Quantum Information and Computation, Vol. 18, No. 13&14 (2018) 1143–1151

A QUANTUM PRIMALITY TEST WITH ORDER FINDING

ALVARO DONIS-VELA

Universidad de Valladolid, G-FOR: Research Group on Photonics, Quantum Information and Radiation and Scattering of Electromagnetic Waves.

JUAN CARLOS GARCIA-ESCARTIN

e-mail: juagar@tel.uva.es

Universidad de Valladolid, G-FOR: Research Group on Photonics, Quantum Information and Radiation and Scattering of Electromagnetic Waves.

Dep. Teoría de la Señal y Comunicaciones e Ingenería Telemática.

E.T.S.I. Telecomunicación, Campus Miguel Delibes, Paseo Belén 15, 47011 Valladolid, Spain.

Received January 13, 2018
Revised October 8, 2018

Determining whether a given integer is prime or composite is a basic task in number theory. We present a primality test based on quantum order finding and the converse of Fermat's theorem. For an integer N, the test tries to find an element of the multiplicative group of integers modulo N with order $N-1$. If one is found, the number is known to be prime. During the test, we can also show most of the times N is composite with certainty (and a witness) or, after $\log\log N$ unsuccessful attempts to find an element of order $N-1$, declare it composite with high probability. The algorithm requires $O((\log n)^2 n^3)$ operations for a number N with n bits, which can be reduced to $O(\log\log n(\log n)^3 n^2)$ operations in the asymptotic limit if we use fast multiplication.

Keywords: Primality tests, quantum algorithms.

Communicated by: R Jozsa & A Harrow

Prime numbers are the fundamental entity in number theory and play a key role in many of its applications such as cryptography. Primality tests are algorithms that determine whether a given integer N is prime or not. A naïve but inefficient solution is trying all the numbers up to $\sqrt{N}$ looking for a factor, which would prove N is prime if no factor is found and show it is composite if we have one. There are more efficient ways to test for primality based on different results from number theory. We are going to use basic theorems which can be found, together with their proofs, in elementary number theory books [1, 2, 3].

Some definitions are useful before we proceed. Let $\mathbb{Z}_N$ be the ring of integers modulo N and (a, N) the greatest common divisor of a and N. We call $\mathbb{Z}_N^*$ to the *multiplicative group of integers modulo* N defined as $\mathbb{Z}_N^* = \{a \in \mathbb{Z}_N : (a, N) = 1\}$. The elements of $\mathbb{Z}_N^*$ are the integers from 1 to $N-1$ which are coprime to N. These integers form a group under multiplication.

The order of a finite group G, $|G|$, is the number of elements of that group (its cardinality). The order of $\mathbb{Z}_N^*$ is given by Euler's totient function $\varphi(N)$ which gives how many integers $1 \leq a < N$ are coprime to N.

The multiplicative order of an element $a \in \mathbb{Z}_N^*$, $\text{ord}(a)$, is the smallest positive integer r such that $a^r \equiv 1 \mod N$.

With these concepts and known theorems from number theory, we can give different tests to check if an integer is prime or not. A simple test is given by Fermat's theorem:

Theorem 1 (Fermat) *If a positive integer N is prime, then $a^{N-1} \equiv 1 \mod N$ for any positive integer a such that $(a, N) = 1$.*

This is a special case of Euler's theorem:

Theorem 2 (Euler) *Let N be a positive integer, then $a^{\varphi(N)} \equiv 1 \mod N$ for any positive integer a such that $(a, N) = 1$.*

For a prime N, $\varphi(N) = N - 1$ and we recover Fermat's theorem.

If we can find an integer a for which $a^{N-1} \not\equiv 1 \mod N$, we have proof N is composite and we call a a Fermat witness for compositeness. Given a, anyone can quickly check N is not prime. This gives a simple test for primality. We pick a random a from 1 to N, verify $(a, N) = 1$ (otherwise we know N is composite) and then check for Fermat's condition. After testing a few different elements, we can declare it prime with high probability.

