Efficient quantum algorithm for factorization

Jarosław A. Miszczak
Institute of Theoretical and Applied Informatics (IITiS)
Polish Academy of Sciences
Gliwice, ul. Bałtycka 5, Poland

Abstract

This article presents an algorithm for fast factorization on quantum computer. Quantum mechanic introduces to information processing new effects that can be used to solve efficiently problems classical known as hard. Simon algorithm was one of the first proofs that quantum computer has some advantages over classical model of computation. Shor generalized ideas introduce by Simon and show that real world problems are in range of quantum computation. Article presents also general description of quantum Fourier transform (QFT), which is without doubt the most powerful tool used in algorithms described in this paper. Generalization of algorithms for Simon problem and fast factorization gives new class of problems for which fast quantum algorithms can be construct.

1. Introduction

First quantum algorithms [1] – Deutsch algorithm or in generalized form Deutch-Jozsa algorithm – demonstrate that quantum computer can potentially solve some problems faster than classical computer. These algorithms where build to solve some special problems (see [1] for details), which have no real-world applications. On the other hand it turns out that those early algorithms can be deduced from Simon and Shor results.

Main purpose of this work is presentation of general scheme of fast factorization algorithm. In section 2 first algorithm of Shor-type – Simon algorithm – is described. In addition, quantum Fourier transform used in both algorithms is presented. Section 3 consists of presentation of factoring algorithm. This algorithm is very similar to another one developed also by Shor, used for discrete logarithms computation [3]. We show how factorization problem can be reduced to order finding problem. Section 4 deals with main problem of quantum information theory – physical implementations of theoretical results – and we give short description of implementation of Shor's algorithm. In section 5, we present RSA cryptosystem and simple scheme for its cryptanalysis with quantum computer. If implemented for large numbers Shor algorithm would make this and some other public key cryptosystem useless.

2. Hidden subgroup and fast factorization problems

Every integer can be represented uniquely as a product of prime numbers. The art of factorization is almost as old as mathematics itself. However, the study of fast algorithms for factoring is only a few decades old. In 1994, Peter W. Shor in his paper [1] presented a polynomial-time algorithm for finding the order of integer in Z_n on quantum computer. This algorithm can be used to solve two problems, which are believed to be hard. In other words, there are not known any efficient classical algorithms for those problems. Shor's algorithm is an improved version of Simon's algorithm for finding the XOR mask invariance for some function $f : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$. Simon's algorithm and Shor's algorithm are both based on quantum Fourier transform and they solving problems very similar from the mathematical point of

view. Simon's problem can be generalized on any abelian group, not only Z_{2^m} [5], and this generalized problem is known as hidden subgroup problem.

2.1. Simon's algorithm

Daniel R. Simon in 1994 in his paper [3] presented an algorithm for quantum Turing machine[3] (QTM) that can solve XOR mask invariance problem: for a given Boolean function $f : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$ find – if there exists – a non-zero string $s \in \{0,1\}^n$ such that for all $x \neq y$, $f(x) = f(y)$ iff $x = y \oplus s$. Simon gave an algorithm that can find s in expected time $O(nT_f(n) + G(n))$ where $T_f(n)$ is the time required to compute f and $G(n)$ is the time required to solve n linear equation over Z_2.

Detailed description and discussion of this algorithm can be found in Simon's original work. This algorithm uses quantum Fourier transform over Z_{2^m} to extract information about function f. As we will see, this is the main task of order finding algorithm.

2.2. Quantum Fourier transform

This section shortly describes the most important part of many quantum algorithms – quantum Fourier transform. Construction of QFT is based on some simple facts about quantum information theory. An introduction to quantum computation can be found, for example, in work [4].

Quantum computer is a quantum system. Description of this system is based on Hilbert space H and evolution – i.e. computation – is described by unitary matrices. Let $\{|a_0\rangle,\ldots,|a_n\rangle\}$ be a basis in H. Every quantum state (i.e. vector in H) can be represented by its coefficients $\{c_0,\ldots,c_n\}$[4] in this basis.

$$|\psi\rangle = \sum_{i=0}^{n-1} c_i |a_i\rangle \tag{1}$$

Quantum Fourier transform is a discrete Fourier transform of (real) function assigning coefficients to vectors from the basis. Every such function defines the state in unique way. For a base state $|a\rangle$ we have

$$\mathrm{QFT}|a\rangle = \sum_{y=0}^{n-1} e^{-\frac{2\pi i x y}{n}} |y\rangle \tag{2}$$

This equation allows us to compute QFT for any state – every state is in general linear combination ob base states. QFT on group Z_n is equivalent to multiplication of the state vector by $n \times n$ matrix.

