Shor’s Factoring Algorithm

Combine quantum order finding with classical number theory to extract factors and recognise unsuccessful bases.

Undergraduate 22 min lesson Dr. Charith Jayasekara

By the end, you’ll be able to

- Perform the gcd preprocessing and half-order factor extraction.
- Work both a successful and a failed base for fifteen.
- Separate polynomial ideal complexity from physical-resource and security claims.

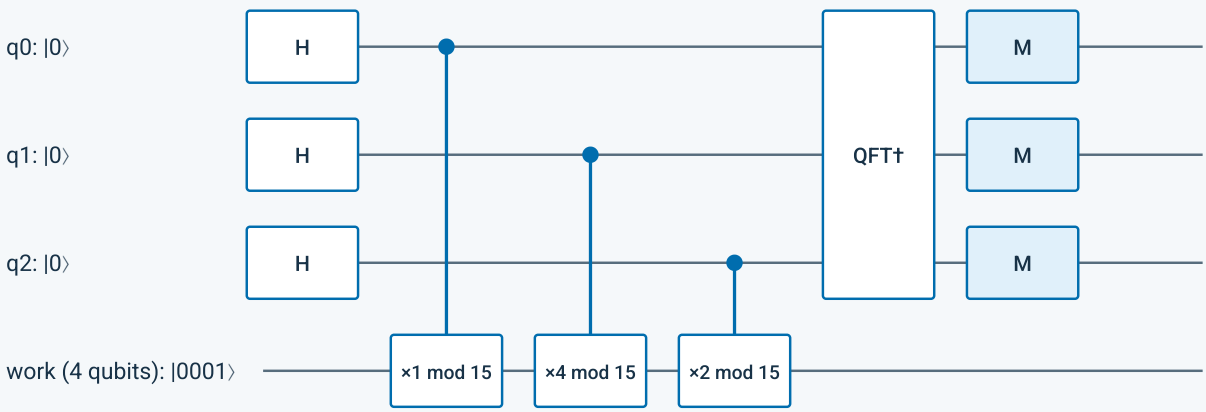
Factoring combines quantum and classical work

Shor’s factoring algorithm finds nontrivial factors of a composite integer. The quantum part obtains an order in modular arithmetic. Classical preprocessing chooses a useful instance, and classical postprocessing turns a suitable order into factors. The complete algorithm is therefore a hybrid mathematical procedure, not a single Fourier circuit that measures two prime numbers directly.

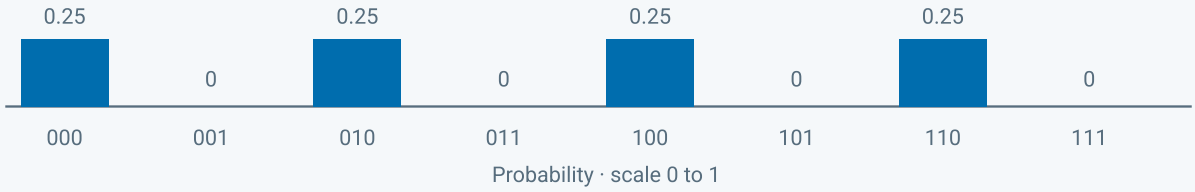
Read order finding first. Useful classical prerequisites are modular exponentiation and the Euclidean algorithm for greatest common divisors. A nontrivial factor is a divisor greater than one and smaller than the original number. Factoring and testing whether a number is prime are different tasks.

WORKED DIAGRAM · A14

Shor: a quantum order, then classical factors



Counting-register output



Measure 010 → phase 1/4 → verified order 4. $2^2 \bmod 15 = 4$; $\gcd(4-1,15)=3$; $\gcd(4+1,15)=5$.

Three counting bits suffice for this exact r=4 teaching instance. A general unknown-order implementation uses a precision chosen from N and error requirements.

The counting register controls reversible modular multipliers computed from a=2 and N=15. After inverse QFT, label 010 yields phase 1/4 and verified order four. Classical gcds give three and five. A different base can require a retry.

