(C_n)$. Each configuration $c \in C_n$ corresponds to the basis vector in $H_{A,n}$. Therefore a global state of A in the space $H_{A,n}$ has a form $\sum_{c \in C_n} \alpha_c |c\rangle$, where $\sum_{c \in C_n} |\alpha_c|^2 = 1$. If the input word vector is x and the automaton A is in its global state $|\psi\rangle = \sum_{c \in C_n} \alpha_c |c\rangle$, then its further step is equivalent to the application of a linear operator U_x^δ over Hilbert space $l_2(C_n)$.

Definition 2. The linear operator U_x^δ is defined as follows:

$$U_x^\delta |\psi\rangle = \sum_{c \in C_n} \alpha_c U_x^\delta |c\rangle.$$

If a configuration $c = |q', s\rangle$, then

$$U_x^\delta |c\rangle = \sum_{(q,d) \in Q \times \{\downarrow, \rightarrow\}^m} \delta(q', \sigma(s), q, d) |q, \tau(s, d)\rangle,$$

where $\sigma(s) = (\sigma^1(s), \ldots, \sigma^m(s))$, $\sigma^i(s)$ specifies the s^i-th symbol on the i-th tape, and $\tau(s, d) = (\tau^1(s, d), \ldots, \tau^m(s, d))$,

$$\tau^i(s, d) = \begin{cases} (s^i + 1) \bmod (n^i + 2), & \text{if } d^i = '\rightarrow' \\ s^i, & \text{if } d^i = '\downarrow'. \end{cases}$$

Lemma 3. *The well-formedness conditions are satisfied iff for any input x the mapping U_x^δ is unitary.*

Language recognition for QFMA is defined as follows. For each input x with the corresponding vector n, $n^i = |x^i|$, and a QFMA $A = (Q; \Sigma; \delta; q_0; Q_a; Q_r)$ we define $C_n^a = \{(q, s) | (q, s) \in C_n, q \in Q_a\}$, $C_n^r = \{(q, s) | (q, s) \in C_n, q \in Q_r\}$, $C_n^{non} = C_n \setminus (C_n^a \cup C_n^r)$. E_a, E_r, E_{non} are the subspaces of $l_2(C_n)$ spanned by C_n^a, C_n^r, C_n^{non} respectively. We use the observable O that corresponds to the orthogonal decomposition $l_2(C_n) = E_a \oplus E_r \oplus E_{non}$. The outcome of each observation is either "accept" or "reject" or "non-halting".

The language recognition is now defined as follows: For an $x \in (\Sigma^*)^m$ we consider as the input ω_x, $\omega_x^k = \#x^k\$$, and assume that the computation starts with A being in the configuration $|q_0, \{0\}^m\rangle$. Each computation step consists of two parts. At first the linear operator $U_{\omega_x}^\delta$ is applied to the current global state and then the resulting superposition, i.e., global state, is observed using the observable O as defined above. If the global state before the observation

is $\sum_{c \in C_a} \alpha_c |c\rangle$, then the probability that the subspace E_i, $i \in \{a, r, non\}$, will be chosen is $\sum_{c \in C_a^i} |\alpha_c|^2$. The computation continues until the result of an observation is "accept" or "reject".

Definition 4. A QFMA $A = (Q; \Sigma; \delta; q_0; Q_a; Q_r)$ is simple if for each $\sigma \in \Gamma^m$ there is a linear unitary operator V_σ over the inner-product space $l_2(Q)$ and a function $D : Q \longrightarrow \{\downarrow, \rightarrow\}^m$, such that

$$\forall q_1 \in Q \; \forall \sigma \in \Gamma^m \quad \delta(q_1, \sigma, q, d) = \begin{cases} \langle q|V_\sigma|q_1\rangle, & \text{if } D(q) = d \\ 0, & \text{otherwise.} \end{cases}$$

Lemma 5. *If the automaton A is simple, then conditions of well-formedness are satisfied iff for every σ V_σ is unitary.*

We shall deal only with simple multitape automata further in the paper.

2 Quantum vs. probabilistic automata

Definition 6. We shall say that an automaton is deterministic reversible finite multitape automaton (RFMA), if it is a simple QFMA with $\delta(q_1, \sigma, q, d) \in \{0, 1\}$.

Definition 7. We say that a language L is $[m, n]$-deterministically recognizable if there are n deterministic automata A_1, A_2, A_n such that:
a) if the input is in the language L, then all n automata $A_1, \ldots, A_n$ accept the input;
b) if the input is not in the language L, then at most m of the automata $A_1, \ldots, A_n$ accept the input.

Definition 8. We say that a language L is $[m, n]$-reversibly recognizable if there are n deterministic reversible automata A_1, A_2, A_n such that:
a) if the input is in the language L, then all n automata $A_1, \ldots, A_n$ accept the input;
b) if the input is not in the language L, then at most m of the automata $A_1, \ldots, A_n$ accept the input.

Lemma 9. *If a language L is $[1, n]$-deterministically recognizable by 2-tape finite automata, then L is recognizable by a probabilistic 2-tape finite automaton with probability $\frac{n}{n+1}$.*

Proof. The probabilistic automaton starts by choosing a random integer $1 \leq r \leq (n + 1)$. After that , if $r \leq n$, then the automaton goes on simulating the deterministic automaton A_r, and, if $r = n + 1$, then the automaton rejects the input. The inputs in L are accepted with probability $\frac{n}{n+1}$, and the inputs not in the language are rejected with a probability no less than $\frac{n}{n+1}$. □

Lemma 10. *If a language L is $[1,n]$-reversibly recognizable by 2-tape finite automata, then L is recognizable by a quantum 2-tape finite automaton with probability $\frac{n}{n+1}$.*

Proof. In essence the algorithm is the same as in Lemma 9. The automaton starts by taking $n+1$ different actions with amplitudes $\frac{1}{\sqrt{n+1}}$. (It is possible to construct a unitary matrix to make such a choice feasible.) After that the automaton simultaneously goes on simulating all the deterministic reversible automata A_r, $1 \leq r \leq (n+1)$, where the automaton A_{n+1} rejects an input. The simulation of each deterministic reversible automaton uses its own accepting and rejecting states. (Hence the probabilities are totaled, not the amplitudes.) □

First, we discuss the following 2-tape language

$$L_1 = \{(x_1 \nabla x_2, y) || x_1 = x_2 = y\},$$

where the words x_1, x_2, y are unary.

Lemma 11. *(Proved by R. Freivalds [2].) For arbitrary natural n, the language L_1 is $[1,n]$-deterministically recognizable.*

Lemma 12. *For arbitrary natural n, the language L_1 is $[1,n]$-reversibly recognizable.*

Proof. By Lemma 11, the language L_1 is $[1,n]$-deterministically recognizable. However it is easy enough to make the construction of the automata $A_1, \ldots, A_n$ in the following manner:
a) every automaton is reversible;
b) if a word pair is in the language L_1, then every automaton consumes the same number of steps to accept the word pair.
The last requirement will be essential further in the paper.
If at least the first requirement is met, then the language is $[1,n]$-reversibly recognizable. □

Theorem 13. *The language L_1 can be recognized with arbitrary probability $1-\epsilon$ by a probabilistic 2-tape finite automaton but this language cannot be recognized by a deterministic 2-tape finite automaton.*

Theorem 14. *The language L_1 can be recognized with arbitrary probability $1-\epsilon$ by a quantum 2-tape finite automaton.*

Proof. By Lemmas 10 and 12. □

In an attempt to construct a 2-tape language recognizable by a quantum 2-tape finite automaton but not by probabilistic 2-tape finite automata we consider a similar language

$$L_2 = \{(x_1 \nabla x_2 \nabla x_3, y) || \; there are exactly 2 values of \; x_1, x_2, x_3 such that they equal \; y\},$$

where the words x_1, x_2, x_3, y are unary.

Theorem 15. *A quantum automaton exists which recognises the language L_2 with a probability $\frac{9}{16} - \epsilon$ for arbitrary positive ϵ.*

Proof. This automaton takes the following actions with the following amplitudes:

a) $\frac{\sqrt{3}}{4} \cdot 1$ – compares $x_1 = x_2 = y$,
b) $\frac{\sqrt{3}}{4} \cdot (\cos\frac{2\pi}{3} + i\sin\frac{2\pi}{3})$ – compares $x_2 = x_3 = y$,
c) $\frac{\sqrt{3}}{4} \cdot (\cos\frac{4\pi}{3} + i\sin\frac{4\pi}{3})$ – compares $x_1 = x_3 = y$,
d) $\frac{\sqrt{7}}{4}$ – says "accept".