In the simplest case of Z_2 quantum Fourier transform is represented by 2×2 matrix

$$U_H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{3}$$

This transformation is called Walsh-Hadamard transformation or Hadamard gate.

---

[3] Quantum Turing machine is an extension of classical Turing machine, which is ruled by laws of quantum physics. See [5].

[4] In general, coefficients can be complex numbers, but for the purpose of the quantum information theory, we can use real number without loosing generality.

Quantum Fourier transform in Z_{2^m} can be done on quantum computer in time $O(m^2)$. Its classical counterpart – discrete Fourier transform – can be implemented using FFT algorithm in time $O(m \log m)$.

3. Shor's algorithm

Shor's algorithm proved that quantum computer could be used to solve real problems. In [3] two algorithms are presented: for fast factorization and for discrete logarithms computation. Instead of giving a quantum algorithm for factorization Shor presents an algorithm for finding the order of an element a in multiplicative group Z_n. The two main components of this algorithm: modular exponentiation and the inverse quantum Fourier transform (QFT) take only $O((\log n)^3)$ operations on quantum machine. Classically factorization takes an exponential number of operations, which makes this problem intractable as n increases. The best classical algorithm for large integers factorization is number field sieve algorithm, which run in expected time $O(\exp(c(\log n)^{1/3}(\log\log n)^{2/3}))$ for some constant c. For an integer n, $\log n$ is its length in bits, so classical algorithm need an exponential time to factorize n. Classical algorithm for finding discrete logarithms in a modification of number filed sieve algorithm.

3.1. Modular exponentiation

To perform order finding algorithm we must be able to compute for a given $a, n\ a \le n$ and $m \le n$ modular exponential i.e. $m^a \pmod n$. On classical computer this can be done in expected time $O((\log n)^2(\log\log n)(\log\log\log n))$. On quantum computer this stem must be – like any othe – reversible. Technique for compute $m^a \pmod n$ on quantum computer is the same as in classical case. First we compute $m^{2^i} \pmod n$ for all $i < \log n$ and then multiply the powers of $m^{2^i} \pmod n$ where 2^i appears in the binary expansion of a.
Construction of reversible gate array for this operation is given in [1]. This gate can compute $m^a \pmod n$ in expected time $O((\log n)^3)$.

3.2. Order finding and factorization

In this section, we present how to construct efficient algorithm for factorization using Shor's algorithm for order finding [3].
Our input is an integer n. Let $n = p_1^{x_1} \dots p_k^{x_k}$ where $k \ge 2$. We can efficiently check if $k = 1$, so w may assume that n is not a prime power. This means that the Shor's algorithm fails for n even or a prime power.

The first step of our algorithm is to choose at random an integer $a \in Z_n$. Using Euclid's algorithm we can decide if $d = \gcd(a, n) = 1$. If $d = 1$ then is a factor of n. Now using Shor's algorithm we can find an order $\mathrm{ord}_n(a)$.
Let $r = \mathrm{ord}_n(a)$. According to definition[5] $a^r = 1 \pmod n$. If r is even we can write

$$(a^{\frac{r}{2}} - 1)(a^{\frac{r}{2}} + 1) = 0 \pmod n \qquad (4)$$

[5] $\mathrm{ord}_n(a) = \min\{r \in Z \mid a^r = 1 \pmod n\}$

So n divides $a^r - 1$ and it must have common factors with $a^{\frac{r}{2}} - 1$ and $a^{\frac{r}{2}} + 1$. Again we can use Euclid's algorithm to find $g_1 = \gcd(n, a^{\frac{r}{2}} - 1)$ and $g_2 = \gcd(n, a^{\frac{r}{2}} + 1)$. We can extract factors of n when we know $g_1 = \gcd(n, a^{\frac{r}{2}} - 1)$. From the definition of an order n cannot divide $a^{\frac{r}{2}} - 1$, because this would imply that $a^{\frac{r}{2}} = 1 (\mathrm{mod}\, n)$ what is imposisible by the definition of order. In [8] one can find proof that, with a chosen at random, the probability that the $r = \mathrm{ord}_n(a)$ is even and that $a^{\frac{r}{2}} \neq -1 (\mathrm{mod}\, n)$ is at least $\frac{9}{16}$. This result is base on two lemmas, which are given here without proofs.
Let k is a number of different prime factors of n.

