WIKIPEDIA

Shor's algorithm

Shor's algorithm is a quantum computer algorithm for integer factorization.[1] Informally, it solves the following problem: Given an integer N, find its prime factors. It was invented in 1994 by the American mathematician Peter Shor.

On a quantum computer, to factor an integer N, Shor's algorithm runs in polynomial time (the time taken is polynomial in $\log N$, the size of the integer given as input).[2] Specifically, it takes quantum gates of order $O\left((\log N)^2(\log\log N)(\log\log\log N)\right)$ using fast multiplication,[3] thus demonstrating that the integer-factorization problem can be efficiently solved on a quantum computer and is consequently in the complexity class **BQP**. This is almost exponentially faster than the most efficient known classical factoring algorithm, the general number field sieve, which works in sub-exponential time — $O\left(e^{1.9(\log N)^{1/3}(\log\log N)^{2/3}}\right)$.[4] The efficiency of Shor's algorithm is due to the efficiency of the quantum Fourier transform, and modular exponentiation by repeated squarings.

If a quantum computer with a sufficient number of qubits could operate without succumbing to quantum noise and other quantum-decoherence phenomena, then Shor's algorithm could be used to break public-key cryptography schemes, such as the widely-used RSA scheme. RSA is based on the assumption that factoring large integers is computationally intractable. As far as is known, this assumption is valid for classical (non-quantum) computers; no classical algorithm is known that can factor integers in polynomial time. However, Shor's algorithm shows that factoring integers is efficient on an ideal quantum computer, so it may be feasible to defeat RSA by constructing a large quantum computer. It was also a powerful motivator for the design and construction of quantum computers, and for the study of new quantum-computer algorithms. It has also facilitated research on new cryptosystems that are secure from quantum computers, collectively called post-quantum cryptography.

In 2001, Shor's algorithm was demonstrated by a group at IBM, who factored 15 into 3×5, using an NMR implementation of a quantum computer with 7 qubits.[5] After IBM's implementation, two independent groups implemented Shor's algorithm using photonic qubits, emphasizing that multi-qubit entanglement was observed when running the Shor's algorithm circuits.[6][7] In 2012, the factorization of 15 was performed with solid-state qubits.[8] Also, in 2012, the factorization of 21 was achieved, setting the record for the largest integer factored with Shor's algorithm.[9]

Contents

Procedure

The problem that we are trying to solve is, given a composite number N, to find a non-trivial divisor of N (a divisor strictly between 1 and N). Before attempting to find such a divisor, one can use relatively quick primality-testing algorithms to verify that N is indeed composite.

We need N to be odd (otherwise 2 is a divisor) and not to be any power of a prime (otherwise that prime is a divisor), so we need to check that there are no integer roots $\sqrt[k]{N}$ for $2 \leq k \leq \log_3(N)$.

Hence we may assume that N is the product of two coprime integers greater than 2. It follows from the Chinese remainder theorem that there are at least four distinct square roots of 1 modulo N (since there are two roots for each modular equation). The aim of the algorithm is to find a square root b of 1 modulo N that is different from 1 and -1, because then

$$b^2 - 1 = (b+1)(b-1) = mN$$

for a non-zero integer m which gives us the non-trivial divisors $\gcd(N, b+1)$ and $\gcd(N, b-1)$ of N. This idea is similar to other factoring algorithms, like the quadratic sieve.

In turn, finding such a b is reduced to finding an element a of even period with a certain additional property (as explained below, it is required that the condition of Step 6 of the classical part does not hold). The quantum algorithm is used for finding the period of randomly chosen elements a, as this is a hard problem on a classical computer.

Shor's algorithm consists of two parts:

1. A reduction, which can be done on a classical computer, of the factoring problem to the problem of order-finding.
2. A quantum algorithm to solve the order-finding problem.