By Theorem 14 comparison in actions a), b), c) can be accomplished. By construction in Lemma 10 the comparison in each action a), b), c) is implemented by starting $n+1$ different branches. Therefore in any action i), $i \in \{a, b, c\}$, if a comparison is successful, the automaton will come respectively into non-halting states $q_{a,1}, \ldots, q_{a,n}$, $q_{b,1}, \ldots, q_{b,n}$, $q_{c,1}, \ldots, q_{c,n}$, reaching the symbol pair (\$, \$) on the tapes. The transition (\$, \$) for every $k = 1, \ldots, n$ is as follows:

	$q_{a,k}$	$q_{b,k}$	$q_{c,k}$
$q_{a1,k}$	$\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}$
$q_{r,k}$	$\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}(\cos\frac{4\pi}{3} + i\sin\frac{4\pi}{3})$	$\frac{1}{\sqrt{3}}(\cos\frac{2\pi}{3} + i\sin\frac{2\pi}{3})$
$q_{a2,k}$	$\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}(\cos\frac{2\pi}{3} + i\sin\frac{2\pi}{3})$	$\frac{1}{\sqrt{3}}(\cos\frac{4\pi}{3} + i\sin\frac{4\pi}{3})$

Here $q_{a1,k}, q_{a2,k}$ are accepting states and $q_{r,k}$ are rejecting states. If y equals all 3 words x_1, x_2, x_3, then it is possible to ensure that it takes the same time to reach the end-marking symbol pair in every action on every branch. Therefore the input is accepted with probability $\frac{7}{16} + \epsilon$ (since the amplitudes of the actions a), b), c) total to 0). If y equals 2 out of 3 words x_1, x_2, x_3, then the input is accepted with probability $\frac{9}{16} - \epsilon$. If y equals at most one of the words x_1, x_2, x_3, then the input is accepted with probability $\frac{7}{16} + \epsilon$ (only if the action d) is taken). □

Unfortunately, the following theorem holds.

Theorem 16. *A probabilistic automaton exists which recognizes the language L_2 with a probability $\frac{21}{40}$.*

Proof. The probabilistic automaton with probability $\frac{1}{2}$ takes an action A or B:
A) Choose a random j and compare $x_j = y$. If yes, accept with probability $\frac{19}{20}$. If no, accept with probability $\frac{1}{20}$.
B) Choose a random pair j, k and compare $x_j = x_k = y$. If yes, reject. If no, accept with probability $\frac{12}{20}$.

If y equals all 3 words x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability $\frac{19}{20}$. If y equals all 3 words x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability 0. This gives the acceptance probability in the case if y equals all 3 words x_1, x_2, x_3, to be $\frac{19}{40}$ and the probability of the correct result "no" to be $\frac{21}{40}$.

If y equals 2 words out of x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability $\frac{13}{20}$. If y equals 2 words out of x_1, x_2, x_3

and the action B is taken, then the input is accepted with relative probability $\frac{8}{20}$. This gives the acceptance probability in the case if y equals 2 words out of x_1, x_2, x_3, to be $\frac{21}{40}$.

If y equals only 1 word out of x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability $\frac{7}{20}$. If y equals only 1 word out of x_1, x_2, x_3 and the action B is taken, then the input is accepted with relative probability $\frac{12}{20}$. This gives the acceptance probability in the case if y equals only 1 word out of x_1, x_2, x_3, to be $\frac{19}{40}$ and the probability of the correct result "no" to be $\frac{21}{40}$.

If y equals no word of x_1, x_2, x_3 and the action A is taken, then the input is accepted with relative probability $\frac{1}{20}$. If y equals no word of x_1, x_2, x_3 and the action B is taken, then the input is accepted with relative probability $\frac{12}{20}$. This gives the acceptance probability in the case if y equals no word of x_1, x_2, x_3, to be $\frac{13}{40}$ and the probability of the correct result "no" to be $\frac{27}{40}$. □

Now we consider a modification of the language L_2 which might be more difficult for a probabilistic recognition:

$$L_3 = \{(x_1 \nabla x_2 \nabla x_3, y_1 \nabla y_2) \| \text{there is exactly one value } k \text{ such that there are exactly two values } j \text{ such that } x_j = y_k.\}$$

Theorem 17. *A quantum finite 2-tape automaton exists which recognizes the language L_3 with a probability $\frac{36}{67} - \epsilon$ for arbitrary positive ϵ.*

However this language also can be recognized by a probabilistic 2-tape finite automaton.

Theorem 18. *A probabilistic finite 2-tape automaton exists which recognizes the language L_3 with a probability $\frac{13}{25} - \epsilon$ for arbitrary positive ϵ.*

Proof. The probabilistic automaton with probability $\frac{6}{25}$ takes action A or B or C or with probability $\frac{7}{25}$ takes action D:
A) Choose a random k and two values of j. Then compare $x_j = y_k$. If yes, accept. If no, reject.
B) Chose a random k and compare $x_1 = x_2 = x_3 = y_k$. If yes, reject. If no, accept. C) Choose two values j and m. Then compare $x_j = x_m = y_1 = y_2$. If yes, reject. If no, accept.
D) Says "reject".

Notice that the actions A, B, C are probabilistic, and they can be performed only with probability $1 - \epsilon$ (actions A and B are described in the proof of Theorem 13 and action C is similar).

The acceptance probabilities equal:

	A	B	C	total
no y_k equals 2 or 3 x_j	0	1	1	$\frac{12}{25}$
one y_k equals 2 x_j	$\frac{1}{6}$	1	1	$\frac{13}{25}$
one y_k equals 3 x_j	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{12}{25}$
two y_k equal 2 x_j	$\frac{1}{3}$	1	$\frac{2}{3}$	$\frac{12}{25}$
all y_k equal all x_j	1	0	0	$\frac{6}{25}$

□

Finally we consider a modification of the languages above which recognition indeed is impossible by probabilistic automata:

$$L_4 = \{(x_1 \nabla x_2, y) || \text{there is exactly one value } j \text{ such that } x_j = y.\}$$

where the words x_1, x_2, y are binary.

Theorem 19. *A quantum finite 2-tape automaton exists which recognizes the language L_4 with a probability $\frac{4}{7}$.*

Proof. The automaton has two accepting q_{a1}, q_{a2} and three rejecting states q_{r1}, q_{r2}, q_{r3} and starts the following actions by reading the pair $(\#, \#)$ with the following amplitudes:

a) with an amplitude $\sqrt{\frac{2}{7}}$ compares x_1 to y,

b) with an amplitude $-\sqrt{\frac{2}{7}}$ compares x_2 to y,

c) with an amplitude $\sqrt{\frac{3}{7}}$ immediately goes to the state q_{a1}.

Actions a) and b) use different non-halting states to process the word pair. All these actions the automaton processes simultaneously. In actions a) and b), if no (not equal), it goes accordingly to the states q_{r1} or q_{r2}, if yes, then reaches correspondent non-halting states q_α or q_β, while the symbol pair on the tapes is $(\$, \$)$. The transition for $(\$, \$)$ and states $q_\alpha, q_\beta, q_{a2}, q_{r3}$ is as follows:

	q_α	q_β
q_{a2}	$\frac{1}{\sqrt{2}}$	$\frac{1}{\sqrt{2}}$
q_{r3}	$\frac{1}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}$

If all the words are equal, it is possible to ensure that it takes the same time to reach the end-markers on both tapes, therefore the automaton reaches the superposition $\sqrt{\frac{2}{7}}|q_\alpha, s, t\rangle - \sqrt{\frac{2}{7}}|q_\beta, s, t\rangle$, where s and t specify the place of \$ on each tape, and the input is accepted with probability $\frac{3}{7}$. (Since the amplitudes of the actions a) and b) equal to 0.) If one of the words x_i equals y, then the input is accepted with probability $\frac{4}{7}$. If none of the words x_i equals y, then the input is accepted with probability $\frac{3}{7}$. □

References

1. Andris Ambainis and Rūsiņš Freivalds. 1-way quantum finite automata: strengths, weaknesses and generalizations. *Proc. 39th FOCS*, 1998, pp. 332–341. http://xxx.lanl.gov/abs/quant-ph/9802062
2. Rūsiņš Freivalds. Fast probabilistic algorithms. *Lecture Notes in Computer Science*, 1979, vol. 74, pp. 57–69.
3. Attila Kondacs and John Watrous. On the power of quantum finite state automata. In *Proc. 38th FOCS*, 1997, pp. 66–75.
4. Christopher Moore, James P. Crutchfield Quantum automata and quantum grammars. http://xxx.lanl.gov/abs/quant-ph/9707031

Projections of multi-tape languages recognized by quantum and probabilistic finite automata

Richard Bonner[1] , Rūsiņš Freivalds[2] and Marats Golovkins[2]

[1] Department of Mathematics and Physics, Mälardalens University
[2] Institute of Mathematics and Computer Science, University of Latvia, Raiņa bulv. 29, Riga, Latvia***
richard.bonner@mdh.se, rusins@cclu.lv, marats@cclu.lv

Abstract. A language $L^{(n)}$ of n-tuples of words which is recognized by a n-tape rational finite-probabilistic automaton with probability $1-\epsilon$, for arbitrary $\epsilon > 0$, is called quasideterministic. It is proved in [Fr 81], that each rational stochastic language is a projection of a quasideterministic language $L^{(n)}$ of n-tuples of words. Had projections of quasideterministic languages on one tape always been rational stochastic languages, we would have a good characterization of the class of the rational stochastic languages. However the opposite is proved in [BFLL98]. A two-tape quasideterministic language exists, the projection of which on the first tape is a nonstochastic language. We prove in this paper that the projection in [BFLL98] cannot be recognized by quantum automata (even with unbounded error) either. This raises a question of a complete counterpart of the results in [BFLL98] where one substitutes "quantum" for "probabilistic".

