数理解析研究所講究録 1295 巻 2002 年 69-74

Factoring Algorithms Based on NMR Quantum Computers

國廣 昇 (Noboru Kunihiro)、山下 茂 (Shigeru Yamashita)
NTT コミュニケーション科学基礎研究所, NTT

Abstract

No polynomial time algorithms have been proposed for the factoring and discrete logarithm problems. However, Shor showed that these problems can be solved by a quantum Turing machine in 1994. Several devices have been proposed toward the realization of quantum computers. Among them, the NMR quantum computer seems to be the nearest to the target goal. Since the NMR quantum computer has different features from ordinary ones, we cannot directly implement Shor's algorithms. In this paper, we propose new simple algorithms that work on NMR quantum computers to solve the factoring and discrete logarithm problems.

1 Introduction

The security of the RSA cryptosystems [1] is based on the difficulty of factoring a large composite integer. Many algorithms for the factoring have been proposed. However, there are no polynomial time algorithms for the factoring problems. Even the best algorithm needs sub–exponential time.

In 1994, Shor showed that the factoring and discrete logarithm problems are solvable in probabilistic polynomial time by using quantum computers [2]. His algorithms utilize quantum parallelism and quantum observation. Some quantum algorithms have been proposed and many experiments also have been done toward the realization of quantum computers. The nuclear magnetic resonance (NMR) seems to be the nearest to the realization.

In this paper, we propose simple and efficient algorithms for the factoring and discrete logarithm problems that use NMR quantum computers. These algorithms output the correct answers in deterministic polynomial time.

2 Preliminaries

We describe quantum algorithms and introduce the bulk quantum Turing machine (BQTM), which is a computational model of NMR quantum computers. As described later, the NMR quantum computers are used to solve the counting problem and the problem of obtaining an approximate value of counting.

2.1 Quantum Algorithms

Many quantum algorithms have been proposed since Deutsch's proposal of quantum Turing machines in 1985 [3]. One of the most famous algorithms is Shor's factoring algorithm [2]. This algorithm and its

variant solve the factoring and discrete logarithm problems on a quantum Turing machine in probabilistic polynomial time. Since then, many experiments have also been done toward the realization of quantum computers. One of the most famous experiments was conducted by Chuang et al. in 2000 [4], who found that an NMR quantum computer with 5 bits solved the order finding problem.

The NMR quantum computer is different from an ordinary quantum computer with respect to the phase of measurement. In an ordinary quantum computer, if we observe the superposition of states, we will find a pure state with probability depending on amplitudes. On the other hand, measurement in NMR quantum computers has two features: (i) it does not collapse the superposition of states and the outcome of the measurement does not become a pure state. But (ii) we can observe a value depending on amplitudes. Since the outcome of the measurement does not become a pure state, we cannot directly implement Shor's algorithms, which use another measurement in addition to a final measurement.

In this paper, we propose two algorithms that use NMR quantum computers with feature (ii). One is for the factoring problem and the other is for the discrete logarithm problem. These algorithms are much different from Shor's algorithms.

2.2 Bulk Quantum Turing Machine

Nishino et al. [5] introduced a *bulk quantum Turing machine* as a computational model of NMR quantum computers.

Definition 1 (Bulk Quantum Turing Machine) A bulk quantum Turing machine (BQTM) is a 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$, where Q is a finite set of states, Γ is a tape alphabet, $B \in \Gamma$ is a blank symbol, $\Sigma \subseteq \Gamma$ is an input alphabet, δ is a state transition function and a mapping from $Q \times \Gamma \times \Gamma \times Q \times \{L, R\}$ to $\mathbf{C}$, $q_0 \in Q$ is an initial state, and $F \subseteq Q$ is a set of final states. If a tape cell of a BQTM is in a superposition $\alpha|0\rangle + \beta|1\rangle$, the real number $|\beta|^2 - |\alpha|^2$ will be measured. In practice, $|\beta|^2 - |\alpha|^2$ will be determined to the accuracy of $1/2^{k-1}$, where k is a positive integer and is called *measurement accuracy.*

Remark 1 Let θ be the value obtained by measurements. It holds that

$$(|\beta|^2 - |\alpha|^2) - \frac{1}{2^{k-1}} < \theta < (|\beta|^2 - |\alpha|^2) + \frac{1}{2^{k-1}}.$$

Next, we define the *counting problem.* This problem is formulated as follows.

Definition 2 (Counting Problem) Given a function $f : \{0,1\}^N \to \{0,1\}$, the counting problem of f is the problem of counting $|f^{-1}(1)|$, i.e., $\sum_{x_1,\ldots,x_N} f(x_1, \ldots, x_N)$, or the number of assignments such that $f = 1$.