Lemma 1. The probability that $r = \mathrm{ord}_n(a)$ is odd for a uniform chosen a is at most $\frac{1}{2^k}$.

Lemma 2. If $a^{\frac{r}{2}} = 1 (\mathrm{mod}\, n)$ is even, then the probability that $a^{\frac{r}{2}} = -1 (\mathrm{mod}\, n)$ is at most $\frac{1}{2^k}$.

3.3. Finding the order

Algorithm for order finding is based on quantum computer ability for fast computation of Fourier transform. Problem of finding the order on an integer $a \in Z_n$ is equivalent to finding the period of function $f : Z_n \to Z_n$ defined as follows:

$$f(x) = a^x (\mathrm{mod}\, n) \tag{5}$$

Fourier transform in its classical version can be use for extracting information about the period of the function. Quantum version of this transformation can also be used for this purpose. QFT is also unitary transformation [4] so it can be easy implemented on quantum computer.
By the definition the order of $a \in Z_n$ is the smallest number $\mathrm{ord}_n(a) = r$ such that $a^r = a (\mathrm{mod}\, n)$. For every $l \in Z$, $a^{x+lr} = a^x a^{lr} = a^x (\mathrm{mod}\, n)$, so the function (2) has period r.
States of quantum computer can't represent every integer and that is why we can't compute Fourier transform on Z. We must choose sum subset $Z_m = \{0,1,\ldots,m-1\}$ as a domain. Integer $m > n$ must be large enough for period to appear. It is also reasonable to choose $m = 2^k$ because of natural representation of Z_{2^k} by k qubits.
Here is an algorithm presented by Shor in [1]:

1. Start with the state $|0\rangle|0\rangle$.
2. Using Walsh-Hadamard on the first register transform prepare superposition
$$\frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |k\rangle|0\rangle \tag{6}$$
3. In the second register compute $a^k (\mathrm{mod}\, n)$ (modular exponentiation).
$$\frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |k\rangle|a^k (\mathrm{mod}\, n)\rangle \tag{7}$$
4. Perform inverse QFT on the first register to extract information about period

$$\frac{1}{\sqrt{m}}\sum_{k=0}^{m-1}\left(\frac{1}{\sqrt{m}}\sum_{p=0}^{m-1}e^{\frac{2\pi ipk}{m}}\left|p\right\rangle\right)\left|a^k \pmod n\right\rangle = \frac{1}{m}\sum_{k=0}^{m-1}\sum_{p=0}^{m-1}e^{\frac{2\pi ipk}{m}}\left|p\right\rangle\left|a^k \pmod n\right\rangle \quad (8)$$

Because function has period r we can write this state as

$$\frac{1}{m}\sum_{p=0}^{m-1}\sum_{q=0}^{s}\sum_{l=0}^{r-1}e^{\frac{2\pi ip(qr+l)}{m}}\left|p\right\rangle\left|a^{qr+l} \pmod n\right\rangle$$

where s is the smallest integer such that $qs+l \le m$

5. Observe the state. The probability that the state of the machine is in the state $\left|p\right\rangle\left|a^l \pmod n\right\rangle$ for a fixed p is equal

$$\mathrm{P}(p) = \frac{1}{m^2}\left|\sum_{q=0}^{s}e^{\frac{2\pi ip(qr+l)}{m}}\right|^2$$

If there is a d such that

$$\frac{-r}{2} \le rp - dm \le \frac{r}{2}$$

then the probability of seeing state $\left|p\right\rangle\left|a^l \pmod n\right\rangle$ will be at least $\frac{1}{3r^2}$ for sufficient large n.

More detailed discussion of this algorithm can be found in [1].