1 Introduction

Let N denote the set of all natural numbers. Let $n \in N$. By σ^n we denote a string consisting of n symbols σ. If Σ is a set, then Σ^n stands for the set of all the n-element strings over the alphabet Σ. A finite probabilistic automaton (FPA) is a system $\omega = (\Sigma, S, \Pi_0, M_\Sigma, F)$, where $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_c\}$ is a finite input alphabet, $S = \{s_1, s_2, \ldots, s_m\}$ is a finite set of states, $\Pi_0 = (p_1, p_2, \ldots, p_m)$ is a stochastic vector (the initial distribution of the probabilities of the states; $p_1 + p_2 + \ldots + p_m = 1$), M_Σ is a system of stochastic $m \times m$-matrices M_{σ_1}, $M_{\sigma_2}, \ldots, M_{\sigma_c}$ (the matrices of the probabilities for the transition from one state to another under the influence of the corresponding input symbol), and $F \subset S$ is a set of accepting states. Let $\eta_F = (\eta_1, \eta_2, \ldots, \eta_m)^T$ be a column matrix defined by $\eta_j = 1$ if $s_j \in F$ and $\eta_j = 0$ otherwise.

We say that a language L over the alphabet Σ is *acceptable* with cut-point γ $(0 \leq \gamma < 1)$ by an automaton ω if the words $x_1 x_2 \ldots x_n$ in L are exactly the strings for which $\Pi_0 M_{x_1} M_{x_2} \ldots M_{x_n} \eta_F > \gamma$. In other words, we represent a language L in FPA ω with cut-point γ. For an arbitrary word x of L, if ω

*** Research supported by Grant No.96.0282 from the Latvian Council of Science

starts to work on x in a random state s_j distributed according to Π_0 then it stops in an accepting state with probability strictly larger than γ. A language L is called *stochastic* if it can be represented in some FPA with some cut-point γ $(0 \leq \gamma < 1)$. A FPA is called *rational* if all the components of its initial probability distribution and all elements its transition probability matrices are rational numbers. A language L is called *rational stochastic* if it can be represented in a rational FPA with a rational cut-point.

A FPA is called a finite *deterministic* automaton (FDA) if all the components of its initial distribution and all elements of its transition probability matrices are numbers from the set $\{0, 1\}$. A language L represented in a FDA is called *regular*.

We will consider in detail the case when a language L is represented in a FPA with cut-point $\gamma \geq \frac{1}{2}$ and so that $\Pi_0 M_{x_1} M_{x_2} \dots M_{x_n} \eta_F \leq 1 - \gamma$ for all strings $x_1 x_2 \dots x_n$ *not* in L. In this case we say that the FPA *recognizes* the language L with probability γ.

Rabin and Scott [RS 59] introduced the concept of a multi-tape FDA, that is, a FDA that processes not words but tuples of words. Such an automaton has n tapes, over each of which a separate head can move in one direction, at most one unit at a time. Input words are written on tapes and every head observes a letter on the tape directly under it. It is presumed that the automaton can recognize the end of a word. This is provided by including in the alphabet Σ a special symbol # which is put on every tape immediately after an input word. So, the automaton is used to recognize sets of words in the alphabet $\Sigma \setminus \{\#\}$. It is assumed that when some head reaches a symbol #, further movement of this head becomes impossible. The work of the automaton ends when all the heads have observed the symbol #. We shall consider that the automaton has *accepted* the given n-tuple of words, if at this moment the automaton transits to an accepting state; otherwise, we consider the automaton to have *rejected* the input.

To formalise the definitions, let $n \in N$, denote by $W(n)$ the set of all the subsets of the set $\{1, 2, \dots, n\}$, and let U_m denote the set of all m-dimensional stochastic vectors.

A *deterministic* n-tape finite automaton (n-FDA) is then a system

$$\omega = (\Sigma, S, s_1, \delta, \lambda, F)$$

where $\Sigma = \{\sigma_1, \sigma_2, \dots, \sigma_c\}$ is a finite input alphabet containing the symbol #, $S = \{s_1, s_2, \dots, s_m\}$ is a finite set of states with a singled out subset $F \subset S$ of accepting states and an initial state $s_1 \in S$, $\delta : S \times \Sigma^n \to S$ is a transition function from one state to another, and $\lambda : S \times \Sigma^n \to W(n)$ is a head movement function satisfying $i \notin \lambda(s_j, x_1, \dots, x_n)$ if $x_i = \#$, $i = 1, \dots, n$, $j = 1, \dots, m$.

A *probabilistic* n-tape finite automaton (n-FPA) is a system

$$\omega = (\Sigma, S, \Pi_0, \delta, \lambda, F)$$

with Σ, S, F defined as above, and where $\Pi_0 = (p_1, p_2, \dots, p_m) \in U_m$ is an initial probability distribution of states, $\delta : S \times \Sigma^n \to U_m$ is a state transition

probability function, and $\lambda : S \times \Sigma^n \to U_{2^n}$ is a head movement function prescribing probabilities of subsets of tapes (points in $W(n)$) whereby subsets containing a tape in state # receive probability zero.

Let $y = (y_1, y_2, \ldots, y_n)$ be a n-tuple of strings over $\Sigma \setminus \{\#\}$. Denote by $P_\omega(y)$ the probability that n-FPA ω operates on y with initial distribution of the states Π_0 and stops operation in a state from the set F. We shall say that n-FPA ω *recognizes* a language $L = L^{(n)}$ of n-tuples of words over $\Sigma \setminus \{\#\}$ with probability γ ($\frac{1}{2} \leq \gamma < 1$), if for any n-tuple y of strings over $\Sigma \setminus \{\#\}$, we have $P_\omega(y) > \gamma$ if $y \in L$ while $P_\omega(y) \leq 1 - \gamma$ if $y \notin L$.

It was proved in [Fr 78] that there exists a language of pairs of words which cannot be recognized by any 2-tape FDA and cannot even be accepted by any 2-tape FNA; this language can however be recognized by a 2-tape FPA with probability $1 - \epsilon$, for arbitrary $\epsilon > 0$. It was proved in [Fr 91] that the class of languages that can be recognized by 2-tape FPA with probability $1 - \epsilon$ for arbitrary $\epsilon > 0$ is rather complex: in this class the emptiness problem is not decidable. In the present paper we characterize the complexity of this class in terms of projection languages. Here, the *projection* onto the first tape of a language $L = L^{(n)}$ of n-tuples of words $y = (y_1, y_2, \ldots, y_n)$ is defined as the language consisting of the words y_1 as y ranges over L.

We call a language L of n-tuples of word *quasideterministic* if for arbitrary $\epsilon > 0$ there exists an n-FPA which recognizes L with probability $1 - \epsilon$.

We consider 1-way quantum finite automata (QFA) as defined in [KW 97]. Namely, 1-way QFA is a tuple $M = (Q, \Sigma, \delta, q_0, Q_{acc}, Q_{rej})$ where Q is a finite set of states, Σ is an input alphabet, δ is a transition function, $q_0 \in Q$ is a starting state and $Q_{acc} \subset Q$ and $Q_{rej} \subset Q$ are sets of accepting and rejecting states. The states in Q_{acc} and Q_{rej} are called *halting states* and the states in $Q_{non} = Q - (Q_{acc} \cup Q_{rej})$ are called *non-halting states*. # and \$ are symbols that do not belong to Σ. We use # and \$ as left and right endmarker, respectively. The *working alphabet* of M is $\Gamma = \Sigma \cup \{\#, \$\}$.

A superposition of M is any element of $l_2(Q)$ (the space of mappings from Q to $\mathbb{C}$). For $q \in Q$, $|q\rangle$ denote the unit vector with value 1 at q and 0 elsewhere. All elements of $l_2(Q)$ can be expressed as linear combinations of vectors $|q\rangle$. We shall use ψ to denote elements of $l_2(Q)$.