Classical part

1. Pick a random number $a < N$.
2. Compute $\gcd(a, N)$, the greatest common divisor of a and N. This may be done using the Euclidean algorithm.
3. If $\gcd(a, N) \neq 1$, then this number is a nontrivial factor of N, so we are done.
4. Otherwise, use the quantum period-finding subroutine (below) to find r, which denotes the period of the following function:

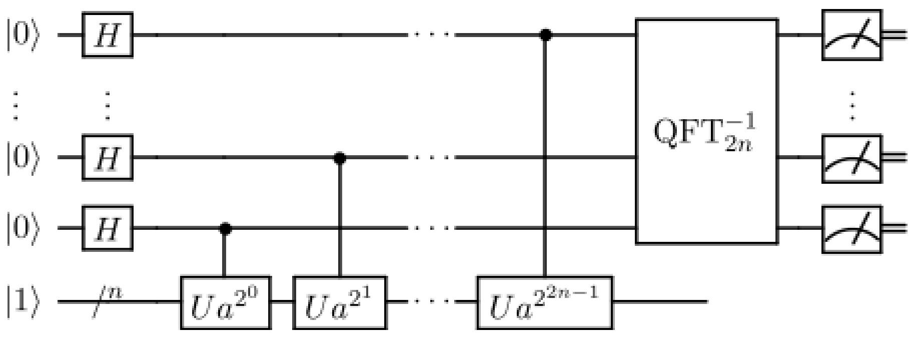
   $$f(x) = a^x \bmod N.$$

   This is the order r of a in the group $(\mathbb{Z}_N)^\times$, which is the smallest positive integer r for which $f(x+r) = f(x)$, or $f(x+r) = a^{x+r} \bmod N \equiv a^x \bmod N$. By Euler's Theorem, r divides $\varphi(N)$, where φ denotes Euler's totient function.
5. If r is odd, then go back to step 1.
6. If $a^{r/2} \equiv -1 \pmod N$, then go back to step 1.
7. Otherwise, at least one of $\gcd(a^{r/2}+1, N)$ and $\gcd(a^{r/2}-1, N)$ is a nontrivial factor of N, so we are done.

For example: Given $N = 15$, $a = 7$, and $r = 4$, we have $\gcd(7^2 \pm 1, 15) = \gcd(49 \pm 1, 15)$, where $\gcd(48, 15) = 3$ and $\gcd(50, 15) = 5$. For N that is a product of two distinct primes, p and q, the value of $\varphi(N)$ is just $N - p - q + 1$, which for $N = 15$ is 8, and r divides 8.

Quantum part: period-finding subroutine

The quantum circuits used for this algorithm are custom designed for each choice of N and each choice of the random a used in $f(x) = a^x \bmod N$. Given N, find $Q = 2^q$ such that $N^2 \leq Q < 2N^2$, which implies that $\frac{Q}{r} > N$. The input and output qubit registers need to hold superpositions of values from 0 to $Q-1$, and so have q qubits each. Using what might appear to be twice as many qubits as necessary guarantees that there are at least N different values of x that produce the same $f(x)$, even as the period r approaches $\frac{N}{2}$.



Quantum subroutine in Shor's algorithm.

Proceed as follows:

1. Initialize the registers to

$$\frac{1}{\sqrt{Q}}\sum_{x=0}^{Q-1}|x\rangle = \left(\frac{1}{\sqrt{2}}\sum_{x_1=0}^{1}|x_1\rangle\right)\otimes\cdots\otimes\left(\frac{1}{\sqrt{2}}\sum_{x_q=0}^{1}|x_q\rangle\right).$$

   where $\otimes$ denotes the tensor product.

   This initial state is a superposition of Q states, and is easily obtained by generating q independent qubits, each a superposition of 0 and 1 states. We can accomplish this by initializing the qubits to the zero-position, and then applying the hadamard gate in parallel to each of the q qubits, where $2^q = Q$.

