Quantum Information and Computation, Vol. 2, No. 4 (2002) 324–332
© Rinton Press

ROM-BASED COMPUTATION: QUANTUM VERSUS CLASSICAL

B. C. Travaglione, M. A. Nielsen
Centre for Quantum Computer Technology, University of Queensland
St Lucia, Queensland, Australia

H. M. Wiseman
Centre for Quantum Dynamics, School of Science, Griffith University
Nathan, Queensland, Australia

A. Ambainis
Computer Science Division, University of California
Berkeley, California, USA

Received January 24, 2002
Revised May 20, 2002

We introduce a model of computation based on *read only memory* (ROM), which allows us to compare the space-efficiency of reversible, error-free classical computation with reversible, error-free quantum computation. We show that a ROM-based quantum computer with one writable qubit is universal, whilst two writable bits are required for a universal classical ROM-based computer. We also comment on the time-efficiency advantages of quantum computation within this model.

Keywords: quantum space complexity, read only memory, reversible computation
Communicated by: R Jozsa & M Mosca

1. Introduction

To date, the main drive of research into quantum computation has been to show that the time requirements for solving certain problems are smaller for a quantum computer than they are for a classical computer. Perhaps the most well known result is Shor's algorithm[1], which enables a quantum computer to factor large integers with a subexponential speed-up over the best known classical solution. Other examples of increased time-efficiency using quantum computation are the Deutsch-Jozsa algorithm[2], which provides an *unbounded* speed-up over classical *deterministic* algorithms[a], and Grover's search algorithm[3], which provides a polynomial speed-up. For a general introduction to quantum computation, the reader could consult Nielsen and Chuang[4] or Preskill[5].

Whilst time is often considered the key resource to be minimized during the solving of a problem, another resource of considerable importance is space. Space complexity is the study of the number of (qu)bits required by a computer to solve a problem. At present the

[a]The Deutsch-Jozsa algorithm makes $O(1)$ quantum queries as opposed to the $\Omega(n)$ required deterministically classically.

experimentally viable 'quantum computers' have fewer than ten qubits[6, 7], thus the question of what can be computed using small quantum computers is of interest. As is conventional in space complexity theory, we shall differentiate between *read-only* memory (ROM) and *writable* memory[8]. The space complexity will be a function of the writable memory only. Previous work on space-bounded quantum computation has looked at quantum Turing machines[9] and quantum finite-state automata[10], both of which are bounded-error models. In this paper we introduce a model which allows us to compare the space complexity of error-free, reversible quantum and classical computation.

Some of the results discussed in this paper are related to those found in Barenco et al. [11], however we shall be requiring that most of the bits in our circuits are *read only*, a restriction which did not need to be addressed in [11]. It is this restriction which allows us to demonstrate the difference in space complexity of quantum and classical ROM computation.

The structure of this paper is as follows. In Section 2 we explain in detail our ROM-based computation model. In Section 3 we show that a ROM-based quantum computer with one writable qubit is universal. In Section 4 we show that two writable bits are required for a universal classical ROM-based computer. Finally, in Section 5 we comment on time-efficiency within the model.

2. ROM-based Computation

In this paper we are considering mappings between strings of boolean variables (bits) of the following form,

$$u_1 u_2 \ldots u_j \underbrace{00\ldots0}_{n\ \text{(qu)bits}} \xrightarrow{F} u_1 u_2 \ldots u_j f_1 f_2 \ldots f_n, \tag{1}$$

where each $u_i \in \{0,1\}$ and each $f_i \in \{0,1\}$. It is evident from Eq. (1) that the first j bits have the same initial and final values, however in our model, we shall require that the values of the first j bits are also not altered during any of the steps of the computation, so we can consider them to be *read-only memory* or ROM bits. Each of the last n bits are mapped to zero or one, depending on the values of the ROM bits. Therefore we can think of each of these n bits as *writable* bits, whose final value is a boolean function of the ROM-bits,

$$f_i(u_1, u_2, \ldots, u_j) \quad : \quad \mathbb{B}_2^j \to \mathbb{B}_2 \quad i \in \{1, 2, \ldots, n\}, \tag{2}$$

where $\mathbb{B}_2^j$ denotes a binary string of length j. In the classical case, a given function f_i is generated by a sequence of arbitrary classical *reversible* gates acting on the n writable bits. Additionally, any of these gates can be applied conditionally upon the value of *one* of the j ROM bits. We are using only reversible gates to preserve the number of writable bits. Any irreversible gate which increases the number of writable bits (e.g. FANOUT) has an associated space complexity cost, whilst irreversible gates which reduce the number of writeable bits (e.g. AND) dissipate energy, and therefore have a thermodynamic cost [12].