The transition function δ maps $Q \times \Gamma \times Q$ to $\mathbb{C}$. The value $\delta(q_1, a, q_2)$ is an amplitude of $|q_2\rangle$ in the superposition of states to which M goes from $|q_1\rangle$ after reading a. For $a \in \Gamma$, V_a is a linear transformation on $l_2(Q)$ such that

$$V_a(|q_1\rangle) = \sum_{q_2 \in Q} \delta(q_1, a, q_2)|q_2\rangle.$$

We require all V_a to be unitary.

The computation of a QFA starts in the superposition $|q_0\rangle$. Then transformations corresponding to the left endmarker #, the letters of the input word x and the right endmarker \$ are applied. A transformation corresponding to $a \in \Gamma$ consists of two steps.

1. First, V_a is applied. The new superposition ψ' is $V_a(\psi)$ where ψ is the superposition
2. Then, ψ' is observed with respect to the observable $E_{acc} \oplus E_{rej} \oplus E_{non}$ where $E_{acc} = span\{|q\rangle : q \in Q_{acc}\}$, $E_{rej} = span\{|q\rangle : q \in Q_{rej}\}$, $E_{non} = span\{|q\rangle : q \in Q_{non}\}$. This observation gives $x \in E_i$ with probability equal to the amplitude of the projection of psi'. After that, the superposition collapses to this projection.
   If we get $\psi' \in E_{acc}$, the input is accepted. If $\psi' \in E_{rej}$, the input is rejected. If $\psi' \in E_{non}$, the next transformation is applied.

We regard these two transformations as reading a letter a. V'_a is the transformation that maps ψ to the non-halting part of $V_a(\psi)$. $V'_a = P_{non} V_a$ where $P_{non}(\psi)$ is a linear transformation which leaves all non-halting components of the configuration ψ unchanged and maps all accepting and rejecting components to 0. If x is a word consisting of letters $a_1 \dots a_k$, then V_x denotes $V_{a_k} \dots V_{a_1}$ and V'_x denotes $V'_{a_k} \dots V_{a_1}$.

For a word x, ψ_x is the non-halting part of the QFA's configuration after reading x. It is easy to see that, for any word x and letter a, $\psi_{xa} = V'_a(\psi_x)$.

For a precise definition of multi-tape quantum finite automata we refer [ABFGK99]. However we repeat the most essential part of this definition here below.

$M =_{def} \{1, 2, \dots, m\}$.

$R_I =_{def} A_1 \times A_2 \times \dots \times A_m$, where $A_i = \begin{cases} \{\downarrow, \rightarrow\}, \ if \ i \notin I \\ \{\text{“}nothing\text{”}\}, \ if \ i \in I. \end{cases}$

$T_I =_{def} B_1 \times B_2 \times \dots \times B_m$, where $B_i = \begin{cases} \{\downarrow, \rightarrow\}, \ if \ i \in I \\ \{\text{“}nothing\text{”}\}, \ if \ i \notin I. \end{cases}$

The function $R_i \times T_i \xrightarrow{d_I} \{\downarrow, \rightarrow\}^m$ is defined as follows:

$d_I(r,t) =_{def} (d_I^1(r,t), d_I^2(r,t), \dots, d_I^m(r,t))$, where $d_I^i(r,t) = \begin{cases} r^i, \ if \ i \notin I \\ t^i, \ if \ i \in I. \end{cases}$

Definition 1. A quantum finite multi-tape automaton (QFMA) $A = (Q; \Sigma; \delta; q_0; Q_a; Q_r)$ is specified by the finite input alphabet Σ, the finite set of states Q, the initial state $q_0 \in Q$, the sets $Q_a \subset Q$, $Q_r \subset Q$ of accepting and rejecting states, respectively, with $Q_a \cap Q_r = \emptyset$, and the transition function

$$\delta : Q \times \Gamma^m \times Q \times \{\downarrow, \rightarrow\}^m \longrightarrow \mathbb{C}_{[0,1]},$$

where m is the number of input tapes, $\Gamma = \Sigma \cup \{\#, \$\}$ is the tape alphabet of A and $\#$,$ are end-markers not in Σ, which satisfies the following conditions (of well-formedness):

1. Local probability condition

$$\forall (q_1, \sigma) \in Q \times \Gamma^m \sum_{(q,d) \in Q \times \{\downarrow, \rightarrow\}^m} |\delta(q_1, \sigma, q, d)|^2 = 1.$$

2. Orthogonality of column vectors condition.

$$\forall q_1, q_2 \in Q, q_1 \neq q_2, \forall \sigma \in \Gamma^m \sum_{(q,d) \in Q \times \{\downarrow, \rightarrow\}^m} \delta^*(q_1, \sigma, q, d) \delta(q_2, \sigma, q, d) = 0.$$

3. Separability condition.

$$\forall I \in P(M) \setminus \{\emptyset\}\ \forall q_1, q_2 \in Q$$

$$\forall \sigma_1, \sigma_2 \in \Gamma^m,\ \textit{where}\ \forall i \notin I\ \sigma_1^i = \sigma_2^i$$

$$\forall t_1, t_2 \in T_I,\ \textit{where}\ \forall j \in I\ t_1^j \neq t_2^j$$

$$\sum_{(r,s) \in Q \times R_I} \delta^*(q_1, \sigma_1, q, d_I(r, t_1))\delta(q_2, \sigma_2, q, d_I(r, t_2)) = 0.$$

States from $Q_a \cup Q_r$ are called halting states and states from $Q_{non} = Q \setminus (Q_a \cup Q_r)$ are called non-halting states.

To process an input word vector $x \in (\Sigma^*)^m$ by A it is assumed that the input is written on every tape k with the end-markers in the form $w_x^k = \#x^k\$$ and that every such a tape, of length $|x^k| + 2$, is circular, i. e., the symbol to the right of \$ is $\#$.

For the fixed input word vector x we can define $n \in \mathbb{N}^m$ to be an integer vector which determines the length of input word on every tape. So for every n we can define C_n to be the set of all possible configurations of A where $|x^i| = n^i$. $|C_n| = |Q| \prod_{i=1}^{m} (n^i + 2)$. Every such a configuration is uniquely determined by a pair $|q, s\rangle$, where $q \in Q$ and $0 \leq s^i \leq |x^i| + 1$ specifies the position of head on the i-th tape.

Every computation of A on an input x, $|x^i| = n^i$, is specified by a unitary evolution in the Hilbert space $H_{A,n} = l_2(C_n)$. Each configuration $c \in C_n$ corresponds to the basis vector in $H_{A,n}$. Therefore a global state of A in the space $H_{A,n}$ has a form $\sum_{c \in C_n} \alpha_c |c\rangle$, where $\sum_{c \in C_n} |\alpha_c|^2 = 1$. If the input word vector is x and the automaton A is in its global state $|\psi\rangle = \sum_{c \in C_n} \alpha_c |c\rangle$, then its further step is equivalent to the application of a linear operator U_x^δ over Hilbert space $l_2(C_n)$.

2 Results

It is known [RS 59] that the projection onto one of the tapes of a language of n-tuples of words that can be recognized by n-FDA, is a regular language. Indeed, more is true: the projection onto one of the tapes of an arbitrary language which is accepted by multi-tape finite nondeterministic automata is a regular language. For probabilistic automata, however, the situation is different. Freivalds [Fr 91] constructs a quasideterministic 3-language, the projection of which on the first tape is a nonstochastic language. The purpose of the present paper is to extend this result to 2-languages.

We begin by recording a simple fact, for reference.

Lemma 2. [BFLL98] *Let $\xi_1, \xi_2, \ldots, \xi_t, \ldots$ be a sequence of random natural numbers. Then, for $t \geq 1$,*

$$\max_{\substack{c, c_1, \ldots, c_t \in R \\ c_t \neq 0}} P\{\sum_{1 \leq i \leq t} c_i \xi_i = c\} \leq \max_{j_1, j_2, \ldots, j_t \in N} P\{\xi_t = j_t \mid \xi_1 = j_1, \ldots, \xi_{t-1} = j_{t-1}\}.$$

Corollary 3. *Let M be a natural number and let $\xi_1, \xi_2, \ldots, \xi_t, \ldots$ be a sequence of independent random numbers uniformly distributed over the set $\{1, 2, \ldots, M\}$. Then $P\left\{\sum_{i=1}^{t} c_i \xi_i = c\right\} \leq \frac{1}{M}$ for arbitrary real numbers $c, c_1, \ldots, c_t$ $(c_t \neq 0)$.*

We consider the following 2-language in the alphabet $\{1, 2, 3\}$:

$$B^{(2)} = \left\{(1^{s^2}, 1^1 2 1^3 2 1^5 2 \ldots 2 1^{2s-3} 3 1^{2s-1} 3) \mid s \in N\right\}.$$

The following theorems was proved in [BFLL98].