The above condition can be rewritten

$$\left|\frac{p}{m} - \frac{d}{r}\right| \le \frac{1}{2m} \quad (9)$$

There is at most one fraction $\frac{d}{r}$ with $r < n$ that satisfies an inequality (9). We can compute this fraction in polynomial time on classical computer by using continued fraction expansion of $\frac{c}{r}$. Rounding $\frac{c}{r}$ to the nearest fraction with a denominator smaller than n we will get $\frac{d}{r}$ in lowest term. If $\gcd(d,r)=1$ this will give us an order r. It can be proofed that the probability that we fin the fraction of this king is equal $\frac{t}{\log\log r}$ for some constant t. So repeating the above procedure $O(\log\log r)$ times, we are assured to get an order with high probability.

3.4. Complexity of Shor's algorithm

Shor's algorithm runs in expected time $O((\log n)^2(\log\log n)(\log\log\log n))$. This include three steps:

1. $O(\log n)$ for Hadamard gates for state preparation
   Hadamard gates can be implemented in linear time.
2. $O((\log n)^2(\log\log n)(\log\log\log n))$ for modular exponentiation
   Complexity of this part of algorithm is equal to complexity of classical algorithm.
3. Quantum Fourier transform take only $O((\log n)^2)$
   As we mention in section 2 QFT is the most important of Shor's algorithm. Without the ability of computing Fourier transform in polynomial time quantum computer wouldn't gave us an ability for factoring integers efficiently.

More information on quantum complexity theory can be found in [11].

3.5. Note on implementation

Nowadays NMR[6] technology gives us the most important device for performing quantum computation. Other devices like trapped ions, atoms and light are also proposed [10]. Experimental realization of Shor's algorithm on NMR quantum computer is presented in [6] (See also [7]). The method of using nuclei to store quantum information is in principle scalable to many quantum bit systems. In [6] authors report an implementation of the simplest instance of Shor's algorithm: factorization of $n = 15$. Quantum bits (qubits [4,10]) where seven spin-$\frac{1}{2}$ molecules[7]. The significance of this experiment lies in the demonstration of techniques for precise control of quantum computers. Main problem for further implementations is control over much larger quantum systems and proper entanglement processing.

4. Some examples

In this section we give two examples of quantum algorithm for fast factorization. Firt using Shor's algorithm we compute prime factors of number 15. This task was realized by NM quantum computer and described above. Second example is connected with cryptography. We consider a cryptanalysis of RSA block cipher. A short description of RSA algorithm is also given.

4.1. Example of factorization

The smallest number that can be factorized using Shor's algorithm is $n = 15$. Let us choose at random an integer $a = 7$, which satisfy condition $\gcd(a,n) = \gcd(7,15) = 1$. The order of 7 is $\mathrm{ord}_{15}(7) = 4$. Fortunately this number is even so we can compute that $7^2 - 1 = 3 (\mathrm{mod}\, 15)$ and $7^2 + 1 = 5 (\mathrm{mod}\, 15)$. Using Euclid's algorithm we check that $\gcd(3,15) = 3$ and $\gcd(5,15) = 5$ so we found nontrivial factors of 15.

4.2. RSA cryptanalysis

RSA cryptosystem was developed by Ron Rivest, Adi Shamir and Leonard Adleman in 1978. Nowadays RSA is the most commonly used public key algorithm. It can be used both for encryption and for digital signatures. The security of RSA is generally considered equivalent to factoring, although this has not been proved. It is also used for key exchange purposes, for example in PGP software.
RSA is a block chipper; it encrypt message in blocks (block by block). Detailed description of this and many others algorithms can be found in [16,17]

4.3. RSA cryptosystem

In this section the algorithm for creating key, encryption and decryption in RSA cryptosystem is described. The signing is the same like decryption and verification is the same like encryption. These three main functions are very simple.

1) Creating keys:

---

[6] NMR = Nuclear Magnetic Resonance

[7] Molecule used as the quantum computer for this experiment contains five $^{19}\mathrm{F}$ and two $^{13}\mathrm{C}$ spin-½ nuclei as qubits

a) Find or generate two large prime numbers (p and q)
b) Calculate $n = pq$ [8]
c) Calculate Euler totient function[9] for n; $m = \phi(pq) = (p-1)(q-1)$
d) Select at random a positive integer $e < m$, such that $\gcd(e,m) = 1$, i.e. co-prime to m
e) Calculate d such that $e \times d = 1 (\mathrm{mod}\, m)$ or equivalently $d = e^{-1} (\mathrm{mod}\, m)$

Pair (e,n) forms public key and the number d is private key. Numbers p and q should be forgotten. The integers e and d are called the encryption exponent and the decryption exponent, respectively, while n is called the modulus.

