A Note on Shor's Quantum Algorithm for Prime Factorization

Zhengjun Cao

Institute of System Science, Chinese Academy of Sciences.

Beijing, P.R. China. zjcamss@hotmail.com

Abstract It's well known that Shor[1] proposed a polynomial time algorithm for prime factorization by using quantum computers. For a given number n, he gave an algorithm for finding the order r of an element x (mod n) instead of giving an algorithm for factoring n directly. The indirect algorithm is feasible because factorization can be reduced to finding the order of an element by using randomization[2]. But a point should be stressed that the order of the number must be even. Actually, the restriction can be removed in a particular case. In this paper, we show that factoring RSA modulus (a product of two primes) only needs to find the order of 2, whether it is even or not.

Keywords Shor's quantum algorithm, RSA modulus.

1 Introduction

Factoring integers is generally thought to be hard on a classical computer. But it is now hold that prime factorization can be accomplished in polynomial time on a quantum computer. This remarkable work is due to Peter W. Shor[1]. For a given number n, he gave a quantum computer algorithm for finding the order r of an element x (mod n) instead of giving a quantum computer algorithm for factoring n directly. The indirect algorithm is feasible because factorization can be reduced to finding the order of an element by using randomization[2]. We now briefly give this reduction.

To find a factor of an odd number n, given a method for computing the order r of x, choose a random x (mod n), find its order r, and compute $\gcd(x^{r/2}-1, n)$. The Euclidean algorithm[3] can be used to compute $\gcd(x^{r/2}-1, n)$ in polynomial time. Since $(x^{r/2}-1)(x^{r/2}+1) = x^r - 1 \equiv 0(mod\ n)$, the numbers $\gcd(x^{r/2}-1, n)$ and $\gcd(x^{r/2}+1, n)$ will be two factor of n. This procedure fails only if r is odd, in which case $r/2$ is not integral, or if $x^{r/2} \equiv -1(mod\ n)$, in which case the procedure yields the trivial factors 1 and n. Using this criterion, it can be shown that this procedure, when applied to a random $x(mod\ n)$, yields a nontrivial factor of n with probability at least $1 - 1/2^{k-1}$, where k is the number of distinct odd prime factors of n. Refer to [1] for a brief sketch of the proof of this result.

One phenomena might be observed that existing prime factorization algorithms[4, 5, 6, 7, 8, 9] as well as Shor's quantum algorithm all aim to factor arbitrary numbers. No algorithm pays more attentions to some numbers of particular structure, for instance, product of two primes. But those numbers are of great importance in public key cryptography. They are usually called RSA modulus. In this paper, we give a new algorithm to factor a product of two primes based on Shor's quantum algorithm, which takes advantage of the special structure. we show that factoring RSA modulus only needs to find the order of 2, whether it is even or not.

2 Preliminary

Let $N = pq$ be a product of two odd primes, $\Phi(N)$ be Euler Totient Function. We know

$$\Phi(N) = (p-1)(q-1) = pq - p - q + 1$$

and

$$N - \Phi(N) + 1 = p + q$$

Considering the following equation

$$x^2 - Mx + N = 0 \tag{$*$}$$

where M is undetermined. Hence, we obtain two roots

$$x_1 = \frac{M + \sqrt{M^2 - 4N}}{2}, \quad x_2 = \frac{M - \sqrt{M^2 - 4N}}{2}$$

If

$$M = N - \Phi(N) + 1$$

then equation $(*)$ can be rewritten as

$$x^2 - (p+q)x + pq = 0$$

Therefore,

$$x_1 \mid N, \quad x_2 \mid N.$$

If

$$M \neq N - \Phi(N) + 1$$

then neither x_1 nor x_2 is an integer (since $x_1 x_2 = pq$).

The above discussion leads to the following theorem:

Theorem 1 *If $N = pq$ is a product of two distinct odd primes, then*

$$\frac{M+\sqrt{M^2-4N}}{2} \mid N \iff M = N - \Phi(N) + 1.$$

Proof $\Longleftarrow$) It is trivial.