2. Construct $f(x)$ as a quantum function and apply it to the above state, to obtain

$$\frac{1}{\sqrt{Q}}\sum_{x=0}^{Q-1}|x, f(x)\rangle.$$

   This is still a superposition of Q states. However, the $q+n$ qubits, i.e, the q input qubits and n ($> \log_2(N)$) output qubits, are now entangled or not separable, as the state cannot be written as a tensor product of states of individual qubits.

3. Apply the inverse quantum Fourier transform to the input register. This transform (operating on a superposition of $Q = 2^q$ states) uses a Q-th root of unity such as $\omega = e^{\frac{2\pi i}{Q}}$ to distribute the amplitude of any given $|x\rangle$ state equally among all Q of the $|y\rangle$ states, and to do so in a different way for each different x. We thus obtain

$$U_{\mathrm{QFT}}(|x\rangle) = \frac{1}{\sqrt{Q}}\sum_{y=0}^{Q-1}\omega^{xy}|y\rangle.$$

   This leads to the final state

$$\frac{1}{Q}\sum_{x=0}^{Q-1}\sum_{y=0}^{Q-1}\omega^{xy}|y, f(x)\rangle.$$

   Now, we reorder this sum as

$$\frac{1}{Q}\sum_{z=0}^{N-1}\sum_{y=0}^{Q-1}\left[\sum_{x\in\{0,\ldots,Q-1\};\ f(x)=z}\omega^{xy}\right]|y, z\rangle.$$

This is a superposition of many more than Q states, but many fewer than Q^2 states, as there are fewer than Q distinct values of $z = f(x)$. Let

- $\omega = e^{\frac{2\pi i}{Q}}$ be a Q-th root of unity,
- r be the period of f,
- x_0 be the smallest of the $x \in \{0, \ldots, Q-1\}$ for which $f(x) = z$ (we actually have $x_0 < r$), and
- b indexes these x, running from 0 to $\left\lfloor \frac{Q - x_0 - 1}{r} \right\rfloor$, so that $x_0 + rb < Q$.

Then ω^{ry} is a unit vector in the complex plane (ω is a root of unity, and r and y are integers), and the coefficient of $\frac{1}{Q} |y, z\rangle$ in the final state is

$$\sum_{x \in \{0,\ldots,Q-1\};\ f(x)=z} \omega^{xy} = \sum_{b=0}^{\lfloor (Q-x_0-1)/r \rfloor} \omega^{(x_0+rb)y} = \omega^{x_0 y} \sum_{b=0}^{\lfloor (Q-x_0-1)/r \rfloor} \omega^{rby}.$$

Each term in this sum represents a *different path to the same result*, and quantum interference occurs — constructive when the unit vectors ω^{ryb} point in nearly the same direction in the complex plane, which requires that ω^{ry} point along the positive real axis.

4. Perform a measurement. We obtain some outcome y in the input register and some outcome z in the output register. As f is periodic, the probability of measuring some state $|y, z\rangle$ is given by

$$\left| \frac{1}{Q} \sum_{x \in \{0,\ldots,Q-1\};\ f(x)=z} \omega^{xy} \right|^2 = \frac{1}{Q^2} \left| \sum_{b=0}^{\lfloor (Q-x_0-1)/r \rfloor} \omega^{(x_0+rb)y} \right|^2 = \frac{1}{Q^2} |\omega^{x_0 y}|^2 \left| \sum_{b=0}^{\lfloor (Q-x_0-1)/r \rfloor} \omega^{bry} \right|^2.$$

Analysis now shows that this probability is higher the closer the unit vector ω^{ry} is to the positive real axis, or the closer $\frac{yr}{Q}$ is to an integer. Unless r is a power of 2, it will not be a factor of Q.