In the quantum case, arbitrary quantum gates can be applied to the n qubits, and once again any of these gates can be applied conditionally upon the value of *one* of the j ROM bits. However, it should be remembered that each of the f_i are boolean expressions, thus whilst the qubits can exist in superpositional states during the computation, at the conclusion they

must be in a computational basis state. This means that the entire computation (including measurement) is deterministic and reversible, as measuring the n qubits at the end of the computation will have no effect on their state. Intermediate measurements can be made in neither the quantum or classical models, as the storing of the measurement result would be effectively expanding the workspace.

It is perhaps natural to question why we are allowing a given gate to be conditional on only *one* of the ROM bits. Generally, in both quantum and classical computation, arbitrary numbers of controls are allowed, as these can always be broken down into gates containing a fixed number of controls (two in the case of quantum computation[13], and three in the case of classical computation[14]). If arbitrary numbers of controls are allowed it is trivial to show that a one (qu)bit ROM computer is universal. However, breaking down such conditional gates requires the conditional bits to be writable, and therefore has an associated space complexity cost. It should also be pointed out that there is nothing unique about allowing only one control ROM bit per gate. The results presented in the Sections 3 and 4 would be unaffected by allowing any *fixed* number of simultaneous conditional ROM bits. In particular, a one bit classical computer will still not be universal, as will be shown in Section 4.

Throughout this paper we shall be using circuit diagrams to represent our ROM-based computations. As is standard in quantum computational circuit diagrams, the writable (qu)bits will be represented as horizontal lines, whose states change as various gates are applied from left to right. The ROM bits will be depicted above the circuit diagram, with a line from a ROM bit to a gate implying that this gate is applied only if the ROM bit has value one. Fig. 1 contains an example of a ROM computation circuit diagram. This diagram depicts the computation

$$u_1u_2u_3|0\rangle|0\rangle \quad \xrightarrow{F} \quad u_1u_2u_3|f_1\rangle|f_2\rangle, \tag{3}$$

where

$$\begin{aligned} |f_1(u_1,u_3)\rangle &= |u_1 \oplus u_3\rangle \quad \text{and} \\ |f_2(u_1,u_2)\rangle &= |u_1 \oplus u_1u_2\rangle. \end{aligned} \tag{4}$$

Please note that we shall be using kets to denote the writable elements of a ROM-based computer, irrespective of whether these elements are bits or qubits.

There are $2^\wedge(n2^j)$ Boolean functions from j bits to n bits. We shall define as *universal* a ROM-based computer which of capable of calculating all of these functions. In Section 3 we show that *one* writable qubit is sufficient for a universal ROM-based quantum computer, whilst in Section 4 we show that *two* writable bits are required for a universal ROM-based classical computer. In either the classical or quantum case it is easy to see that if the ROM model is universal with m writable (qu)bits then it is universal for any $m' \geq m$, so the main interest is in determining the minimal m for which universality holds.

The arguments contained in the following sections depend upon the fact that XOR and conjunction produce a distinguished normal form. In order to define this distinguished normal form, let us first review some propositional logic theory. It is well known that AND and negation are sufficient to express any boolean proposition[15]. Using the three simple

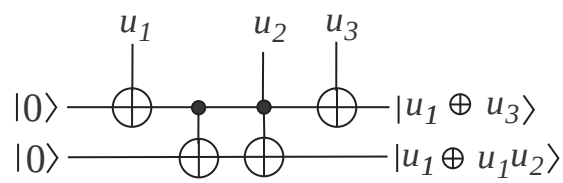

Fig. 1. An example of a ROM-based circuit diagram, the circles containing crosses indicate NOT gates and the black circles indicate controls. The variables at the top of the diagram are the ROM bits.