If $k = N$, the following algorithm solves the counting problem of f on BQTM.

Step 1 Apply the Walsh–Hadamard Transformation to $|0, 0, \ldots, 0\rangle|0\rangle$, which changes the state as follows.

$$|0, 0, \ldots, 0\rangle|0\rangle \to \frac{1}{\sqrt{2^N}} \sum_{x_1,\ldots,x_N} |x_1, \ldots, x_N\rangle|0\rangle$$

Step 2 Compute $f(x_1, x_2, \ldots, x_N)$ for all $x_1, \ldots, x_N$ and store the result in the last register.

$$\frac{1}{\sqrt{2^N}} \sum_{x_1,\ldots,x_N} |x_1, \ldots, x_N\rangle|0\rangle \to \frac{1}{\sqrt{2^N}} \sum_{x_1,\ldots,x_N} |x_1, \ldots, x_N\rangle|f(x_1, x_2, \ldots, x_N)\rangle$$

Step 3 Perform a measurement on the last register. Let θ be the value obtained. The number of assignments satisfying $f = 1$ is $2^{N-1}(1+\theta)$.

Let A be a set of assignments satisfying $f = 1$ and $|A|$ be the number of elements in A. From Definition 1, $|A|$ and θ satisfy the following inequality. $\left(\frac{|A|}{2^N} - \frac{2^N-|A|}{2^N}\right) - \frac{1}{2^{k-1}} < \theta < \left(\frac{|A|}{2^N} - \frac{2^N-|A|}{2^N}\right) + \frac{1}{2^{k-1}}$. From the above inequality, it holds that

$$2^{N-1}(1+\theta) - 2^{N-k} < |A| < 2^{N-1}(1+\theta) + 2^{N-k}.$$

Since $k = N$ and $|A|$ is an integer, $|A|$ becomes exactly $2^{N-1}(1+\theta)$.

Next, suppose that an accurate measurement cannot be executed (i.e. $k < N$). From the above inequality, we can find that the correct value $|A|$ lies between $2^{N-1}(1+\theta) - 2^{N-k}$ and $2^{N-1}(1+\theta) + 2^{N-k}$.

Remark 2 Assume that $k = N - c\log N$, where c is a positive constant. In this case, by slightly modifying the above algorithm, we can also solve the counting problem. If we can solve the counting problem, we can also solve the NP-complete problem. In other words, if $k \geq N - c\log N$, we can solve the NP-complete problem.

Remark 3 The measurement accuracy k is determined by physical devices. For example, k depends on the number of molecules used in NMR quantum computers.

We propose two algorithms that use NMR quantum computers. These algorithms consist of three steps. First, a function f is set. This function depends on the problems to be solved. Second, the counting problem of the function f is solved by using the output of the NMR quantum computer. Finally, the problems are solved from the result obtained in the second step by classical computers. One algorithm is for the factoring problem. The other is for the discrete logarithm problem. In these algorithms, we need not assume accurate measurement. We just assume that $k = N - c\log_2 N$, where c is a positive constant.

3 Algorithm for Factoring

First, we define the Euler ϕ function.

Definition 3 Let n be $p_1^{e_1}p_2^{e_2}\cdots p_l^{e_l}$, where $p_1, \ldots, p_l$ are distinct primes and $e_1, \ldots, e_l$ are positive integers. $\phi(n)$ is defined as follows.

$$\phi(n) = p_1^{e_1-1}(p_1-1)p_2^{e_2-1}(p_2-1)\cdots p_l^{e_l-1}(p_l-1).$$

For $\phi(n)$ and factoring n, the following two facts are widely known [6][7].

Fact 1 Let n be a composite number. If we know $\phi(n)$, we can factorize n in *probabilistic* polynomial time. Furthermore, assuming that Extended Riemann Hypothesis is true, if we know $\phi(n)$, we can factorize n in *deterministic* polynomial time.

Fact 2 For composite n, the product of two distinct odd primes p_1 and p_2, if we know $\phi(n)$, we can factorize n in *deterministic* polynomial time.

Proof of Fact 2. From Definition 3, $\phi(n) = (p_1 - 1)(p_2 - 1)$. Hence, $p_1 + p_2 = n + 1 - \phi(n)$. Thus, p_1 and p_2 can be determined by solving the quadratic equation $x^2 - (n + 1 - \phi(n))x + n = 0$ over a real field **R**. This leads to

$$p_1,\ p_2 = \frac{n + 1 - \phi(n) \pm \sqrt{(n + 1 - \phi(n))^2 - 4n}}{2}.$$

This can be calculated in deterministic polynomial time. ∎

Facts 1 and 2 tell that it is enough to obtain $\phi(n)$ for factoring the composite n. Our algorithm utilizes this feature.