2) Encryption
   a) Original plain text (a block value) X, $X < n$. Each block X must have unique representation $\mathrm{mod}\, n$.
   b) Chipertext $C = X^e (\mathrm{mod}\, n)$
3) Decryption
   a) Chipertext $= C$
   b) Dechiper text $Y = C^d (\mathrm{mod}\, n)$

   Proof that decryption works can be found in [15]

RSA is currently the most important public key algorithm. The problem for the attacker is that computing the reverse d of e is assumed to be no easier than factorizing n. It is clear that polynomial time algorithm for factorization can be use to attack this algorithm.

4.4. Simple attack on RSA

Efficient algorithm for factoring numbers allows us to decipher message encrypted with RSA algorithm. Let us assume that we have encrypted message M and public key (e,n). To find original message we need to find prime factors of n. This will allow us to find $\phi(n)$ and reproduce private key – that is d. In this step we can also use quantum Fourier transform. Schönhage-Strassen algorithm for fast multiplication uses itself Fourier transform. We could perform Euclid's algorithm to find $d = e^{-1} (\mathrm{mod}\, \phi(n))$ using super-fast multiplication on quantum computer. Next, we can perform decryption on our message M. This step is essentially the same as modular exponentiation used in order finding algorithm.

This leads to simple protocol for RSA cryptanalysis:

1. Input: encrypted message M and public key (e,n)
2. Using Shor's algorithm find the prime factors of n
3. We knew what number was selected in step 1.d) of key generation – it is part of public key, so we need to calculate $\phi(n)$ and find an inverse of e, $d = e^{-1} (\mathrm{mod}\, \phi(n))$
4. The last step is decryption of a message M.

---

[8] The key size (the size of the modulus) should be greater than 1024 bits (i.e. it should be of magnitude 10^{300}) for a reasonable margin of security.

[9] Euler totient function (or Euler phi function) $\phi(n)$ for $n \geq 1$ denotes the number of integers in the interval $[1,n]$, which are relatively prime.

This algorithm is rather out of range of present-day quantum computers because of technical problems we need to deal when implementing Shor's algorithm. Scalable NMR quantum computation requires coherent control over many qubits in the course of a long sequence of controlled interactions, even after maximal reduction of the quantum circuit. In realization of Shor's algorithm interactions between almost all pairs of qubits are needed. To realize real attack on RSA algorithm NRM quantum computer need to operate on more then 1024 qubits. The complexity of apparatus needed for such operation is the best protection for present-day cryptosystems.

5. Conclusions and further work

Algorithms presented in this paper exploit quantum Fourier transform to solve problems known as hidden subgroup problems [9]. All quantum algorithms know at present time can be divided in two classes: Grover's type algorithms [12,13] and hidden subgroup algorithms [10]. Algorithms from second group uses quantum Fourier transform for manipulating quantum states. General formulation of these algorithms can lead us to new quantum algorithms. Other algorithms of that type can be important from complexity theory point of view and can answer if quantum mechanical model of computation is indeed more powerful then classical model.

Another matter which can gave us ability to solve wider class of problems is entanglement and its usage in quantum algorithms. Entanglement is an effect specific for quantum mechanics and it is believe to be one of the most important resources in quantum information processing. It can be also used to ensure secure communication and replace old cryptographic methods with quantum cryptography.

Any classical algorithm can be made reversible and thus implemented on quantum computer. In contrast, not every quantum algorithms can be simulated efficiently on classical probabilistic Turing machine. Simulation of order finding algorithm would give us an efficient solution of factoring problem on classical computer. But the presence of entanglement in Shor's algorithm [17] makes this simulation impossible. This doesn't mean that there is no efficient classical algorithm for factorization, but it shows that the entanglement is important resource in quantum information processing.

Physical implementation of new paradigm of computation and known quantum algorithms will be the best proof that quantum computation can be used to solve problems. Physical realization of quantum computer would give us also a key to new concepts in physics.