equivalences,

$$\begin{aligned} 1a &\equiv a \\ \bar{a} &\equiv a \oplus 1 \\ a(b \oplus c) &\equiv ab \oplus ac, \end{aligned} \tag{5}$$

it follows that AND and XOR are also sufficient, as every negated sentence, $\bar{a}$, can be replaced by $a \oplus 1$. This implies that all $2\hat{}(2^j)$ propositions composed of j boolean variables can be expressed as an XOR disjunction of conjunctions, involving no negations. Hence, XOR and AND produce a *normal form.* XOR and AND also produce a *distinguished normal form*, as every expression involving only XOR disjunctions of conjunctions, with no negations, is unique up to transposition of conjunctions[b]. To see that each expression is unique, we note that there are exactly $\binom{j}{k}$ distinct conjunctions involving exactly k of j variables. Thus, the total number of conjunctions is $\sum_{k=0}^{j} \binom{j}{k} = 2^j$. The presence or absence of each of these terms gives the $2\hat{}(2^j)$ different boolean propositions.

To prove that a ROM-based computer is universal, we need to show that each writable (qu)bit can be mapped from 0 to any of the $2\hat{}(2^j)$ different boolean propositions. As every boolean expression can be written as an XOR disjunction of conjunctions, it is sufficient to show that we can transform $|f\rangle$ to $|f \oplus u_1 u_2 \ldots u_m\rangle$ where f is an arbitrary boolean function and $m \in \{1, 2, \ldots, j\}$.

3. One writable qubit is universal

We will now use the Pauli operators,

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad \text{and} \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \tag{6}$$

as well as the operators

$$Z^{\pm\frac{1}{2}} = \begin{pmatrix} 1 & 0 \\ 0 & \pm i \end{pmatrix} \quad \text{and} \quad X^{\pm\frac{1}{2}} = \frac{1}{2}\begin{pmatrix} 1 \pm i & 1 \mp i \\ 1 \mp i & 1 \pm i \end{pmatrix}, \tag{7}$$

[b]For more information on such algebraic forms the reader should see for example MacWilliams and Sloane (p. 371)[16].

to show that a ROM-based quantum computer with one writable qubit is universal. We denote by W_{u_i} an operator W which is applied conditionally on the ROM bit u_i. The sequence of one-qubit gates,

$$X_{u_i}^{-\frac{1}{2}} Z_{u_j} X_{u_i}^{\frac{1}{2}} Z_{u_j} \quad = \quad (iX)_{u_i,u_j} \tag{8}$$

performs a bit flip *if and only if* ROM bits $u_i = u_j = 1$. If the ROM bits were qubits, this would be equivalent to the Toffoli gate construction given in Section VI B of Barenco et al.[11]. Evidently, if both u_i and u_j are zero, no gate is performed, whilst if only one of u_i or u_j is one, then a gate is performed, followed immediately by its inverse, leaving the writable qubit unaltered. However, if both u_i and u_j are one, the sequence of four gates combine to give the Pauli X matrix, which has the effect of flipping the qubit in the computational basis. A circuit diagram for this computation is depicted in Fig. 2(a), whilst Fig. 2(b) uses the Bloch sphere representation[c] to show how a qubit initial in the state $|0\rangle$ is transformed into the state $|1\rangle$ iff $u_i = u_j = 1$. Thus, the sequence in Eq. (8) takes a writable qubit from $|f\rangle$ to $|f \oplus u_i u_j\rangle$.

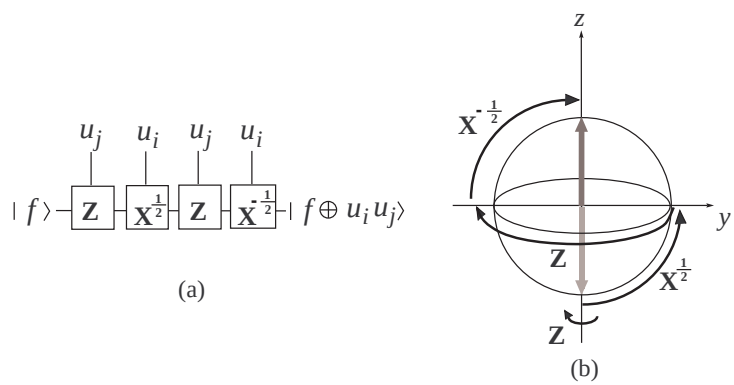

Fig. 2. (a) Circuit diagram of the ROM sequence used to transform $|f\rangle$ to $|f \oplus u_i u_j\rangle$. (b) Bloch sphere representation showing the state $|0\rangle$ transforming to the state $|1\rangle$, when $u_i = u_j = 1$. For all other values of u_i and u_j, $|f\rangle$ remains unchanged.