5. Since $\frac{yr}{Q}$ is close to some integer c, the known value $\frac{y}{Q}$ is close to the unknown value $\frac{c}{r}$. Performing [classical] continued fraction expansion on $\frac{y}{Q}$ allows us to find approximations $\frac{d}{s}$ of it that satisfy two conditions:

   A. $s < N$.
   B. $\left| \frac{y}{Q} - \frac{d}{s} \right| < \frac{1}{2Q}$.

   Given these conditions (and assuming $\frac{d}{s}$ is irreducible), s is very likely to be the appropriate period r, or at least a factor of it.
6. Check (classically) if $f(x) = f(x+s) \Leftrightarrow a^s \equiv 1 \bmod N$. If so, then we are done.
7. Otherwise, (classically) obtain more candidates for r by using multiples of s, or by using other s with $\frac{d}{s}$ near $\frac{y}{Q}$. If any candidate works, then we are done.
8. Otherwise, try again starting from step 1 of this subroutine.

Explanation of the algorithm

The algorithm is composed of two parts. The first part of the algorithm turns the factoring problem into the problem of finding the period of a function and may be implemented classically. The second part finds the period using the quantum Fourier transform and is responsible for the quantum speedup.

Obtaining factors from period

The integers less than N and coprime with N form the multiplicative group of integers modulo N, which is a finite abelian group G. The size of this group is given by $\varphi(N)$. By the end of step 3, we have an integer a in this group. As the group is finite, a must have a finite order r, which is the smallest positive integer such that

$$a^r \equiv 1 \bmod N.$$

Therefore, N divides $a^r - 1$ (also written $N \mid a^r - 1$). Suppose that we are able to obtain r and that it is even. (If r is odd, then see step 5.) Now $b \equiv a^{r/2} \bmod N$ is a square root of 1 modulo N that is different from 1. This is because r is the order of a modulo N, so $a^{r/2} \not\equiv 1 \bmod N$, or else the order of a in this group would be $\frac{r}{2}$. If $a^{r/2} \equiv -1 \bmod N$, then by step 6, we have to restart the algorithm with a different random number a.

Eventually, we must hit an a of order r in G such that $b \equiv a^{r/2} \not\equiv 1, -1 \bmod N$. This is because such a b is a square root of 1 modulo N other than 1 and -1, whose existence is guaranteed by the Chinese remainder theorem, as N is not a prime power.

We claim that $d = \gcd(b-1, N)$ is a proper factor of N, i.e., $d \neq 1, N$. In fact, if $d = N$, then N divides $b - 1$, so that $b \equiv 1 \bmod N$, which goes against the construction of b. If, on the other hand, $d = \gcd(b-1, N) = 1$, then by Bézout's identity, there are integers u, v such that

$$(b-1)u + Nv = 1.$$

Multiplying both sides by $b + 1$, we obtain

$$(b^2 - 1)u + N(b+1)v = b + 1.$$

As N divides $b^2 - 1 \equiv a^r - 1 \bmod N$, we find that N divides $b + 1$, so that $b \equiv -1 \bmod N$, again contradicting the construction of b.

Therefore, d is the required proper factor of N.

Finding the period

Shor's period-finding algorithm relies heavily on the ability of a quantum computer to be in many states simultaneously. Physicists call this behavior a "superposition" of states. To compute the period of a function f, we evaluate the function at all points simultaneously.

Quantum physics does not allow us to access all this information directly, though. A measurement will yield only one of all possible values, destroying all others. If not for the no cloning theorem, we could first measure $f(x)$ without measuring x, and then make a few copies of the resulting state (which is a superposition of states all having the same $f(x)$). Measuring x on these states would provide different x values which give the same $f(x)$, leading to the period. Because we cannot make exact copies of a quantum state, this method does not work. Therefore, we have to carefully transform the superposition to another state that will return the correct answer with high probability. This is achieved by the quantum Fourier transform.

Shor thus had to solve three "implementation" problems. All of them had to be implemented "fast", which means that they can be implemented with a number of quantum gates that is polynomial in $\log N$.

1. Create a superposition of states. This can be done by applying Hadamard gates to all qubits in the input register. Another approach would be to use the quantum Fourier transform (see below).
2. Implement the function f as a quantum transform. To achieve this, Shor used repeated squaring for his modular exponentiation transformation. It is important to note that this step is more difficult to implement than the quantum Fourier transform, in that it requires ancillary qubits and substantially more gates to accomplish.