While this test is simple, there are certain composite numbers, called Carmichael numbers [4, 5], which obey Fermat's condition for every possible a. Any other integer will fail Fermat's test at least half of the times. To see this, we can use Lagrange's theorem:

Theorem 3 (Lagrange) *Let $|G|$ be the number of elements of a finite group G, then any subgroup S of G must have a number of elements $|S|$ which is a divisor of the size of the group.*

There is a direct application of Lagrange's theorem to Fermat's test. Fermat liars are the integers a such that $a^{N-1} \equiv 1 \mod N$ for a composite N. The liars form a subgroup of $\mathbb{Z}_N^*$. If $a_1^{N-1} \equiv 1 \mod N$ and $a_2^{N-1} \equiv 1 \mod N$ then $a_3 = a_1 a_2$ is also a liar, but multiplication outside the subgroup breaks closure. The remaining subgroup properties, like the existence of an inverse, an identity, associativity and commutativity, come immediately from the properties of $\mathbb{Z}_N^*$ and the multiplication property we just described.

From Lagrange's theorem, we see the subgroup $\mathbb{L}$ of Fermat liars must have a size which is a divisor of the size of $\mathbb{Z}_N^*$. The number of possible liars divides $\varphi(N)$ and, if there is at least one Fermat witness which shows N is composite, there must be at least $\varphi(N)/2$ witnesses. $|\mathbb{L}|$ must have a number of elements $\varphi(N)/d$, where $d|\varphi(N)$ (d is divisor of $\varphi(N)$). For $|\mathbb{L}| \neq \varphi(N)$, $d > 1$ and $|\mathbb{L}|$ is always equal to or smaller than $\varphi(N)/2$.

There are also more refined probabilistic tests similar to Fermat's which are based on more sophisticated properties. Most take advantage of Lagrange's theorem to show there is at least one witness and, therefore, the subgroup of liars has a cardinality of, at most, $|\mathbb{Z}_N^*|/2$. By choosing a random a for the test, after a few attempts with different elements, we either find a witness of compositeness or we can be satisfied that there is an exponentially small probability the number is not a prime. The two most important such methods are the Solovay-Strassen test [6] and the Miller-Rabin test [7, 8] with liar subgroups at most a half and a quarter of the total size respectively and a computational complexity essentially cubic in the number of bits of N. A good description of these and other probabilistic primality tests can be found in Dixon's review [9].

Notably, there is also a deterministic algorithm for primality testing. The AKS (Agrawal-Kayal-Saxena) primality test [10] is based on a generalization of Fermat's theorem which

states that:

Theorem 4 *A positive integer N is prime if and only if*

$$(x+a)^N \equiv x^N + a \mod N \tag{1}$$

for one a such that $(a, N) = 1$.

The AKS test is deterministic and requires a number of operations essentially of the order of the sixth power of the number of bits of N [11].

In this paper, we provide a different quantum primality test based on the *converse of Fermat's theorem* [12]

Theorem 5 (Lucas) *If*

$$a^{N-1} \equiv 1 \mod N \tag{2}$$

and

$$a^x \not\equiv 1 \mod N \tag{3}$$

for any $x < N-1$, then N is prime.

Apart from Lucas theorem, we make use of a couple of additional results (see chapter 8 of Burton's book [2]):

Theorem 6 *The elements $a \in \mathbb{Z}_N^*$ have an order* $\mathrm{ord}(a)|\varphi(N)$ *(the order is always a divisor of $|\mathbb{Z}_N^*|$).*

Theorem 7 *For a prime p and an integer $d|p-1$, $\mathbb{Z}_p^*$ has exactly $\varphi(d)$ elements a of order* $\mathrm{ord}(a) = d$.