Remark 4 Shor's algorithm also uses this feature in a different manner. It obtains r satisfying $x^r \equiv 1 (\bmod n)$ and factorizes n by using r. Note that r is a divisor of $\phi(n)$. Furthermore, note that if we know r instead of $\phi(n)$, we can also factorize n in polynomial time.

Next, we define the function $f : \{0,1\}^N \to \{0,1\}$ as follows, where $N = \lceil \log_2 n \rceil$.

$$\begin{cases} f(x_1, \ldots, x_N) = 1 & \text{if } \gcd((x_1 x_2 \cdots x_N)_2, n) = 1 \text{ and } (x_1 x_2 \cdots x_N)_2 < n, \\ f(x_1, \ldots, x_N) = 0 & \text{otherwise,} \end{cases}$$

where $(x_1 x_2 \cdots x_N)_2$ means $\sum_{i=1}^{N} 2^{N-i} x_i$. In other words, $x_1 x_2 \cdots x_N$ is a binary representation of an integer $(x_1 x_2 \cdots x_N)_2$. Note that $\sum_{x_1,\ldots,x_N} f(x_1, x_2, \ldots, x_N) = \phi(n)$.

Next, we describe the algorithm for the factoring problem. In this algorithm, we assume that the measurement accuracy $k = N - c \log_2 N$, where c is a positive constant.

Algorithm for factoring

Step 1 Apply the Walsh–Hadamard Transformation to $|0, 0, \ldots, 0\rangle|0\rangle$, which changes the state as follows.

$$|0, 0, \ldots, 0\rangle|0\rangle \to \frac{1}{\sqrt{2^N}} \sum_{x_1,\ldots,x_N} |x_1, \ldots, x_N\rangle|0\rangle.$$

Step 2 Compute $f(x_1, x_2, \ldots, x_N)$ for all $x_1, \ldots, x_N$ and store the result in the last register.

$$\frac{1}{\sqrt{2^N}} \sum_{x_1,\ldots,x_N} |x_1, \ldots, x_N\rangle|0\rangle \to \frac{1}{\sqrt{2^N}} \sum_{x_1,\ldots,x_N} |x_1, \ldots, x_N\rangle|f(x_1, \ldots, x_N)\rangle.$$

Step 3 Perform a measurement on the last register. Let θ be the value obtained. The value $2^{N-1}(1 + \theta)$ becomes the approximate value of $\phi(n)$. Note that

$$2^{N-1}(1 + \theta) - N^c < \phi(n) < 2^{N-1}(1 + \theta) + N^c.$$

Step 4 Try to factorize n by setting $\phi(n) = 2^{N-1}(1 + \theta) + t$, where $-N^c < t < N^c$. If prime factors are obtained, output these factors. Otherwise, change t. By repeating at most $2N^c - 1$, we can obtain correct prime factors.

Let us now estimate the computation amount of the above algorithm. The computation amount for computing $f(x_1, \ldots, x_N)$ at Step 2 is $O(N^3)$. The computation amount for obtaining prime factors of n from $\phi(n)$ at Step 4 is $O(N^3)$. The average and worst numbers of repetitions are $O(N^c)$. Hence, the total computation amount is $O(N^{3+c})$. So, the factoring is executed in deterministic polynomial time.

Let us look at a numerical example. Suppose that $n = 15$. Hence N becomes 4.

At Step 1, the initial state changes as $|0,0,0,0\rangle|0\rangle \rightarrow \frac{1}{\sqrt{2^4}} \sum_{x_1,x_2,x_3,x_4} |x_1,x_2,x_3,x_4\rangle|0\rangle$. At Step 2, the states change as follows.

$$\frac{1}{\sqrt{2^4}} \sum_{x_1,x_2,x_3,x_4} |x_1,x_2,x_3,x_4\rangle|0\rangle \rightarrow \frac{1}{\sqrt{2^4}} \sum_{x_1,x_2,x_3,x_4} |x_1,x_2,x_3,x_4\rangle|f(x_1,x_2,x_3,x_4)\rangle.$$

The assignments satisfying $f(x_1,x_2,x_3,x_4) = 1$ are

$$(x_1,x_2,x_3,x_4) = \{(0,0,0,1),(0,0,1,0),(0,1,0,0),$$
$$(0,1,1,1),(1,0,0,0),(1,0,1,1),(1,1,0,1),(1,1,1,0)\}.$$