3. Perform a quantum Fourier transform. By using controlled rotation gates and Hadamard gates, Shor designed a circuit for the quantum Fourier transform (with $Q = 2^q$) that uses just $\frac{q(q-1)}{2} = O\big((\log Q)^2\big)$ gates.[10]

After all these transformations a measurement will yield an approximation to the period r. For simplicity assume that there is a y such that $\frac{yr}{Q}$ is an integer. Then the probability to measure y is 1. To see this, we notice that then

$$e^{-\frac{2\pi i byr}{Q}} = 1$$

for all integers b. Therefore, the sum whose square gives us the probability to measure y will be $\frac{Q}{r}$, as b takes roughly $\frac{Q}{r}$ values and thus the probability is $\frac{1}{r^2}$. There are r possible values of y such that $\frac{yr}{Q}$ is an integer, and also r possibilities for $f(x_0)$, so the probabilities sum to 1.

Note: Another way to explain Shor's algorithm is by noting that it is just the quantum phase estimation algorithm in disguise.

The bottleneck

The runtime bottleneck of Shor's algorithm is quantum modular exponentiation, which is by far slower than the quantum Fourier transform and classical pre-/post-processing. There are several approaches to constructing and optimizing circuits for modular exponentiation. The simplest and (currently) most practical approach is to mimic conventional arithmetic circuits with reversible gates, starting with ripple-carry adders. Knowing the base and the modulus of exponentiation facilitates further optimizations.[11][12] Reversible circuits typically use on the order of n^3 gates for n qubits. Alternative techniques asymptotically improve gate counts by using quantum Fourier transforms, but are not competitive with fewer than 600 qubits due to high constants.

Discrete logarithms

Given a group G with order p and generator $g \in G$, suppose we know that $x = g^r \in G$, for some $r \in \mathbb{Z}_p$, and we wish to compute r, which is the discrete logarithm: $r = \log_g(x)$. Consider the abelian group $\mathbb{Z}_p \times \mathbb{Z}_p$, where each factor corresponds to modular addition of values. Now, consider the function

$$f\colon \mathbb{Z}_p \times \mathbb{Z}_p \to G\ ;\ f(a,b) = g^a x^{-b}.$$

This gives us an abelian hidden subgroup problem, as f corresponds to a group homomorphism. The kernel corresponds to the multiples of $(r, 1)$. So, if we can find the kernel, we can find r.

In popular culture

On the television show *Stargate Universe*, the lead scientist, Dr. Nicholas Rush, hoped to use Shor's algorithm to crack *Destiny*'s master code. He taught a quantum cryptography class at the University of California, Berkeley, in which Shor's algorithm was studied.

In the animated film *Summer Wars*, the character Kenji Koiso reads an article titled "Shor's Factorization Algorithm" while riding on a train, foreshadowing his ability to understand and calculate complex equations.

In the TV series *The Big Bang Theory* it was mentioned in a Physics bowl competition in season 1 episode 13.

See also

- In 2017, Daniel J. Bernstein et al. proposed a factorization algorithm known as GEECM said to be "often much faster than Shor's".[13]