Now each of the Z_{u_j} terms in Eq. (8) can be replaced by

$$Z_{u_k}^{-\frac{1}{2}} X_{u_j} Z_{u_k}^{\frac{1}{2}} X_{u_j} \quad = \quad iZ_{u_k u_j}, \tag{9}$$

which gives the sequence

$$X_{u_i}^{-\frac{1}{2}} Z_{u_j u_k} X_{u_i}^{\frac{1}{2}} Z_{u_j u_k} \quad = \quad X_{u_i u_j u_k}, \tag{10}$$

ignoring an overall phase factor. This new sequence of gates takes $|f\rangle$ to $|f \oplus u_i u_j u_k\rangle$. By replacing the X_{u_j} terms in Eq. (9) by sequences of the form given in Eq. (8) it is easy to see, by

[c] The pure state of a single qubit can always be represented by a point on a unit sphere, known as the Bloch sphere. For more information see [4] page 15.

recursion, that we can generate a sequence of gates which transforms $|f\rangle$ to $|f \oplus u_1 u_2 \ldots u_m\rangle$. This shows that a ROM-based quantum computer with one writable qubit is universal. We note that this scheme is time inefficient, it requires a number of ROM calls which scales exponentially with the number of ROM bits, however in Section 5 we will introduce a time efficient scheme.

4. Two writable bits are Universal

A ROM-based classical computer with one writeable bit, and the ability to apply a NOT gate conditioned on any fixed number of ROM bits will not be universal. This can be seen as a consequence of Theorem 5.2 from [17], which states that there exist invertible functions of order n which cannot be obtained by composition of generalized Toffoli gates of order strictly less than n.

Now consider a ROM-based classical computer with two writable bits. It is possible to deduce that this will be universal using Lemma 7.3 from Barenco et al. [11]. Here, we show that a two-bit ROM-based classical computer is universal using the four gates depicted in Fig. 3, which perform the transforms

$$|\alpha\rangle|\beta\rangle \xrightarrow{N^{(1)}_{u_i}} |\alpha \oplus u_i\rangle|\beta\rangle \tag{11a}$$

$$|\alpha\rangle|\beta\rangle \xrightarrow{N^{(2)}_{u_i}} |\alpha\rangle|\beta \oplus u_i\rangle \tag{11b}$$

$$|\alpha\rangle|\beta\rangle \xrightarrow{C^{(1)}_{u_i}} |\alpha \oplus u_i\beta\rangle|\beta\rangle \tag{11c}$$

$$|\alpha\rangle|\beta\rangle \xrightarrow{C^{(2)}_{u_i}} |\alpha\rangle|\beta \oplus u_i\alpha\rangle. \tag{11d}$$

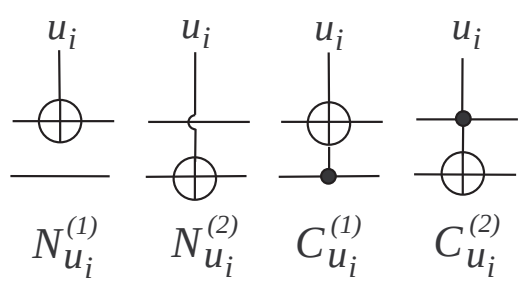

Fig. 3. Circuit diagram representation of the four transforms given in Eq. (11).

We now wish to show, using the four transforms from Eq. (11) that it is possible to transform the writable bits from the state $|\alpha\rangle|\beta\rangle$ to $|\alpha\rangle|\beta \oplus u_1 u_2 \ldots u_m\rangle$. Let us denote by S_0 the gate $N^{(1)}_{u_1}$, which takes $|\alpha\rangle|\beta\rangle$ to $|\alpha \oplus u_1\rangle|\beta\rangle$. It is not hard to show that the sequence

$$S_1 \quad : \quad C^{(2)}_{u_2} S_0 C^{(2)}_{u_2} S_0 \tag{12}$$

performs the transform

$$|\alpha\rangle|\beta\rangle \xrightarrow{S_1} |\alpha\rangle|\beta \oplus u_1 u_2\rangle. \tag{13}$$

Now, suppose we have a sequence of gates, S_{m-1}, which performs the transform

$$|\alpha\rangle|\beta\rangle \quad \xrightarrow{S_{m-1}} \quad |\alpha\rangle|\beta \oplus u_1 u_2 \ldots u_{m-1}\rangle. \tag{14}$$

Then there exists a sequence of gates,

$$S_m \quad : \quad C^{(1)}_{u_m} S_{m-1} C^{(1)}_{u_m} S_{m-1} \tag{15}$$

which perform the transform

$$|\alpha\rangle|\beta\rangle \quad \xrightarrow{S_m} \quad |\alpha \oplus u_1 u_2 \ldots u_m\rangle|\beta\rangle. \tag{16}$$

Therefore a ROM-based classical computer with two writable bits is universal.