Theorem 4. [BFLL98] *For arbitrary $\epsilon > 0$, there exists 2-FPA ω recognizing $B^{(2)}$ with probability $1 - \epsilon$.*

Theorem 5. [BFLL98] *Let $P(x) = \sum_{j=0}^{l} c_j x^j$ be a polynomial of degree $l \geq 2$ with non-negative coefficients, mapping the set of natural numbers into itself. Then the language $L = \{1^{P(s)} \mid s \in N\}$ is nonstochastic.*

Now we prove a quantum counterpart of Theorem 5.

Theorem 6. *Let $P(x) = \sum_{j=0}^{l} c_j x^j$ be a polynomial of degree $l \geq 2$ with non-negative coefficients, mapping the set of natural numbers into itself. Then the language $L = \{1^{P(s)} \mid s \in N\}$ is not recognizable by any quantum finite automaton even with unbounded error.*

Before proving Theorem 6 we recall a known fact of Diophantine approximation.

Lemma 7. *Let $1, \psi_1, \psi_2, \ldots, \psi_t$ be real numbers forming a linearly independent system over the field of the rational numbers. Let $P(x)$ be a polynomial of positive degree with rational coefficients. Then the set of the fractional parts of the vectors $P(k)\Psi = (P(k)\psi_1, P(k)\psi_2, \ldots, P(k)\psi_t)$, $k \in N$, is everywhere dense in the unit t-dimensional hypercube.*

Proof: See, for example, [Ca 57] Ch. IV, Theorems III, IV, and VI. The sequence $P(k)\Psi$ of t-dimensional vectors is indeed uniformly distributed modulo 1 if (and only if) so is the 1-dimensional sequence $\tau_1 P(k)\psi_1 + \tau_2 P(k)\psi_2 + \ldots + \tau_t P(k)\psi_t = P(k)\sum_i \tau_i \psi_i$ for every non-zero vector $(\tau_1, \tau_2, \ldots, \tau_t)$ of integers. Since the numbers $1, \psi_1, \psi_2, \ldots, \psi_t$ are linearly independent over the rationals, the number $\psi = \sum_i \tau_i \psi_i$ is irrational, and hence the polynomial ψP has irrational coefficients. However, it is known that the values $Q(k)$, $k \in N$, of a polynomial Q having at least one irrational coefficient are uniformly distributed modulo 1.

Proof of Theorem 6. Assume, on the contrary, that the language L is recognizable by a 1-way quantum finite automaton (maybe with an unbounded error). Then by [BFLL98] there exist a FPA $\mathcal{L} = (\{1\}, S, \Pi_0, M_1, \eta_F)$ and a number γ such that

$$x = 1^k \in L \Longleftrightarrow \Pi_0 M_1^k \eta_F > \gamma. \tag{1}$$

Let s be the cardinality of S and let $\tilde{\lambda}_j = |\tilde{\lambda}_j| e^{2\pi i \varphi_j}$, $j = 1, 2, \ldots, s$, the eigenvalues of the matrix M_1. Let $1, \psi_1, \psi_2, \ldots, \psi_t$ $(t \leq s)$ be a system of linearly independent real numbers over the field of rational numbers such that $\varphi_j = r_{0j} + r_{1j}\psi_1 + \ldots + r_{tj}\psi_t$ with r_{uj} rational, and denote by R the common denominator of the numbers r_{uj}, $u = 0, 1, \ldots, t$, $j = 1, 2, \ldots, s$. Let $c < R$ be a nonnegative integer congruent to c_0 modulo R. Let the bold numeral 1 denote a generic column matrix consisting of 1's only, and put $M = M_1^R$ and $\eta = M_1^c(\eta_F - \gamma \cdot \mathbf{1})$. Since $\Pi_0 M_1^k \mathbf{1} = 1$, it follows from (1) that

$$1^{Rk+c} \in L \Longleftrightarrow \Pi_0 M^k \eta > 0. \tag{2}$$

Put $|\lambda_j| = |\tilde{\lambda}_j|^R$, $\Theta_j = (\varphi_j - r_{oj})$ and $\lambda_j = |\lambda_j| e^{2\pi i \Theta_j}$, and let Z be a Jordan normal form of the matrix M, $M = T^{-1} Z T$. The numbers λ_j are clearly the eigenvalues of the matrix M. We have

$$F(k) = \Pi_0 M^k \eta = \Pi_0 T^{-1} Z^k T \eta = \sum_{m=1}^{s} |\lambda_m|^k \sum_{j=0}^{s-1} k^j a_{mj} e^{2\pi i (k\Theta_m + \alpha_{mj})}$$

where a_{mj}, α_{mj} are real, and, since $F(k)$ is also real, we may forthwith replace the complex exponential by the cosine. To simplify further, partition the set $J = \{\lambda_j \mid 1 \leq j \leq s\}$ of eigenvalues by their modulus: $J = \cup_{1 \leq \nu \leq N} J_\nu$, with J_ν consisting of all the eigenvalues of equal modulus ρ_ν and $\rho_1 < \ldots < \rho_N$. Write $F(k)$ in the form

$$F(k) = \sum\nolimits_\nu \rho_\nu^k \sum\nolimits_j k^j B_\nu(k, j)$$

where we have put

$$B_\nu(k, j) = \sum\nolimits_{m \in J_\nu} a_{mj} \cos 2\pi (k\Theta_m + \alpha_{mj}).$$

Notice that the functions $k \mapsto B_\nu(k, j)$ cannot all vanish identically because F itself does not; indeed, since $P(Rk) \equiv c_0 \equiv c (mod R)$, we have by (2),

$$F(Q(l)) > 0 \ \ if \ \ Q(l) = \frac{P(Rl) - c}{R} \ \ and \ l \in N. \tag{3}$$

One may thus pick the largest index ν_0 among all ν for which $\sum_j k^j B_\nu(k, j)$ is not identically zero as a function of k, and then pick the largest index j_0 among all j for which $k \mapsto B_{\nu_0}(k, j)$ is not identically zero. Put for short $B(k) = B_{\nu_0}(k, j_0)$ and pick a value $k = k_0$ for which $B(k_0) \neq 0$.

Notice that if $k_\nu \to \infty$ is a sequence of integers for which $B(k_\nu) \to B(k_0)$ then

$$F(k) = \rho_{\nu_0}^k k^{j_0} (B(k) + o(1)) \ as \ \ k = k_\nu \to \infty, \tag{4}$$

which for large ν forces the signs of $F(k_\nu)$ and $B(k_0)$ to coincide. By Lemma 7 we may first pick such a sequence k_ν of the form $Q(n_\nu)$ and conclude by (3) that $B(k_0) > 0$. Picking now k_ν a second time, this time of the form $Q(m_\nu)+1$, we then see that $F(Q(m_\nu)+1) > 0$ for large ν. This implies by (2) that for all such ν one has $R(Q(m_\nu)+1)+c = P(l_\nu)$ for some $l_\nu \in N$, or, simplified, $P(Rm_\nu)+R = P(l_\nu)$. However, since P is of degree at least two and has positive coefficients, it is clear that this equation has no solution $l_\nu \in N$ if m_ν is large enough.

The assumption has lead us to a contradiction. ⊠

Theorem 8. *There exists a language of pairs of words such that for each positive number ϵ this language is recognized by a finite probabilistic automaton with probability $1-\epsilon$, but the projection of this language to one of the tapes is not recognized by any quantum finite automaton (even with unbounded error).*

Proof: Consider the language $B^{(2)}$. By Theorem 4, there exists a 2-FPA ω which recognizes this language with probability $1-\epsilon$. On the other hand, the projection of $B^{(2)}$ to the first tape is the language $\{1^{s^2} \mid s \in N\}$, which is nonstochastic by Theorem 5. ⊠

This result raises a question. We would like to know whether the language $B^{(2)}$ can be recognized with a bounded error by a quantum finite automaton. We believe that it cannot. Moreover, we believe that this result may be obtained by a technique similar to that of the proofs of Proposition 7 in [KW 97] and Theorem 2 in [AF 98]. However, by [ABFGK99] there exists a language, namely L_1, which is recognized by a quantum finite 2-tape automaton with probability $1-\epsilon$, but the projection of this language to the first tape is not regular.

References

[AF 98] Andris Ambainis and Rūsiņš Freivalds. 1-way quantum finite automata: strengths, weaknesses and generalizations. *Proc. 39th FOCS*, 1998, pp. 332–341. *http://xxx.lanl.gov/abs/quant-ph/9802062*

[ABFGK99] Andris Ambainis, Richard Bonner, Rūsiņš Freivalds, Marats Golovkins, and Marek Karpinski. *Quantum Finite Multitape Automata vs Probabilistic Finite Multitape Automata*. This volume.

[BFLL98] Richard Bonner, Rūsiņš Freivalds, Jānis Lapiņš, and Antra Lukjanska. *Nonstochastic languages as projections of 2-tape quasideterministic languages.* "Lecture Notes in Computer Science", Springer, 1998, v. 1450, p. 213-219.