We can now restate Lucas theorem as

Theorem 8 (Lucas) *If $a \in \mathbb{Z}_N^*$ has order* $\mathrm{ord}(a) = N-1$, *then N is prime.*

This formulation is contained in Theorem 6. The order of any element must divide $\varphi(N)$ which is only equal to $N-1$ for prime numbers. The only way we can have order $N-1$ is if N is prime. Additionally, from Theorem 7, we see there must be exactly $\varphi(N-1)$ integers with this property. If we can find such an integer, we have a way to prove primality.

There is no known classical algorithm that can determine the order of an integer efficiently. In order to apply Lucas theorem to primality certification on a classical computer, we need alternative methods. A solution is the Lucas-Lehmer test [13] based on:

Theorem 9 (Lucas-Lehmer) *If*

$$a^{N-1} \equiv 1 \mod N \tag{4}$$

and

$$a^{\frac{N-1}{p}} \not\equiv 1 \mod N \tag{5}$$

for any prime $p|N-1$, then N is prime.

With this and other refinements, there have been multiple proposals for the efficient implementation of modified Lucas tests on classical computers. In most of them, we require a complete factorization of $N-1$, or, in some, a partial factorization with large factors, both of which might be easier than factoring N (if possible). For instance, these tests are particularly easy to perform on numbers of the form $2^m - 1$ [14] which, if prime, are called Mersenne primes and include many of the largest known prime numbers [15]. The reader can find many of these methods in chapter 4 of Crandall and Pomerance's book [3].

The tests based on Lucas theorem have the advantage that they allow us to prove primality. With the Solovay-Strassen or the Rabin-Miller test we could only give a witness of compositeness, but there was no efficient way to show N was prime with certainty.

A *certificate of primality* for N is a collection of data which allows anyone to prove N is prime, ideally with few operations on short certificates with a number of bits of the order of $\log(N)$. For instance, if we have a complete factorization of $N-1$, a list of the factors and an element $a \in \mathbb{Z}_N^*$ with $\mathrm{ord}(a) = N-1$ give a fast way to show N is prime using the Lucas-Lehmer Theorem. In principle, we can always use the AKS test to check if a number is prime and, from a certain point of view, N is itself a valid certificate of primality. However, for large integers, there exist more efficient ways to prove primality if we can factor $N-1$. Pratt certificates were the first examples [16] and there are primality proofs requiring only $O(\log p)$ multiplications modulo p for any prime p [17].

At this point, it is interesting to turn to quantum computers. Shor's algorithm gives an efficient way to factor composite integers of n bits with a number of expected operations $O((\log n)n^3)$ [18], which can become $O(\log\log n(\log n)^2 n^2)$ with fast multiplication circuits. The quantum primality test of Chau and Lo [19] combines Shor's quantum factoring algorithm with the Lucas-Lehmer test to prove primality in an expected number of operations $O(n^3 \log n \log\log n)$, essentially cubic with the number of bits of N. Using quantum factoring has the nice side effect of producing a succinct certificate of primality with the results: the factors of $N-1$ and an element a which passes the Lucas-Lehmer test are enough for anyone with a classical computer to prove N is prime using only a logarithmic number of modular exponentiations, for which there are efficient classical algorithms.

Here, we propose a new quantum primality testing algorithm inspired by Shor's algorithm. By using directly the quantum order finding algorithm behind Shor's factoring and discrete logarithm algorithms, we can reduce the number of quantum operations. Instead of factoring $N-1$, we check the order of different elements in $\mathbb{Z}_N^*$ until we find one with order $N-1$ or a witness that N is composite.

This also contrasts with the quantum primality test of Carlini and Hosoya [20], which applies the concepts of quantum counting and quantum period finding to give an improved version of the Miller-Rabin test.