The problem and input size

The input is an ordinary classical integer N. The goal is to return a nontrivial factor when N is composite, with a controlled success probability. The size parameter is approximately the number of binary digits of N. A polynomial algorithm in that size is very different from trying every integer up to N.

Handle easy cases classically first. An even N has factor two. Perfect powers can be recognised and reduced to smaller tasks. For the main quantum reduction, consider an odd composite that is not a prime power. Choose an integer a between one and N and compute its greatest common divisor with N. If that divisor is nontrivial, the factor has already been found without a quantum call.

The cost is measured in the number of input bits, and a classical gcd check can already reveal an easy factor.

$$n = \lfloor \log_2 N \rfloor + 1, \qquad d = \gcd(a, N), \qquad 1 < d < N \implies d \text{ is a nontrivial factor.}$$

For the main odd-composite reduction, first choose $1 < a < N$ and handle a nontrivial gcd immediately. Continue to order finding when $d = 1$.

The quantum subroutine

If a is coprime to N, find its order r: the smallest positive exponent returning one modulo N. Quantum phase estimation on reversible modular multiplication supplies samples from which r can be reconstructed and verified. The easy initial modular state is a superposition of relevant eigenstates, so repeated phase samples and classical processing may be required.

The modular arithmetic dominates much of the circuit engineering. Controlled powers exploit repeated squaring and structured reversible multiplication. They should not be implemented by blindly repeating one base operation an exponentially large number of times. Equally, a demonstration that hard-codes a known small period is not a scalable implementation of unknown order finding.

Turn an even order into factors

If r is even, set x to a raised to half r modulo N. Then x squared is one modulo N. Therefore N divides the product of x minus one and x plus one. If x is neither plus one nor minus one modulo N, those two neighbouring expressions can share different nontrivial factors with N. Compute their greatest common divisors with N.

If r is odd, this route does not apply. If x is minus one modulo N, the gcds are trivial. Retry with another base. A correctly verified order should not give x equal to plus one for a positive even minimal r, since that would contradict minimality. The usual success analysis shows that random suitable choices provide a useful constant success probability for the standard class of composite inputs, with repetition increasing confidence.

An even order produces a square root of one modulo N. A useful root splits the product into factors through two gcds.

$$x = a^{r/2} \bmod N, \qquad x^2 \equiv 1 \pmod N, \qquad (x-1)(x+1) \equiv 0 \pmod N, \qquad d_{\pm} = \gcd(x \pm 1, N).$$

Use a verified even minimal order r. If $x \not\equiv \pm 1 \pmod N$, the gcd step returns nontrivial factors. An odd order or $x \equiv -1$ calls for another base.

Checkpoint 2

PAUSE & THINK

A verified minimal order is even, but x=−1 modulo N. What should this factoring attempt do?

- ☐ Return −1 as a factor
- ☐ Claim that N is prime
- ☐ Retry with another base

Answer: Retry with another base. When x≡−1, the gcds are trivial in the standard reduction. Repetition with another suitable base is part of the algorithm.

Work N equal to fifteen

Choose a equal to two. Its greatest common divisor with fifteen is one. The powers cycle through two, four, eight and one, so r is four. Half the order is two, giving x equal to four. The two gcd calculations are

$$\gcd(4-1,15)=3, \qquad \gcd(4+1,15)=5.$$

Both are nontrivial factors, and multiplying them checks the result. The quantum subroutine's purpose was to expose the order four, not to output three and five directly. The final arithmetic is entirely classical and cheap for this tiny example.

Now choose a equal to fourteen, which is negative one modulo fifteen. Its order is two, but x is fourteen, also negative one modulo fifteen. The gcds of thirteen and fifteen, and fifteen and fifteen, are one and fifteen. This attempt produces no nontrivial factor. Finding an even order is necessary for this route but is not sufficient on its own.

Put the successful and failed bases beside each other to expose the extra condition beyond an even order.

a	r	$x=a^{r/2} \bmod 15$	$\gcd(x-1,15)$	$\gcd(x+1,15)$
2	4	4	3	5
4	2	4	3	5
14	2	$14\equiv -1$	1	15

All three bases are coprime to fifteen and have even order. The last row fails because the half-order power is minus one.