5. Time efficiency

A simple counting argument shows that there exists boolean expressions which will require an exponential number of ROM calls on either a quantum or classical ROM computer with a fixed number of writeable (qu)bits. However, it is an open question as to whether there exist specific boolean expressions which can be generated on a one qubit quantum computer using a polynomial number of ROM calls, which require an exponential number of ROM calls on a two bit classical computer. Here, we look at one function in particular, and show that it can be computed efficiently on a one qubit quantum ROM computer. Whether this function can be computed efficiently on a two bit classical ROM computer rests on an unanswered question in classical complexity theory.

Consider the transform

$$|f\rangle \quad \xrightarrow{F} \quad |f \oplus u_1 u_2 \ldots u_j\rangle. \tag{17}$$

Eq. (8) indicates that the transform $|f\rangle \to |f \oplus u_1 u_2\rangle$ can be accomplished using four ROM calls. Now, by making the following replacements,

$$X_{u_1}^{-\frac{1}{2}} \quad \text{with} \quad X_{u_1}^{-\frac{1}{4}} Z_{u_2} X_{u_1}^{\frac{1}{4}} Z_{u_2} \tag{18a}$$

$$X_{u_1}^{\frac{1}{2}} \quad \text{with} \quad X_{u_1}^{\frac{1}{4}} Z_{u_2} X_{u_1}^{-\frac{1}{4}} Z_{u_2} \tag{18b}$$

$$Z_{u_2} \quad \text{with} \quad Z_{u_3}^{-\frac{1}{2}} X_{u_4} Z_{u_3}^{\frac{1}{2}} X_{u_4}, \tag{18c}$$

we can transform $|f\rangle \to |f \oplus u_1 u_2 u_3 u_4\rangle$ using 16 ROM calls. A direct extension of this method, replacing each $X^{\pm 1/2^n}$ by

$$X^{\pm 1/2^{n+1}} Z X^{\mp 1/2^{n+1}} Z, \tag{19}$$

and each $Z^{\pm 1/2^n}$ by

$$Z^{\mp 1/2^{n+1}} X Z^{\pm 1/2^{n+1}} X, \tag{20}$$

allows us to take the AND of up to 2^k ROM bits using exactly 4^k ROM calls. Thus, to take the AND of $O(j)$ ROM bits requires only $O(j^2)$ quantum gates. (Note that if the number of ROM bits is not a power of two we need simply include some dummy ROM bits set equal to 1.)

Now let us consider the classical case. First, suppose that we have a three bit classical ROM computer. We wish to efficiently perform the transform

$$|f\rangle|g\rangle|h\rangle \quad \xrightarrow{F} \quad |f \oplus u_1 u_2 \ldots u_j\rangle|g\rangle|h\rangle. \tag{21}$$

Another way of stating this transform, is that we wish to perform the permutation,

$$(0\ 1)(2\ 3)(4\ 5)(6\ 7) \tag{22}$$

on the eight states of the writable bits, if all the ROM bits are in the state 1, and perform the identity permutation otherwise. The permutation in Eq. (22) can be generated by applying the following five state permutations in succession from left to right,

$$(0\ 1\ 2\ 5\ 4)\ (0\ 4\ 5\ 3\ 2)\ (4\ 5\ 6\ 1\ 0)\ (4\ 0\ 1\ 7\ 6) \tag{23}$$

where each of these permutations is applied if and only if all ROM bits are in the 1 state. Now suppose that ρ is a permutation of 5 states, and F is a Boolean function that can be computed by a depth d Boolean circuit consisting of NOT gates and 2-input, 1-output AND/OR gates. Then Barrington's Theorem [18] states that there is a permutation branching program[d] of length $\leq 4^d$ on 5 states such that:

- it maps every state to itself if $F = 0$.
- it permutes the states according to ρ of $F = 1$.