[BP99] Alex Brodsky and Nicholas Pippenger. *Characterizations of 1-way quantum finite automata. http : //xxx.lanl.gov/abs/quant − ph/9903014*

[Fr 78] Rūsiņš Freivalds. *Recognition of languages with high probability by various types of automata.* "Dokladi AN SSSR", 1978, v. 239, No. 1, p. 60-62 (in Russian)

[Fr 81] Rūsiņš Freivalds. *Projections of languages recognizable by probabilistic and alternating finite multitape automata.* "Information Processing Letters", 1981, v. 13, No. 4/5, p. 195-198.

[Fr 91] Rūsiņš Freivalds. *Complexity of probabilistic versus deterministic automata.* "Lecture Notes in Computer Science", Springer, 1991, v. 502, p. 565-613

[KW 97] Attila Kondacs and John Watrous. On the power of quantum finite state automata. In *Proc. 38th FOCS*, 1997, pp. 66–75.

[RS 59] M.O. Rabin and D. Scott. *Finite automata and their decision problems.* "J. Res. Develop.", 1959, v. 3, No. 2, p. 114-125.

[Mi 66] B.T. Mirkin. *Towards the theory of multitape automata.* "Kibernetika", 1966, No 5, p. 12-18. (in Russian)

[Ca 57] J.W.S. Cassels. *An Introduction to Diophantine Approximation.* Cambridge Tracts in Mathematics and Mathematical Physics, vol. 45, 1957.

An Introduction to Quantum Pushdown Automata

Marats Golovkins

Institute of Mathematics and Computer Science, University of Latvia, Raiņa bulv. 29, Riga, Latvia*
marats@cclu.lv

Abstract. Quantum finite automata, as well as quantum pushdown automata were first introduced by C. Moore, J. P. Crutchfield [MC 97]. In this paper we introduce the notion of quantum pushdown automata in a non-equivalent way, including unitarity criteria (well-formedness conditions), by using the definition of quantum finite automata of [KW 97]. It is also introduced the notion of simplified quantum pushdown automata and corresponding well-formedness conditions. It is established that the unitarity criteria of quantum pushdown automata are not equivalent to the corresponding unitarity criteria of quantum Turing machines [BV 97].

1 Introduction

Nobel prize winner physicist R. Feynman asked in 1982, what effects may have the principles of quantum mechanics on computation [Fe 82]. He gave arguments that it may require exponential time to simulate quantum mechanical processes on classical computers. This served as a basis to the opinion that quantum computers may have advantages versus classical ones. It was in 1989, when D. Deutsch introduced the notion of quantum Turing machine [De 89] and proved that quantum Turing machines compute the same recursive functions as classical deterministic Turing machines do. P. Shor discovered that by use of quantum algorithms it is possible to factorize large integers and compute discrete logarithms in a polynomial time [Sh 94], what resulted into additional interest in quantum computing and attempts to create quantum computers. First steps have been made to this direction, and first quantum computers which memory is limited by a few quantum bits have been constructed [KLMT 99]. To make quantum computers with larger memory feasible, one of the problems is to minimize error possibilities in quantum bits. Quantum error correction methods are developed [CRSS 98] which would enable quantum computers with larger quantum memory.

Quantum mechanics differs from the classical physics substantially. It is enough to mention *Heisenberg's uncertainty principle*, which states that it is impossible to get information about different parameters of quantum particle

* Research supported by Grant No.96.0282 from the Latvian Council of Science

simultaneously precisely. Another well known distinction is the impossibility to observe quantum object without changing it.

Fundamental concept of quantum information theory is *quantum bit*. Classical information theory is based on classical bit, which has two states 0 and 1. The next step is *probabilistic bit*, which can be 0 with probability α and 1 with probability β, where $\alpha + \beta = 1$. Quantum bit or *qbit* is similar to probabilistic bit with the difference that α and β are complex numbers with the property $|\alpha|^2 + |\beta|^2 = 1$. It is common to denote qbit as $\alpha|0\rangle + \beta|1\rangle$. As a result of *measurement*, we get 0 with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$.

Every computation done on qbits is accomplished by means of unitary operators. Informally, every unitary operator can be interpreted as a revolution in complex space. Therefore one of the basic properties of unitary operators is that every quantum computing process not disturbed by measurements is reversible. Unitarity is rather hard requirement which complicates programming of quantum devices. The following features of quantum computers are most important:

1. Information is represented by qbits.
2. Any step of computation can be represented as a unitary operation, therefore computation is reversible.
3. Quantum information cannot be copied.
4. Quantum parallelism; quantum computer can compute several paths simultaneously, however as a result of measurement it is possible to get the results of only one computation path.

Quantum finite automata [KW 97] are considered to be the most elementary model of quantum device. Quantum pushdown automata were first defined in [MC 97], however the authors do not mention clear criteria to ensure unitarity in the sense of respective definition. The definition in [MC 97] is not equivalent to that given in this paper.

The following notations will be used further in the paper:
z^* is the complex conjugate of a complex number z.
U^* is the Hermitian conjugate of a matrix U.
I is the identity matrix.
ϵ is empty word.

Definition 1. Matrix U is called *unitary*, if $UU^* = U^*U = I$.

If U is a finite matrix, then $UU^* = I$ iff $U^*U = I$. However this is not true for infinite matrices:

Example 1.

$$U = \begin{pmatrix} \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & \dots \\ \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & 0 & 1 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

Here $U^*U = I$ but $UU^* \neq I$.

Theorem 2. *If infinite matrices A, B, C have finite number of nonzero elements in each row and column, then their multiplication is associative: $(AB)C = A(BC)$.*

Proof. The element of matrix $(AB)C$ in i-th row and j-th column is $k_{ij} = \sum_{s=1}^{\infty}\sum_{r=1}^{\infty} a_{ir}b_{rs}c_{sj}$. The element of matrix $A(BC)$ in the same row and column is $l_{ij} = \sum_{r=1}^{\infty}\sum_{s=1}^{\infty} a_{ir}b_{rs}c_{sj}$. As in each row and column of matrices A, B, C there is finite number of nonzero elements, it is also finite in the given series. Therefore the elements of series can be rearranged, and $k_{ij} = l_{ij}$. □

As noted further in the paper infinite matrices with finite number of nonzero elements in each row and column describe the work of pushdown automata. In further theorems there are stated some properties of such matrices.

Theorem 3. *If $U^*U = I$, then the norm of any row in the matrix U does not exceed* 1.

Proof. Let us consider the matrix $S = UU^*$. The element of this matrix $s_{ij} = \langle r_j|r_i\rangle$, where r_i is i-th row of the matrix U. Let us consider the matrix $T = S^2$. The diagonal element of this matrix is

$$t_{ii} = \sum_{k=1}^{\infty} s_{ik}s_{ki} = \sum_{k=1}^{\infty}\langle r_k|r_i\rangle\langle r_i|r_k\rangle = \sum_{k=1}^{\infty}|\langle r_k|r_i\rangle|^2.$$

On the other hand, taking into account Theorem 2, we get that

$$T = S^2 = (UU^*)(UU^*) = U(U^*U)U^* = UU^* = S.$$

Therefore $t_{ii} = s_{ii} = \langle r_i|r_i\rangle$. It means that

$$\sum_{k=1}^{\infty}|\langle r_k|r_i\rangle|^2 = \langle r_i|r_i\rangle. \tag{1}$$

This implies that every element of series (1) does not exceed $\langle r_i|r_i\rangle$. Hence $|\langle r_i|r_i\rangle|^2 = \langle r_i|r_i\rangle^2 \leq \langle r_i|r_i\rangle$. The last inequality implies that $0 \leq \langle r_i|r_i\rangle \leq 1$. Therefore $|r_i| \leq 1$. □

Theorem 4. *Let us assume that $U^*U = I$. Then the rows of the matrix U are orthogonal iff every row of the matrix has norm* 0 *or* 1.

Proof. Let us assume that the rows of the matrix U are orthogonal. Let us consider equation (1) from the proof of Theorem 3, i.e., $\sum_{k=1}^{\infty}|\langle r_k|r_i\rangle|^2 = \langle r_i|r_i\rangle$.

As the rows of the matrix U are orthogonal, $\sum_{k=1}^{\infty} |\langle r_k|r_i\rangle|^2 = |\langle r_i|r_i\rangle|^2$. Hence $\langle r_i|r_i\rangle^2 = \langle r_i|r_i\rangle$, i.e., $\langle r_i|r_i\rangle = 0$ or $\langle r_i|r_i\rangle = 1$. Therefore $|r_i| = 0$ or $|r_i| = 1$.