While we reduce the number of quantum operations, we loose the classical certificate of primality of the Chau-Lo test. Instead, we can give a *quantum certificate of primality*. Any of the $\varphi(N-1)$ elements $a \in \mathbb{Z}_N^*$ with $\mathrm{ord}(a) = N-1$ serves to prove N is prime to anyone with a quantum computer, which can find the order of a efficiently. This certificate is quantum because order finding is only known to be efficient in quantum computers. Any integer with $\mathrm{ord}(a) = N-1$ allows a quantum computer to check N is prime without running the whole algorithm, but it is not valid as a classical certificate because classical computers have no efficient way to compute $\mathrm{ord}(a)$.

We deal with numbers $2^{n-1} < N \leq 2^n$ represented with n bits. We consider quantum order finding as a black box which requires $O((\log n)n^3)$ operations and see that, on average, with $\log n$ uses of the black box we can find an element of order $N-1$ and prove N is prime when it is or show it must be composite with high probability, with a proof of compositeness in most cases.

Algorithm 1 Quantum primality test

```
Choose at random an integer 1 < a < N.
Compute (a, N):
if (a, N) ≠ 1 then
    Declare N composite; return factor (a, N) as a proof.
else if (a, N) = 1 then
    Compute a^((N−1)/2):
    if a^((N−1)/2) ≢ ±1 mod N then
        Declare N composite; return a as a witness.
    else if a^((N−1)/2) ≡ 1 mod N then
        Go back to Step 1.
    else if a^((N−1)/2) ≡ −1 mod N then
        QUANTUM ORDER FINDING. Compute ord(a):
        if ord(a) = N − 1 then
            Declare N prime; return a as a quantum certificate of primality.
        else
            Go back to Step 1.
        end if
    end if
end if
```

The proposed algorithm (Algorithm 1) checks the order of random integers in the group until it can prove either primality or compositeness. First we choose an integer at random from $\mathbb{Z}_N^*$, excluding $a = 1$, which has a trivial order 1. We take an integer $a < N$ and compute (a, N). If $(a, N) \neq 1$, we have a factor of N which serves as a proof N is composite. Before going into the quantum part, we perform a basic screening to reduce the number of quantum operations, which are the most challenging in terms of technology, and replace them by classical steps.

We need to check $a^{N-1} \equiv 1 \mod N$. Instead of performing this Fermat test directly, we check $a^{\frac{N-1}{2}}$, which should be ± 1 if a passes Fermat's test. Otherwise, N cannot be prime and we return a as a primality witness for the Fermat test. If $a^{\frac{N-1}{2}} \equiv 1 \mod N$, the order of a is, at most, $\frac{N-1}{2}$, which gives no information on whether N is prime or not. Then we start again and choose a new random integer. We only proceed if $a^{\frac{N-1}{2}} \equiv -1 \mod N$.

At this point, we need to resort to the quantum order finding algorithm. If $\mathrm{ord}(a) = N-1$ the number is prime with certainty and we can stop the procedure and return a as a quantum certificate of primality.

The classical screening guarantees $\mathrm{ord}(a)|N-1$. This follows from the condition $a^{\frac{N-1}{2}} \equiv -1 \mod N$, which means $a^{N-1} \equiv 1 \mod N$, and from [2]:

Theorem 10 *Let $a \in \mathbb{Z}_N^*$ have order $\mathrm{ord}(a)$. Then $a^h \equiv 1 \mod N$ if and only if $\mathrm{ord}(a)|h$.*

If $\mathrm{ord}(a) \neq N-1$, we cannot tell anything about N. We need to start again the search with a new element.

The average number of iterations before finding an element of order $N-1$ when N is prime is of the order of $\log\log N$. From Theorem 7, we know there are $\varphi(N-1)$ elements of order $N-1$ among the $N-1$ elements of $\mathbb{Z}_N^*$. The probability of finding an element which

confirms primality at each iteration is

$$\frac{\varphi(N-1)}{N-1} > \frac{1}{3\log\log(N-1)} \tag{6}$$

for a large enough N. In order to prove this bound we can turn to known estimations, starting from the lower bound [21]:

$$\frac{\varphi(M)}{M} > \frac{1}{e^{\gamma}\log\log M + \frac{2.50637}{\log\log M}} \tag{7}$$

where $\gamma \approx 0.57721$ is the Euler-Mascheroni constant and $e^{\gamma} \approx 1.781$. For $M > e^{e^{\sqrt{\frac{2.50637}{3-1.781}}}} \approx 49.2$ there is a lower bound

$$\frac{\varphi(M)}{M} > \frac{1}{3\log\log M}, \tag{8}$$

which tells us that for a prime N of $n > 6$ bits Equation (6) will be valid. With $3\log\log(N-1)$ attempts we have a high probability of finding an element of order $N-1$. While there is some room to improve the estimate, in the general case we cannot give a much tighter bound as it is known that for infinitely many integers

$$\frac{\varphi(M)}{M} < \frac{1}{e^{\gamma}\log\log M} < \frac{0.562}{\log\log M} \tag{9}$$

will hold [22]. We need a number of repetitions logarithmic with the number of bits of the integer under test. The complexity of each iteration is determined by the order finding subroutine.

Quantum order finding requires $O((\log n)n^3)$ quantum operations, with $O(\log n)$ uses of modular exponentiation. The quantum order finding subroutine of Shor has two main steps: modular exponentiation and a Quantum Fourier Transform. The Quantum Fourier Transform circuit is quadratic in n [18]. Modular exponentiation with the binary method needs $O(n)$ multiplications [23]. There are many quadratic quantum multiplication circuits, for instance [24, 25], which gives a total complexity of $O(n^3)$ for exponentiation. In principle, with fast multiplication using the Schönhage-Strassen algorithm [26], for which there is a quantum circuit [27], the total complexity for order finding would be $O(\log\log n(\log n)^2 n^2)$. However, the constant factors involved make it only worthwhile in the asymptotic limit for very large N [28].

The number of operations in the classical part is also dominated by modular exponentiation. Computing the greatest common divisor of two integers up to n bits using Euclid's algorithm has a complexity $O(n^2)$ and there are faster modern methods (see chapter 4 of [29]). The total expected complexity of our algorithm is $O((\log n)^2 n^3)$ for the $\log n$ repetitions needed to find an element of order $N-1$ with high probability. For very large N we can use fast multiplication to have an expected number of operations $O(\log\log n(\log n)^3 n^2)$.

The screening in line 7 of the algorithm identifies N is not prime when $\mathrm{ord}(a) \nmid N-1$, but it is possible for N to be composite and still give inconclusive results when tested. For instance, Carmichael numbers satisfy $a^{N-1} \equiv 1 \mod N$ for all a such that $(a, N) = 1$ and, from Theorem 10, $\mathrm{ord}(a) | N-1$ for all $a \in \mathbb{Z}_N^*$. In any case, the order will be smaller than

$N-1$ and, after $3\log n$ tested elements, we can say that N is composite with high probability and stop there to avoid entering an infinite loop.

In order to reduce the number of quantum operations, we can introduce a previous classical selection phase that uses the Miller-Rabin test. The elements $a \in \mathbb{Z}_N^*$ for which $a^{\frac{N-1}{2}} \equiv \pm 1 \mod N$ for a composite N are sometimes called Euler liars.[a] Euler liars are a subgroup of Fermat liars, the a for which a composite N passes the Fermat test. We can impose harder constraints on the integers that survive to the quantum part of the algorithm. For a composite number N with $N-1=2^s d$, with odd d, the bases for which $a^d \equiv 1 \mod N$ or $a^{d2^r} \equiv -1 \mod N$ for some $0 \leq r < s$ are called strong liars. For a prime N the condition always holds, but, if N is composite, at most one fourth of the $a \in \mathbb{Z}_N^*$ are strong liars [8]. If we perform a classical Rabin-Miller test on k random bases and do not find a witness for compositeness, the probability of N not being prime is bounded by 4^{-k} and we are left with a collection of k elements of order $\text{ord}(a)|N-1$. We can discard the bases with $a^d \equiv 1 \mod N$, which have order $d < N-1$, and use the rest of the elements in the quantum order finding subroutine.