$\Longrightarrow$) Since $N = pq$ is a product of two distinct odd primes and $\frac{M+\sqrt{M^2-4N}}{2} \mid N$, without loss of generality, we assume that $\frac{M+\sqrt{M^2-4N}}{2} = p$. Hence $M + \sqrt{M^2-4N} = 2p$, $M^2 - 4N = 4p^2 - 4pM + M^2$. Therefore, $M = p + q = N - \Phi(N) + 1$. $\square$

3 A quantum computer algorithm for factoring RSA modulus

Denote by $\mathrm{ord}_N(2)$ the order of 2 relative to N, where N is a product of two distinct odd primes. Obviously,

$$\mathrm{ord}_N(2) \mid \Phi(N)$$

Set $s := [\frac{N}{\mathrm{ord}_N(2)}]$, where $[x]$ denotes the integer part of number x. Clearly,

$$\Phi(N) \le s \times \mathrm{ord}_N(2)$$

Therefore,

$$\Phi(N) \in \{\mathrm{ord}_N(2), 2 \times \mathrm{ord}_N(2), \cdots, (s-1) \times \mathrm{ord}_N(2), s \times \mathrm{ord}_N(2)\}.$$

It is well known that $\Phi(N)$ must be kept in secret. How to search for $\Phi(N)$ in the set

$$\{\mathrm{ord}_N(2), 2 \times \mathrm{ord}_N(2), \cdots, (s-1) \times \mathrm{ord}_N(2), s \times \mathrm{ord}_N(2)\}$$

In the following, We design a quantum computer algorithm by theorem 1, which takes advantage of the relation between computing $\Phi(N)$ and factoring N. The algorithm succeeds to compute $\Phi(N)$ and factor N synchronously.

A Quantum Algorithm for Factoring RSA Modulus:

(1) *input N, compute $ord_N(2)$ by using Shor's quantum algorithm*
(2) $s \leftarrow [\frac{N}{ord_N(2)}]$
(3) $M \leftarrow N - s \times ord_N(2) + 1$
(4) *if $M^2 - 4N$ is not a square, then* $s \leftarrow s - 1$, *goto step (3)*
(5) $t \leftarrow \frac{M+\sqrt{M^2-4N}}{2}$, *if t is not an integer, then* $s \leftarrow s - 1$, *goto step (3)*
(6) *output* t, N/t.

How much time does this algorithm take? Apart from the time of computing $\mathrm{ord}_N(2)$ in step (1), it seems that the running time of the algorithm mainly depends on the number of loops, i.e., the value of s. In fact, it only depends on the upper bound for $\frac{p+q-1}{\mathrm{ord}_{pq}(2)}$. If $\frac{p+q-1}{\mathrm{ord}_{pq}(2)} \leq k$, where k is an integer, then above algorithm will halt in k loops.

As for to verify that whether $M^2 - 4N$ is a square, easy!

4 Conclusion

In this paper, we take advantage of the particular structure of a product of two primes to design a quantum computer algorithm for factoring RSA modulus. we show that factoring RSA modulus does not need to randomly choose number x such that the order of x relative to modulus N is even. It only needs to find the order of 2 relative to modulus N, whether it is even or not.

References

[1] Peter W. Shor. Polynomial-time algorithm for prime factorization and discrete logarithms on a quantum computer. SIAM Journal on Computing Vol. 26, No. 5, pp. 1484-1509. 1997.

[2] G.L. Miller. Riemann's hypothesis and tests for primality. J. Comput. System Sci., 13, pp. 300-317. 1976.

[3] D.E. Knuth. The art of computer programming, Vol. 2: Seminumerical algorithms, 2nd ed., Addison-Wesley. 1981.

[4] L.M. Adleman. Algorithm number theory–the complexity contribution, in Pro. 35th Annual symposium on foundations of computer science, IEEE Computer Society Press, pp. 88-113. 1994.

[5] A.K. Lenstra and H.W. Lenstra. The development of the number field sieve. Lecture Notes in Mathematics 1554, 1993. Springer-Verlag, Berlin.

[6] A.K. Lenstra and H.W. Lenstra, JR., M.S. Manasse, and J.M. Pollard. The number field sieve, in Proc. 22nd Annual ACM symposium on theory of computing, Association for Computing Machinery, New York, pp. 564-572, 1990.

[7] J.M. Pollard. A Monte Carlo method for factorization. BIT 15, 331-334, 1975.

[8] Carl Pomerance and S.S. Wagstaff. Implementation of the continued fraction integer factoring algorithm. Congressus Numerantium 37, 99-118, 1983.

[9] Carl Pomerance. The quadratic sieve factoring algorithm. In Advances in Cryptology: Proceedings of Euro'1984. LNCS 209, pp. 169-182. Springer-Verlag, Berlin.