Checkpoint 1

PAUSE & THINK

For a=4 and N=15, r=2. Which factors follow?

- [ ] 1 and 15
- [ ] 3 and 5
- [ ] 2 and 4

Answer: 3 and 5. x=4^(2/2) mod 15=4. Its neighbouring integers give gcds three and five.

Why the reduction is useful

The identity x squared equal to one modulo N can have nontrivial solutions when N is composite. Modular order finding supplies a way to encounter such a solution. The classical random choice makes it likely to find a helpful case after a manageable number of attempts. The proof combines number theory with the quantum order-finding guarantee.

Do not infer that factoring is classically proven to require exponential time. No polynomial-time classical factoring algorithm is known in the conventional model, and the best known classical methods have a different asymptotic form. Shor establishes a polynomial-time quantum algorithm; an exact unconditional separation from all possible classical factoring algorithms is a stronger statement than the known result.

What the speedup includes and leaves out

The theoretical algorithm uses a number of ideal operations polynomial in the bit-length, with sufficiently accurate arithmetic and phase estimation. Practical execution requires logical reliability over a long coherent calculation. Physical qubits, runtime, error correction, control, routing and resource-state production depend on architecture and error assumptions. A small compiled demonstration cannot establish those resources for a large input.

RSA’s security relevance comes from its mathematical dependence on factoring-related structure, but the existence of Shor’s algorithm is different from possession of a device capable of running it at a cryptographic scale. Resource estimates must be read with their assumptions and dates. The book’s hardware chapter discusses that distinction separately from this correctness proof.

Shor also gave a quantum algorithm for discrete logarithms, a related algebraic task with a different reduction. It is not the same as applying the factoring gcd steps to every public-key scheme. Understanding which mathematical problem underlies a cryptosystem is essential before drawing an algorithmic conclusion about it.

A sound learning experiment

Work several small bases for fifteen or twenty-one by hand. Record the initial gcd, the order, whether it is even, the half-order power and the final gcds. Mark failed attempts honestly. Then study a modular arithmetic circuit that computes the order without inserting the known answer into its construction. This separates a correctness illustration from a scalable implementation claim.

Chapter summary

- Shor combines classical preprocessing, quantum order finding and classical factor extraction.
- An even minimal order is useful when its half-order power is a nontrivial square root of one.
- A failed base is an expected retry, while large-number execution also requires reliable arithmetic and error correction.

A common misconception

The QPE measurement directly prints the prime factors. It supplies phase information for reconstructing an order. Classical gcd calculations then extract factors if that order and base satisfy the reduction's conditions.

Code practice

Download the tested Qiskit notebook and follow the pinned environment instructions. Inspect the circuit, predict a result and run its independent assertions. The modular-action example is a small-instance demonstration; the notebook states its precision and post-processing limits.

Exercises

1. For N fifteen and a equal to four, find the order and perform the factor extraction.
2. A report says, "An even order always gives factors." Use the base fourteen example to correct the statement.

Worked answers

1. Four squared is sixteen, congruent to one modulo fifteen, so r is two. The half-order power is four. The gcds are three and five, giving the factors.
2. An even order must also yield a half-order power other than minus one modulo N. For base fourteen, r is two but the half-order power is minus one, so the gcds are one and fifteen. The algorithm must retry with another base when the extraction conditions fail.

Check your understanding

PAUSE & THINK

Why does base fourteen fail to factor fifteen through the half-order step?

- [ ] Its order is odd
- [ ] The half-order power is −1 modulo fifteen
- [ ] Fourteen and fifteen share a factor

Answer: The half-order power is −1 modulo fifteen. The order is two and the half-order power is fourteen, or −1 modulo fifteen. The resulting gcds are one and fifteen, so no nontrivial factor is obtained.

Sources

Shor's primary paper presents the polynomial-time quantum factoring and discrete-logarithm algorithms. IBM's Shor derivation gives a guided order-finding reduction.