As the AND of j Boolean variables can be computed by a Boolean circuit of depth $\lceil \log_2 j \rceil$, this theorem, and Eq. (23) indicates that a three bit ROM classical computer can perform F using $O(j^2)$ ROM calls.

A two bit classical ROM computer is equivalent to a 4 state permutation branching program. The power of a 4 state permutation program is unknown[18]. However, here we conjecture that the transform

$$|0\rangle|0\rangle \quad \xrightarrow{F'} \quad |u_1 u_2 \ldots u_j\rangle|0\rangle, \tag{24}$$

requires a number of ROM calls which scales exponentially with j. There is sequence of gates which can perform F' on a two bit classical ROM computer using exactly $R(j)$ ROM calls, where

$$R(j) \quad = \quad R(j-1) + 2^{\lfloor j/2 \rfloor}, \quad R(1) = 1. \tag{25}$$

This is clearly exponential in j, and exhaustive numerical searches for $j < 5$ have shown Eq. (25) to be minimal.

[d] A permutation branching program is the same as the classical ROM model of computation except that the workspace can be in one of k states, and each step of the computation is a permutation of the states conditional on the ROM bit.

6. Discussion

In conclusion, we have introduced a model, which allows the comparison of space-efficiency between error-free, reversible quantum and classical computation. We have shown that quantum computation is more space efficient within this model, requiring only one qubit for universality, as opposed to two bits. We have also conjectured that the minimal quantum ROM computer can calculate certain boolean functions exponentially faster than the minimal classical ROM computer.

It would be interesting to compare the classical and quantum models, allowing for bounded-error computation, that is, the writeable bits are mapped to the correct boolean functions of the ROM bits with some probability $1 - \epsilon$. Preliminary investigations indicate that the quantum model would still be more powerful than the classical model.

References

1. P. W. Shor (1994), *Algorithms for quantum computation: Discrete logarithms and factoring*, Proc. 35th Ann. Symp. on Foundations of Comp. Sc., p. 124.
2. D. Deutsch and R. Jozsa (1992), *Rapid solutions of problems by quantum computation*, Proc. Royal Soc. of London. A, **439**, p. 553.
3. L. K. Grover (1997), *Quantum mechanics helps in seaching for a needle in a haystack*, Phys. Rev. Lett., **79**, p. 325.
4. M. A. Nielsen and I. L. Chuang (2000), *Quantum Computation and Quantum Information*, Cambridge University Press, (Cambridge).
5. J. Preskill (1998), *Quantum Information and Computation*, California Institute of Technology, (Pasadena).
6. C. A. Sackett et al. (2000), *Experimental entanglement of four particles*, Nature, **404**, p. 256.
7. E. Knill et al. (2000), *An algorithmic benchmark for quantum information processing*, Nature, **404**, p. 368.
8. C. H. Papadimitriou (1994), *Computational Complexity*, Addison-Wesley, (Reading).
9. J. Watrous (1999), *Space-bounded quantum complexity*, Journal of Comp. and Sys. Sc., **59**, p. 281.
10. A. Ambainis and R. Freivalds (1998), *1-way quantum finite automata: strengths, weaknesses and generalizations*, Proceedings of FOCS'98, p. 322.
11. A. Barenco et al. (1995), *Elementary gates for quantum computation*, Phys. Rev. A, **52**, p. 3457.
12. R. Landauer (1961), *Irreversibility and Heat Generation in the Computing Process*, IBM Journal of Reasearch and Development, **5**, p. 183.
13. D. P. DiVincenzo (1995), *Two-bit gates are universal for quantum computation*, Phys. Rev. A, **51**, p. 1015.
14. E. Fredkin and T. Toffoli (1982), *Conservative Logic*, International Journal of Theoretical Physics, **21**, p. 219.
15. D. Hilbert and W. Ackermann (1950), *Principles of Mathematical Logic*, Chelsea Publishing Co., (USA).
16. F. J. MacWilliams and N. J. A. Sloane (1977), *The Theory of Error-correcting Codes*, North-Holland Publishing Co., (Amsterdam).
17. T. Toffoli (1980), *Reversible Computing*, p. 632 in Automata, Languages and Programming (1980), edited by J. W. de Bakker and J. van Leeuwen.
18. D. A. Barrington (1989), *Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1*, Journal of Comp. and Sys. Sc., **38**, p. 150.