Finally, we can further reduce the number of steps with some insights from the analysis of quantum order finding. The factor $\log n$ which appears in the complexity of quantum order finding is due to the average number of times we have to measure in order to find two divisors of the order from which we can deduce its exact value, $\text{ord}(a)$. However, with some classical processing testing small multiples of the values extracted from each measurement, it is possible to reduce the $\log n$ repetitions to a constant number [18].

The algorithm we have proposed offers an alternative quantum primality test which harnesses quantum order finding to give a direct proof an integer is prime by producing an element $a \in \mathbb{Z}_N^*$ with order $N-1$. The quantum part uses the same circuits as Shor's factoring algorithm and it could serve as a previous stage when factoring on a quantum machine. Shor's algorithm requires its input integer N not to be of the form p^k or $2p^k$ for a prime p and an integer $k \geq 1$. For odd inputs, we only need to worry about detecting primes. There are classical efficient methods to detect prime powers p^k for $k \geq 2$ [30], but we can also use a modified version of our quantum primality test. Theorems 6 and 7 can be generalized to show that an integer $N > 1$ has an element of order $\varphi(N)$, called a *primitive root*, only when $N = 2, 4, p^k$ or p^{2k}, in which case there are $\varphi(\varphi(N))$ of them (chapter 4 of [2]). The analysis then is essentially the same we have used in our primality test. For $N = p^k$, $\varphi(N) = p^{k-1}(p-1)$. If we find a primitive root, its order $\text{ord}(a)$ gives $(\text{ord}(a), N) = p^{k-1} \neq 1$, which factors N. We can check by repeated division by $p = \frac{N}{(\text{ord}(a),N)}$ that N is a prime power. The number of divisions is polynomial in the number of bits of N and the bound of $\log n$ order finding steps is still valid. The probability of finding a primitive root is $\frac{\varphi(\varphi(N))}{\varphi(N)}$ and $3\log\log\varphi(N)$ repetitions give a high probability of getting a valid basis. $\varphi(N) \leq N-1$, so our bound for the primes also holds in this situation.

Our algorithm reduces the asymptotic complexity of the Chau-Lo quantum primality test from $O((\log n)(\log\log n)n^3)$ to $O((\log\log n)(\log n)^3 n^2)$ at the cost of replacing the classical primality certificate which includes the factors of $N-1$ by a quantum certificate of primality consisting in an element a of order $N-1$, which can be checked on a quantum computer. The test can prove with certainty that a given integer is prime and it can be complemented with

[a]The most usual definition of Euler liars is the a which satisfy $a^{\frac{N-1}{2}} \equiv \left(\frac{a}{N}\right) \mod N$ for a composite N, where $\left(\frac{a}{N}\right)$ is the Jacobi symbol.

the Miller-Rabin test in the initial screening stage to also identify composite numbers with high probability. Our test has a complexity comparable to classical tests for compositeness which can convince us a number is prime with an exponentially small probability of error. Its complexity is essentially quadratic in the asymptotic limit, which is more efficient than classical tests that prove primality with certainty, which are usually restricted to integers of a particular form, or require a number of operations of the order of the sixth power of the number of bits (AKS test).

Acknowledgements

This work has been funded by Spanish Ministerio de Economía y Competitividad, Project TEC2015-69665-R, MINECO/FEDER, UE and Junta de Castilla y León VA089U16.