Let A be a set of assignments satisfying $f = 1$ and $\bar{A}$ be the complementary set of A: $\bar{A} = \{0,1\}^4 - A$. In this case, it holds that $|A| = 8$ and $|\bar{A}| = 8$. Hence, we can rewrite the above states as

$$\frac{1}{\sqrt{2^4}} \sum_{(x_1,\ldots,x_4)\in A} |x_1,\ldots,x_4\rangle|1\rangle + \frac{1}{\sqrt{2^4}} \sum_{(x_1,\ldots,x_4)\in \bar{A}} |x_1,\ldots,x_4\rangle|0\rangle.$$

At Step 3, the last register is measured on. Let θ be the value obtained. Assume that the measurement is executed with arbitrary accuracy. Then,

$$\theta = \frac{|A|}{2^4} - \frac{|\bar{A}|}{2^4} = \frac{8}{16} - \frac{8}{16} = \frac{0}{16} = (0.000)_2.$$

Hence, $\phi(n)$ becomes $\phi(n) = 2^{4-1}(1-0) = 8$.

Remark 5 If n is known as the product of two distinct primes in advance, n can be easily factorized as shown in the proof of Fact 2. All we should do is to solve the equation: $x^2 - 8x + 15 = 0$. By solving this equation, we obtain the prime factorization of n : $n = 3 \times 5$.

4 Algorithm for the Discrete Logarithm Problem

First, we formulate the discrete logarithm problem.

Discrete Logarithm Problem Given a prime p, g and $a \in \mathbf{F}_p$, find a positive smallest integer s such that $g^s \equiv a (\bmod p)$. Let q be the order of the cyclic group generated by g.

As described later, we can count the number of integers x $(0 \leq x \leq 2^N - 1)$, which is called $M(w)$, such that $g^x \equiv a \cdot g^{-w} (\bmod p)$ for given integers w and N by using the NMR quantum computer. We propose an algorithm that solves the discrete logarithm problem by invoking it as a subroutine at most $\log q$ times. In this algorithm, we assume that $k = N$.

Algorithm for the discrete logarithm problem

Step 1 Find N such that $\frac{2^N}{q} - C = \mathrm{O}(1)$ and $C + 1 - \frac{2^N}{q} = \mathrm{O}(1)$, where $C = \left\lfloor \frac{2^N}{q} \right\rfloor$. We can find an integer N for arbitrary q.

Step 2 Find w such that $M(w) = C + 1$ and $M(w+1) = C$ by *binary search*. $M(w)$ can be calculated by invoking the *algorithm for calculating* $M(w)$. We can find such w by invoking this algorithm at most $\log q$ times.

Step 3 Set $s = w$.

We omit the details of the algorithm for calculating $M(w)$.

We show the process of finding s.

Example. Set $p = 97, g = 5, a = 74, q = 96, N = 7$ and $2^N = 128$. In this case, the solution is $s = 29$.

We can find s by executing binary search: $M(0) = 2 \rightarrow M(48) = 1 \rightarrow M(24) = 2 \rightarrow M(36) = 1 \rightarrow M(30) = 1 \rightarrow M(27) = 2 \rightarrow M(29) = 2$.

We find $M(29) = 2$ and $M(30) = 1$. Hence, we obtain the solution $s = 29$. In the above process, we solve the discrete logarithm problem by invoking an NMR quantum computer seven times.

5 Conclusion

We proposed simple NMR–quantum–computer–based algorithms for the factoring, discrete logarithm problems. Our future work is to find an efficient algorithm that solves the problems when the measurement accuracy k is much less than the problem size N.

References

[1] R.L. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signature and public key cryptosystems," *Comm. of ACM*, vol.21 no. 2, pp.120–126, 1978.

[2] P.W. Shor, "Algorithms for Quantum Computation: Discrete Log and Factoring," in Proc. of the 35th FOCS, pp. 124–134, 1994.

[3] D. Deutsch, "Quantum Theory, the Church–Turing Principle and the Universal Quantum Computer," Proc. R. Soc. Lond., pp. 97–117, 1985.

[4] L. M.K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoi, R. Cleve and I. L. Chuang, "Experimental realization of order-finding with a quantum computer," Phys. Rev. Lett. 85, pp. 1054–1057, 2000.

[5] T. Nishino, H. Shibata, K. Atsumi and T. Shima, "Solving Function Problems and NP-complete Problems by NMR Quantum Computation," Technical Report of IEICE, COMP 98-71, 1998.

[6] G.L. Miller, "Riemann's hypothesis and test for primality," Journal of Computer and System Sciences, vol. 13, pp. 300–317, 1976.

[7] N. Kunihiro and K. Koyama, "Equivalent of Counting the Number of Points on Elliptic Curve over the Ring Z_n and Factoring n," Proc. of Eurocrypt'98, LNCS 1403, Springer-Verlag, pp. 47–58, 1998.