References

1. Shor, P.W. "Algorithms for quantum computation: discrete logarithms and factoring" (https://doi.org/10.1109/sfcs.1994.365700). *Proceedings 35th Annual Symposium on Foundations of Computer Science*. IEEE Comput. Soc. Press. doi:10.1109/sfcs.1994.365700 (https://doi.org/10.1109%2Fsfcs.1994.365700). ISBN 0818665807.
2. See also pseudo-polynomial time.
3. Beckman, David; Chari, Amalavoyal N.; Devabhaktuni, Srikrishna; Preskill, John (1996). "Efficient Networks for Quantum Factoring" (https://authors.library.caltech.edu/2179/1/BECpra96.pdf) (PDF). *Physical Review A*. **54** (2): 1034–1063. arXiv:quant-ph/9602016 (https://arxiv.org/abs/quant-ph/9602016). Bibcode:1996PhRvA..54.1034B (https://ui.adsabs.harvard.edu/abs/1996PhRvA..54.1034B). doi:10.1103/PhysRevA.54.1034 (https://doi.org/10.1103%2FPhysRevA.54.1034).
4. "Number Field Sieve" (http://mathworld.wolfram.com/NumberFieldSieve.html). *wolfram.com*. Retrieved 23 October 2015.
5. Vandersypen, Lieven M. K.; Steffen, Matthias; Breyta, Gregory; Yannoni, Costantino S.; Sherwood, Mark H. & Chuang, Isaac L. (2001), "Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance" (http://cryptome.org/shor-nature.pdf) (PDF), *Nature*, **414** (6866): 883–887, arXiv:quant-ph/0112176 (https://arxiv.org/abs/quant-ph/0112176), Bibcode:2001Natur.414..883V (https://ui.adsabs.harvard.edu/abs/2001Natur.414..883V), CiteSeerX 10.1.1.251.8799 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.251.8799), doi:10.1038/414883a (https://doi.org/10.1038%2F414883a), PMID 11780055 (https://www.ncbi.nlm.nih.gov/pubmed/11780055)
6. Lu, Chao-Yang; Browne, Daniel E.; Yang, Tao & Pan, Jian-Wei (2007), "Demonstration of a Compiled Version of Shor's Quantum Factoring Algorithm Using Photonic Qubits" (http://discovery.ucl.ac.uk/153160/1/153160.pdf) (PDF), *Physical Review Letters*, **99** (25): 250504, arXiv:0705.1684 (https://arxiv.org/abs/0705.1684), Bibcode:2007PhRvL..99y0504L (https://ui.adsabs.harvard.edu/abs/2007PhRvL..99y0504L), doi:10.1103/PhysRevLett.99.250504 (https://doi.org/10.1103%2FPhysRevLett.99.250504), PMID 18233508 (https://www.ncbi.nlm.nih.gov/pubmed/18233508)
7. Lanyon, B. P.; Weinhold, T. J.; Langford, N. K.; Barbieri, M.; James, D. F. V.; Gilchrist, A. & White, A. G. (2007), "Experimental Demonstration of a Compiled Version of Shor's Algorithm with Quantum Entanglement" (http://espace.library.uq.edu.au/view/UQ:134503/UQ134503.pdf) (PDF), *Physical Review Letters*, **99** (25): 250505, arXiv:0705.1398 (https://arxiv.org/abs/0705.1398), Bibcode:2007PhRvL..99y0505L (https://ui.adsabs.harvard.edu/abs/2007PhRvL..99y0505L), doi:10.1103/PhysRevLett.99.250505 (https://doi.org/10.1103%2FPhysRevLett.99.250505), PMID 18233509 (https://www.ncbi.nlm.nih.gov/pubmed/18233509)
8. Lucero, Erik; Barends, Rami; Chen, Yu; Kelly, Julian; Mariantoni, Matteo; Megrant, Anthony; O'Malley, Peter; Sank, Daniel; Vainsencher, Amit; Wenner, James; White, Ted; Yin, Yi; Cleland, Andrew N.; Martinis, John M. (2012). "Computing prime factors with a Josephson phase qubit quantum processor". *Nature Physics*. **8** (10): 719. arXiv:1202.5707 (https://arxiv.org/abs/1202.5707). Bibcode:2012NatPh...8..719L (https://ui.adsabs.harvard.edu/abs/2012NatPh...8..719L). doi:10.1038/nphys2385 (https://doi.org/10.1038%2Fnphys2385).
9. Martín-López, Enrique; Martín-López, Enrique; Laing, Anthony; Lawson, Thomas; Alvarez, Roberto; Zhou, Xiao-Qi; O'Brien, Jeremy L. (12 October 2012). "Experimental realization of Shor's quantum factoring algorithm using qubit recycling" (http://www.nature.com/nphoton/journal/vaop/ncurrent/full/nphoton.2012.259.html). *Nature Photonics*. **6** (11): 773–776. arXiv:1111.4147 (https://arxiv.org/abs/1111.4147). Bibcode:2012NaPho...6..773M (https://ui.adsabs.harvard.edu/abs/2012NaPho...6..773M). doi:10.1038/nphoton.2012.259 (https://doi.org/10.1038%2Fnphoton.2012.259). Retrieved October 23, 2012.
10. Shor 1999, p. 14
11. Markov, Igor L.; Saeedi, Mehdi (2012). "Constant-Optimized Quantum Circuits for Modular Multiplication and Exponentiation". *Quantum Information and Computation*. **12** (5–6): 361–394. arXiv:1202.6614 (https://arxiv.org/abs/1202.6614). Bibcode:2012arXiv1202.6614M (https://ui.adsabs.harvard.edu/abs/2012arXiv1202.6614M).