Let as assume that every row of the matrix has norm 0 or 1. Then $\langle r_i|r_i\rangle^2 = \langle r_i|r_i\rangle$ and in compliance with the equation (1), $\sum_{k\in\mathbf{N}^+\setminus\{i\}} |\langle r_k|r_i\rangle|^2 = 0$. This implies that $\forall k \neq i\ |\langle r_k|r_i\rangle| = 0$. Hence the rows of the matrix are orthogonal. □

Theorem 5. *The matrix U is unitary iff $U^*U = I$ and its rows are normalized.*

Proof. Let us assume that the matrix U is unitary. Then in compliance with Definition 1, $U^*U = I$ and $UU^* = I$, i.e, the rows of the matrix are orthonormal.

Let us assume that $U^*U = I$ and the rows of the matrix are normalized. Then in compliance with Theorem 4 the rows of the matrix are orthogonal. Hence $UU^* = I$ and the matrix is unitary. □

2 Quantum pushdown automata

Definition 6. A quantum pushdown automaton (QPA) $A = (Q, \Sigma, T, q_0, Q_a, Q_r, \delta)$ is specified by the finite set of states Q, the finite input alphabet Σ and the stack alphabet T, the initial state $q_0 \in Q$, the sets $Q_a \subset Q, Q_r \subset Q$ of accepting and rejecting states, respectively, with $Q_a \cap Q_r = \emptyset$, and the transition function

$$\delta : Q \times \Gamma \times \Delta \times Q \times \{\downarrow, \rightarrow\} \times \Delta^* \longrightarrow \mathbb{C}_{[0,1]},$$

where $\Gamma = \Sigma \cup \{\#, \$\}$ is the input tape alphabet of A and $\#, \$$ are end-markers not in Σ, $\Delta = T \cup \{Z_0\}$ is the working stack alphabet of A and $Z_0 \notin T$ is the stack base symbol; $\{\downarrow, \rightarrow\}$ is the set of directions of input tape head. The automaton must satisfy *the conditions of well-formedness*, which will be expressed below. Furthermore, the transition function is restricted to the following requirement: If $\delta(q, \alpha, \beta, q', d, \omega) \neq 0$, then$|\omega| \leq 2$; if $|\omega| = 2$, then $\omega_1 = \beta$; if $\beta = Z_0$, then $\omega \in Z_0T^*$; if $\beta \neq Z_0$, then $\omega \in T^*$.

Definition 6 utilizes that of classical pushdown automata from [Gu 89].

Well-formedness conditions 2.1 *1. Local probability condition.*

$$\forall(q_1, \sigma_1, \tau_1) \in Q \times \Gamma \times \Delta \qquad \sum_{(q,d,\omega)\in Q\times\{\downarrow,\rightarrow\}\times\Delta^*} |\delta(q_1, \sigma_1, \tau_1, q, d, \omega)|^2 = 1. \tag{2}$$

2. Orthogonality of column vectors condition.

$$\text{For all triples } (q_1, \sigma_1, \tau_1) \neq (q_2, \sigma_1, \tau_2) \text{ in } Q \times \Gamma \times \Delta \qquad \sum_{(q,d,\omega)\in Q\times\{\downarrow,\rightarrow\}\times\Delta^*} \delta^*(q_1, \sigma_1, \tau_1, q, d, \omega)\delta(q_2, \sigma_1, \tau_2, q, d, \omega) = 0. \tag{3}$$

3. Row vectors norm condition.

$$\forall(q_1,\sigma_1,\sigma_2,\tau_1,\tau_2)\in Q\times\Gamma^2\times\Delta^2$$
$$\sum_{(q,\tau,d,\omega)\in Q\times\Delta\times\{\downarrow,\rightarrow\}\times\{\varepsilon,\tau_2,\tau_1\tau_2\}}|\delta(q,\sigma_1,\tau,q_1,d,\omega)|^2=1. \tag{4}$$

4. Separability condition I.

$$\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_1,\tau_2)\in Q\times\Gamma\times\Delta,\ \forall\tau_3\in\Delta$$
$$a)\ \sum_{(q,d,\tau)\in Q\times\{\downarrow,\rightarrow\}\times\Delta}\delta^*(q_1,\sigma_1,\tau_1,q,d,\tau)\delta(q_2,\sigma_1,\tau_2,q,d,\tau_3\tau)+$$
$$+\sum_{(q,d)\in Q\times\{\downarrow,\rightarrow\}}\delta^*(q_1,\sigma_1,\tau_1,q,d,\varepsilon)\delta(q_2,\sigma_1,\tau_2,q,d,\tau_3)=0; \tag{5}$$
$$b)\ \sum_{(q,d)\in Q\times\{\downarrow,\rightarrow\}}\delta^*(q_1,\sigma_1,\tau_1,q,d,\varepsilon)\delta(q_2,\sigma_1,\tau_2,q,d,\tau_2\tau_3)=0. \tag{6}$$

5. Separability condition II.

$$\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_2,\tau_2)\in Q\times\Gamma\times\Delta$$
$$\sum_{(q,\omega)\in Q\times\Delta^*}\delta^*(q_1,\sigma_1,\tau_1,q,\downarrow,\omega)\delta(q_2,\sigma_2,\tau_2,q,\rightarrow,\omega)=0. \tag{7}$$

6. Separability condition III.

$$\forall(q_1,\sigma_1,\tau_1),(q_2,\sigma_2,\tau_2)\in Q\times\Gamma\times\Delta,\ \forall\tau_3\in\Delta,\ \forall d_1,d_2\in\{\downarrow,\rightarrow\},\ d_1\neq d_2$$
$$a)\ \sum_{(q,\tau)\in Q\times\Delta}\delta^*(q_1,\sigma_1,\tau_1,q,d_1,\tau)\delta(q_2,\sigma_2,\tau_2,q,d_2,\tau_3\tau)+$$
$$+\sum_{q\in Q}\delta^*(q_1,\sigma_1,\tau_1,q,d_1,\varepsilon)\delta(q_2,\sigma_2,\tau_2,q,d_2,\tau_3)=0; \tag{8}$$
$$b)\ \sum_{q\in Q}\delta^*(q_1,\sigma_1,\tau_1,q,d_1,\varepsilon)\delta(q_2,\sigma_2,\tau_2,q,d_2,\tau_2\tau_3)=0. \tag{9}$$

Let us assume that the automaton is in a state q, its input tape head is above a symbol α and the stack head is above a symbol β. Then the automaton undertakes the following actions with an amplitude $\delta(q,\alpha,\beta,q',d,\omega)$:

1. goes into the state q';
2. if $d=\text{`}\rightarrow\text{'}$, moves the input tape head one cell forwards;
3. takes out of the stack the symbol β (deletes it and moves the stack head one cell backwards);
4. starting with the first empty cell, puts into the stack the string ω, moving the stack head $|\omega|$ cells forwards.

Definition 7. The *configuration* of a pushdown automaton is a pair $|c\rangle = |\nu_i q_j \nu_k, \omega_l\rangle$, where the automaton is in a state $q_j \in Q$, $\nu_i\nu_k \in \#\Sigma^*\$$ is a finite word on the input tape, $\omega_l \in Z_0 T^*$ is a finite word on the stack tape, the input tape head is above the first symbol of the word ν_k and the stack head is above the last symbol of the word ω_l.

We shall denote by C the set of all configurations of the pushdown automaton. The set C is countably infinite. Every configuration $|c\rangle$ denotes a basis vector in the space $H_A = l_2(C)$. Therefore a global state of A in the space H_A has a form $|\psi\rangle = \sum_{c\in C} \alpha_c |c\rangle$, where $\sum_{c\in C} |\alpha_c|^2 = 1$ and $\alpha_c \in \mathbb{C}$ denotes the amplitude of a configuration $|c\rangle$. If the automaton is in its global state (superposition) $|\psi\rangle$, then its further step is equivalent to the application of a linear operator (evolution) U_A over the space H_A.

Definition 8. The linear operator U_A is defined as follows:

$$U_A|\psi\rangle = \sum_{c\in C} \alpha_c U_A |c\rangle.$$

If a configuration $c = |\nu_i q_j \sigma \nu_k, \omega_l \tau\rangle$, then

$$U_A|c\rangle = \sum_{(q,d,\omega)\in Q\times\{\downarrow,\rightarrow\}\times\Delta^*} \delta(q_j,\sigma,\tau,q,d,\omega)|f(|c\rangle,d,q),\omega_l\omega\rangle,$$

where

$$f(|\nu_i q_j \sigma \nu_k, \omega_l \tau\rangle, d, q) = \begin{cases} \nu_i q \sigma \nu_k, \text{ if } d = \text{`}\downarrow\text{'} \\ \nu_i \sigma q \nu_k, \text{ if } d = \text{`}\rightarrow\text{'}. \end{cases}$$

Remark. Although the QPA evolution operator matrix is infinite, it has a finite number of nonzero elements in each row and column, as it is possible to reach only a finite number of other configurations from a given configuration within one step, all the same, within one step the given configuration is reachable only from a finite number of different configurations.