References

1. Alfred J. Menezes, Scott A. Vanstone and Paul C. Van Oorschot. *Handbook of Applied Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996.
2. David M. Burton. *Elementary Number Theory*. McGraw-Hill Higher Education, 6th revised edition edition, 2005.
3. Richard Crandall and Carl Pomerance. *Prime Numbers: A Computational Perspective*. Springer New York, NY, 2005.
4. R. D. Carmichael. On composite numbers P which satisfy the Fermat congruence $a^{P-1} \equiv 1$ mod P. *The American Mathematical Monthly*, 19(2):22–27, 1912.
5. W. R. Alford, Andrew Granville, and Carl Pomerance. There are infinitely many Carmichael numbers. *Annals of Mathematics*, 139(3):703–722, 1994.
6. R. Solovay and V. Strassen. A fast Monte-Carlo test for primality. *SIAM Journal on Computing*, 6(1):84–85, 1977.
7. Gary L. Miller. Riemann's hypothesis and tests for primality. In *Proceedings of the Seventh Annual ACM Symposium on Theory of Computing*, STOC '75, pages 234–239, New York, NY, USA, 1975. ACM.
8. Michael O Rabin. Probabilistic algorithm for testing primality. *Journal of Number Theory*, 12(1):128 – 138, 1980.
9. John D. Dixon. Factorization and primality tests. *The American Mathematical Monthly*, 91(6):333–352, 1984.
10. Manindra Agrawal, Neeraj Kayal and Nitin Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2):781–793, 2004.
11. H. W. Lenstra and C. Pomerance. Primality testing with Gaussian periods. *Preprint: https://www.math.dartmouth.edu/~carlp/aks240817.pdf*, 2003.
12. Edouard Lucas. Théorie des fonctions numériques simplement périodiques. *American Journal of Mathematics*, 1(4):289–321, 1878.
13. D. H. Lehmer. Tests for primality by the converse of Fermat's theorem. *Bulletin of the American Mathematical Society*, 33(3):327–340, 05 1927.
14. John Brillhart, D. H. Lehmer and J. L. Selfridge. New primality criteria and factorizations of $2^m \pm 1$. *Mathematics of Computation*, 29(130):620–647, 1975.
15. Collective work. The Great Internet Mersenne Prime Search (GIMPS), Ongoing from 1996, https://www.mersenne.org/.
16. Vaughan R. Pratt. Every prime has a succinct certificate. *SIAM Journal on Computing*, 4(3):214–220, 1975.
17. Carl Pomerance. Very short primality proofs. *Mathematics of Computation*, 48(177):315–322, 1987.
18. P.W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484, 1997.

19. H. F. Chau and H.-K. Lo. Primality test via quantum factorization. *International Journal of Modern Physics C*, 08(02):131–138, 1997.
20. A. Carlini and A. Hosoya. Quantum probabilistic subroutines and problems in number theory. *Physical Review A*, 62:032312, Aug 2000.
21. J. Barkley Rosser and Lowell Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics*, 6(1):64–94, 03 1962.
22. Jean-Louis Nicolas. Petites valeurs de la fonction d'Euler. *Journal of Number Theory*, 17(3):375 – 388, 1983.
23. Donald E Knuth. *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.
24. Vlatko Vedral, Adriano Barenco and Artur Ekert. Quantum networks for elementary arithmetic operations. *Physical Review A*, 54(1):147–153, Jul 1996.
25. David Beckman, Amalavoyal N. Chari, Srikrishna Devabhaktuni and John Preskill. Efficient networks for quantum factoring. *Phys. Rev. A*, 54:1034–1063, Aug 1996.
26. A. Schönhage and V. Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7(3):281–292, Sep 1971.
27. Christof Zalka. Fast versions of Shor's quantum factoring algorithm. *Preprint quant-ph/9806084*, 1998.
28. Rodney Van Meter and Kohei M. Itoh. Fast quantum modular exponentiation. *Physical Review A*, 71:052320, May 2005.
29. Eric Bach and Jeffrey Shallit. *Algorithmic Number Theory, Volume I: Efficient Algorithms*. The MIT Press, Cambridge, MA, USA, 1996.
30. Daniel J. Bernstein. Detecting perfect powers in essentially linear time. *Mathematics of Computation*, 67(223):1253–1283, July 1998.