12. Markov, Igor L.; Saeedi, Mehdi (2013). "Faster Quantum Number Factoring via Circuit Synthesis". *Phys. Rev. A*. **87** (1): 012310. arXiv:1301.3210 (https://arxiv.org/abs/1301.3210). Bibcode:2013PhRvA..87a2310M (https://ui.adsabs.harvard.edu/abs/2013PhRvA..87a2310M). doi:10.1103/PhysRevA.87.012310 (https://doi.org/10.1103%2FPhysRevA.87.012310).
13. Bernstein, Daniel J.; Heninger, Nadia; Lou, Paul; Valenta, Luke (2017). "Post-quantum RSA" (https://cr.yp.to/papers/pqrsa-20170419.pdf) (PDF). *International Workshop on Post-Quantum Cryptography*: 311–329. Archived (https://web.archive.org/web/20170420151942/https://cr.yp.to/papers/pqrsa-20170419.pdf) (PDF) from the original on 2017-04-20.

Further reading

- Nielsen, Michael A. & Chuang, Isaac L. (2010), *Quantum Computation and Quantum Information, 10th Anniversary Edition*, Cambridge University Press, ISBN 9781107002173.
- Phillip Kaye, Raymond Laflamme, Michele Mosca, *An introduction to quantum computing*, Oxford University Press, 2007, ISBN 0-19-857049-X
- "Explanation for the man in the street" (http://scottaaronson.com/blog/?p=208) by Scott Aaronson, "approved (http://scottaaronson.com/blog/?p=208#comment-9958)" by Peter Shor. (Shor wrote "Great article, Scott! That's the best job of explaining quantum computing to the man on the street that I've seen."). An alternate metaphor for the QFT was presented in one of the comments (http://www.scottaaronson.com/blog/?p=208#comment-5187). Scott Aaronson suggests the following 12 references as further reading (out of "the $10^{10^{5000}}$ quantum algorithm tutorials that are already on the web."):
- Shor, Peter W. (1997), "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer", *SIAM J. Comput.*, **26** (5): 1484–1509, arXiv:quant-ph/9508027v2 (https://arxiv.org/abs/quant-ph/9508027v2), Bibcode:1999SIAMR..41..303S (https://ui.adsabs.harvard.edu/abs/1999SIAMR..41..303S), doi:10.1137/S0036144598347011 (https://doi.org/10.1137%2FS0036144598347011). Revised version of the original paper by Peter Shor ("28 pages, LaTeX. This is an expanded version of a paper that appeared in the Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, Nov. 20--22, 1994. Minor revisions made January, 1996").
- Quantum Computing and Shor's Algorithm (https://quantum-algorithms.herokuapp.com/299/paper/index.html), Matthew Hayward's Quantum Algorithms Page (https://quantum-algorithms.herokuapp.com/), 2005-02-17, imsa.edu, LaTeX2HTML version of the original LaTeX document (https://quantum-algorithms.herokuapp.com/299/paper.tex), also available as PDF (https://quantum-algorithms.herokuapp.com/299/paper.pdf) or postscript (https://quantum-algorithms.herokuapp.com/299/paper.ps) document.
- Quantum Computation and Shor's Factoring Algorithm (http://homepages.cwi.nl/~rdewolf/publ/qc/survey.ps), Ronald de Wolf, CWI and University of Amsterdam, January 12, 1999, 9 page postscript document.