Language recognition for QPA is defined as follows. For a QPA $A = (Q, \Sigma, T, q_0, Q_a, Q_r, \delta)$ we define $C_a = \{|\nu_i q \nu_k, \omega_l\rangle \in C \mid q \in Q_a\}$, $C_r = \{|\nu_i q \nu_k, \omega_l\rangle \in C \mid q \in Q_r\}$, $C_n = C \setminus (C_a \cup C_r)$. E_a, E_r, E_n are subspaces of H_A spanned by C_a, C_r, C_n respectively. We use the observable O that corresponds to the orthogonal decomposition $H_A = E_a \oplus E_r \oplus E_n$. The outcome of each observation is either "accept" or "reject" or "non-halting".

The language recognition is now defined as follows: For an $x \in \Sigma^*$ we consider as the input $\#x\$$, and assume that the computation starts with A being in the configuration $|q_0 \#x\$, Z_0\rangle$. Each computation step consists of two parts. At first the linear operator U_A is applied to the current global state and then the resulting superposition is observed using the observable O as defined above. If the global state before the observation is $\sum_{c\in C} \alpha_c |c\rangle$, then the probability that the resulting superposition is projected onto the subspace E_i, $i \in \{a, r, n\}$, is $\sum_{c\in C_i} |\alpha_c|^2$. The computation continues until the result of an observation is "accept" or "reject".

Lemma 9. *The columns system of a QPA evolution matrix is normalized iff the condition (2), i.e., local probability condition, is satisfied.*

Lemma 10. *The columns system of a QPA evolution matrix is orthogonal iff the conditions (3,5,6,7,8,9), i.e., orthogonality of column vectors and separability conditions, are satisfied.*

Lemma 11. *The rows system of a QPA evolution matrix is normalized iff the condition (4), i.e., row vectors norm condition, is satisfied.*

Theorem 12. *Well-formedness conditions 2.1 are satisfied iff the evolution operator U_A is unitary.*

Proof. Lemmas 9, 10, 11 imply that Well-formedness conditions 2.1 are satisfied iff the columns of the evolution matrix are orthonormal and rows are normalized. In compliance with Theorem 5, columns are orthonormal and rows are normalized iff the matrix is unitary. □

Remark. Well-formedness conditions 2.1 contain the requirement that rows system has to be normalized, which is not necessary in the case of quantum Turing machine [BV 97]. Here is taken into account the fact that the evolution of QPA can violate the unitarity requirement if the row vectors norm condition is omitted.

Example 2. A QPA, which evolution matrix columns are orthonormal, however the evolution is not unitary.

$$Q = \{q\},\ \Sigma = \{1\},\ T = \{1\}.$$
$$\delta(q, \#, Z_0, q, \rightarrow, Z_0 1) = 1, \qquad \delta(q, \#, 1, q, \rightarrow, 11) = 1,$$
$$\delta(q, 1, Z_0, q, \rightarrow, Z_0 1) = 1, \qquad \delta(q, 1, 1, q, \rightarrow, 11) = 1,$$
$$\delta(q, \$, Z_0, q, \rightarrow, Z_0 1) = 1, \qquad \delta(q, \$, 1, q, \rightarrow, 11) = 1,$$

other values of arguments yield $\delta = 0$.

By Well-formedness conditions 2.1, the columns of the evolution matrix are orthonormal, but the matrix is not unitary, because the norm of the rows specified by the configurations $|\omega, Z_0\rangle$ is 0.

Example 3. A QPA, which evolution is unitary.

$$Q = \{q_1, q_2\},\ \Sigma = \{1\},\ T = \{1\}.$$
$$\forall \sigma \in \{\#, \$, 1\};$$
$$\delta(q_2, \sigma, Z_0, q_2, \rightarrow, Z_0) = 1, \qquad \delta(q_1, \sigma, Z_0, q_2, \rightarrow, Z_0 1) = 1,$$
$$\delta(q_2, \sigma, 1, q_1, \rightarrow, \epsilon) = \tfrac{1}{\sqrt{2}}, \qquad \delta(q_1, \sigma, 1, q_1, \rightarrow, \epsilon) = \tfrac{1}{\sqrt{2}}$$
$$\delta(q_2, \sigma, 1, q_2, \rightarrow, 11) = \tfrac{1}{\sqrt{2}}, \qquad \delta(q_1, \sigma, 1, q_2, \rightarrow, 11) = -\tfrac{1}{\sqrt{2}},$$

other values of arguments yield $\delta = 0$.

Even in the case of trivial QPA, it is a cumbersome task to check all the conditions of well-formedness 2.1. It is possible to relax the conditions slightly by introducing the notion of *simplified* QPA.

Definition 13. We shall say that a QPA is *simplified*, if there exists a function $D : Q \longrightarrow \{\downarrow, \rightarrow\}$, and $\delta(q_1, \sigma, \tau, q, d, \omega) = 0$, if $D(q) \neq d$. Therefore the transition function of a simplified QPA is

$$\varphi(q_1, \sigma, \tau, q, \omega) = \delta(q_1, \sigma, \tau, q, D(q), \omega).$$

Taking into account Definition 13, following well-formedness conditions correspond to simplified QPA:

Well-formedness conditions 2.2 *1. Local probability condition.*

$$\forall (q_1, \sigma_1, \tau_1) \in Q \times \Gamma \times \Delta \qquad \sum_{(q,\omega) \in Q \times \Delta^*} |\varphi(q_1, \sigma_1, \tau_1, q, \omega)|^2 = 1. \tag{10}$$

2. Orthogonality of column vectors condition.

$$\text{For all triples } (q_1, \sigma_1, \tau_1) \neq (q_2, \sigma_1, \tau_2) \text{ in } Q \times \Gamma \times \Delta \qquad \sum_{(q,\omega) \in Q \times \Delta^*} \varphi^*(q_1, \sigma_1, \tau_1, q, \omega)\varphi(q_2, \sigma_1, \tau_2, q, \omega) = 0. \tag{11}$$

3. Row vectors norm condition.

$$\forall (q_1, \sigma_1, \sigma_2, \tau_1, \tau_2) \in Q \times \Gamma^2 \times \Delta^2 \qquad \sum_{(q,\tau,\omega) \in Q \times \Delta \times \{\varepsilon, \tau_2, \tau_1\tau_2\}} |\varphi(q, \sigma_1, \tau, q_1, \omega)|^2 = 1. \tag{12}$$

4. Separability condition.

$$\forall (q_1, \sigma_1, \tau_1), (q_2, \sigma_1, \tau_2) \in Q \times \Gamma \times \Delta, \ \forall \tau_3 \in \Delta$$

$$a) \sum_{(q,\tau) \in Q \times \Delta} \varphi^*(q_1, \sigma_1, \tau_1, q, \tau)\varphi(q_2, \sigma_1, \tau_2, q, \tau_3\tau) + \sum_{q \in Q} \varphi^*(q_1, \sigma_1, \tau_1, q, \varepsilon)\varphi(q_2, \sigma_1, \tau_2, q, \tau_3) = 0; \tag{13}$$

$$b) \sum_{q \in Q} \varphi^*(q_1, \sigma_1, \tau_1, q, \varepsilon)\varphi(q_2, \sigma_1, \tau_2, q, \tau_2\tau_3) = 0. \tag{14}$$

Theorem 14. *The evolution of a simplified QPA is unitary iff Well-formedness conditions 2.2 are satisfied.*

Proof. By Theorem 12 and Definition 13. □

References

[BV 97] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26:1411-1473, 1997.

[CRSS 98] A. Robert Calderbank, Eric M. Rains, Peter W. Shor, Neil J. A. Sloane. Quantum error correction via codes over GF(4). *IEEE Transactions on Information Theory*, 1998, vol. 44, p. 1369-1387.

[De 89] David Deutsch. Quantum Theory, the Church-Turing principle and the universal quantum computer. *Proc. Royal Society London, A400*, 1989. p. 96-117.

[Fe 82] Richard Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 1982, vol. 21, No 6/7, p. 467-488.

[Gu 89] E. Gurari. An introduction to the theory of computation. *Computer Science Press*, 1989.

[KLMT 99] E. Knill, R. Laflamme, R. Martinez, C.-H. Tseng. A cat-state benchmark on a seven bit quantum computer. http://xxx.lanl.gov/abs/quant-ph/9908051

[KW 97] Attila Kondacs and John Watrous. On the power of quantum finite state automata. In *Proc. 38th FOCS*, 1997, p. 66–75.

[MC 97] Christopher Moore, James P. Crutchfield. Quantum automata and quantum grammars. http://xxx.lanl.gov/abs/quant-ph/9707031

[Sh 94] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. *Proc. 35th FOCS*, 1994, p. 124-134.