Streszczenie

Artykuł prezentuje algorytm szybkiej faktoryzacji na komputerze kwantowym. Mechanika kwantowa wprowadza do informatyki nowe efekty, które pozwalają na efektywne rozwiązywanie problemów klasycznie uważanych za trudne. Algorytm Simona był jednym z pierwszych dowodów na to, że komputer kwantowych posiada pewną przewagę nad klasycznym modelem obliczeń. Shor uogólniając idee Simona pokazał, że komputer kwantowy może służyć do rozwiązywania rzeczywistych problemów takich jak kryptoanaliza systemu RSA. W artykule omówiona jest także kwantowa transformata Fouriera, która jest bez wątpienia najpotężniejszym narzędziem wykorzystywanym w omawianych algorytmach. Uogólnienie kwantowych algorytmów dla problemu Simona i faktoryzacji liczb pierwszych daje nową klasę problemów, dla których możliwa jest konstrukcja szybkich algorytmów kwantowych.

References

[1] David Deutsch, Rapid Solutions of Problems by Quantum Computation, Proc. R. Soc. Lond A 439, 1992, p. 553;

[2] Daniel R. Simon, On the Power of Quantum Computation, Proceedings of the 35th Annual Symposium on Foundations of Computer Science, 1994, pp. 116-123.Available at http://citeseer.nj.nec.com/8731.html

[3] Peter W. Shor, Algorithms for Quantum Computation: Discrete Logarithms and Factoring, IEEE Symposium on Foundations of Computer Science, 1994, pp. 124-134. Available at http://citeseer.nj.nec.com/14533.html.

[4] Sławomir Bugajski, Jerzy Klamka, Stefan Węgrzyn, Foundations of quantum computing. Part I, Archiwum Informatyki Teoretycznej i Stosowanej, 13, 2001, pp. 97-142.

[5] David Deutsch, Quantum theory, the Church-Turing principle and the universal quantum computer, Proceedings of the Royal Society, London, vol. A400, 1985, pp. 97-117.

[6] Lieven M.K. Vandersypen, Matthias Steffen, Gregory Breyta, Costantino S. Yannoni, Mark H. Sherwood, Isaac L. Chuang, Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance, Nature 414, 2001, pp. 883-887. Available at http://www.arxiv.org/abs/quant-ph/0112176

[7] Lieven M.K. Vandersypen, Matthias Steffen, Gregory Breyta, Costantino S. Yannoni, Richard Cleve, Isaac L. Chuang, Experimental Realization of an Order-Finding Algorithm with an NMR Quantum Computer, Phys. Rev. Lett. 85, 25, pp. 5452-5455. Available at http://www.arxiv.org/abs/quant-ph/0007017

[8] Mika Hirvensalo, Quantum Computing, Springer-Verlag, 2001.

[9] David Deutsch, Quantum Theory, the Church-Turing Principle, and the Universal Quantum Computer, Proceedings of the Royal Society of London, vol. A400, 1985, pp. 97-117.

[10] John Preskill, Lecture notes on physics: quantum compuation. Available at http://www.theory.caltech.edu/people/preskill/ph229/

[11] Ethan Bernstein and Umesh Vazirani, Quantum Complexity Theory, Proceedings of the 25th Annual ACM Symposium on the Theory of Computing, 1993, pp. 11-20. Available at http://citeseer.nj.nec.com/bernstein97quantum.html

[12] Lov K. Grover, Quantum Mechanics Helps in Searching for a Needle in a Haystack, Phys. Rev. Lett. vol. 79, 325, (1997).
Available at http://xxx.lanl.gov/abs/quant-ph/9706033.

[13] Richard Jozsa, Searching in Grover's Search Algorithm. Available at http://xxx.lanl.gov/abs/quant-ph/9701021.

[14] Samuel J. Lomonaco, Jr., Louis H. Kauffman, Quantum hidden subgroup algorithms: a mathematical perspective. Available at http://xxx.lanl.gov/abs/quant-ph/0201095.

[15] Richard Jozsa, Quantum factoring, discrete logarithms and the hidden subgroup problem, Available at http://xxx.lanl.gov/abs/quant-ph/0012084.

[16] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, Handbook of Applied Cryptography. Available at http://www.cacr.math.uwaterloo.ca/hac/.

[17] Bruce Schneier, Kryptografia dla praktyków, WNT, 1996.

[18] Richard Jozsa, Noan Linden, On the role of entanglement in quantum computational speedup. Available at http://xxx.lanl.gov/abs/quant-ph/0201143.