- Shor's Factoring Algorithm (http://www.cs.berkeley.edu/~vazirani/f04quantum/notes/lec9.ps), Notes from Lecture 9 of Berkeley CS 294-2, dated 4 Oct 2004, 7 page postscript document.
- Chapter 6 Quantum Computation (http://www.theory.caltech.edu/people/preskill/ph229/notes/chap6.ps), 91 page postscript document, Caltech, Preskill, PH229.
- Quantum computation: a tutorial (http://www-users.cs.york.ac.uk/~schmuel/comp/comp.html) by Samuel L. Braunstein (http://www.cs.york.ac.uk/~schmuel/).
- The Quantum States of Shor's Algorithm (http://www.cs.ucr.edu/~neal/1996/cosc185-S96/shor/high-level.html), by Neal Young, Last modified: Tue May 21 11:47:38 1996.
- III. Breaking RSA Encryption with a Quantum Computer: Shor's Factoring Algorithm (https://web.archive.org/web/20121115112940/http://people.ccmr.cornell.edu/~mermin/qcomp/chap3.pdf), Lecture notes on Quantum computation, Cornell University, Physics 481-681, CS 483; Spring, 2006 by N. David Mermin. Last revised 2006-03-28, 30 page PDF document.
- Lavor, C.; Manssur, L. R. U.; Portugal, R. (2003). "Shor's Algorithm for Factoring Large Integers". arXiv:quant-ph/0303175 (https://arxiv.org/abs/quant-ph/0303175).
- Lomonaco, Jr (2000). "Shor's Quantum Factoring Algorithm". arXiv:quant-ph/0010034 (https://arxiv.org/abs/quant-ph/0010034). This paper is a written version of a one-hour lecture given on Peter Shor's quantum factoring algorithm. 22 pages.
- Chapter 20 Quantum Computation (http://www.cs.princeton.edu/theory/complexity/quantumchap.pdf), from *Computational Complexity: A Modern Approach*, Draft of a book: Dated January 2007, Sanjeev Arora and Boaz Barak, Princeton University. Published as Chapter 10 Quantum Computation of Sanjeev Arora, Boaz Barak, "Computational Complexity: A Modern Approach", Cambridge University Press, 2009, ISBN 978-0-521-42426-4
- A Step Toward Quantum Computing: Entangling 10 Billion Particles (http://blogs.discovermagazine.com/80beats/2011/01/19/a-step-towards-quantum-computing-entangling-10-billion-particles/), from "Discover Magazine", Dated January 19, 2011.

- Josef Gruska - *Quantum Computing Challenges* (http://www.fi.muni.cz/usr/gruska/survey1.ps) also in Mathematics unlimited: 2001 and beyond (https://www.amazon.com/Mathematics-Unlimited-Bj%C3%B6rn-Engquist/dp/3540669132), Editors Björn Engquist, Wilfried Schmid, Springer, 2001, ISBN 978-3-540-66913-5

External links

- Version 1.0.0 of libquantum (http://www.libquantum.de/): contains a C language implementation of Shor's algorithm with their simulated quantum computer library, but the width variable in shor.c should be set to 1 to improve the runtime complexity.
- PBS Infinite Series created two videos explaining the math behind Shor's algorithm, "How to Break Cryptography (http://www.pbs.org/video/how-to-break-cryptography-ahby1s/)" and "Hacking at Quantum Speed with Shor's Algorithm (http://www.pbs.org/video/hacking-at-quantum-speed-with-shors-algorithm-8jrjkq/)".