Indian Journal of Pure & Applied Physics
Vol. 64, April 2026, pp. 418-424
DOI: 10.56042/ijpap.v64i4.27189

Deconstructing Shor's Algorithm Using Quantum Fourier Transform

Dushyant Kumar[a], Balwinder Raj[a] & Gurmohan Singh[b*]

[a] Department of Electronics and Communication Engineering, Dr B R Ambedkar National Institute of Technology Jalandhar, Punjab 144 008, India

[b] Quantum Technologies Lab, Centre for Development of Advanced Computing, Mohali, Punjab 160 071 India

Received:2nd January 2026; accepted: 5th February 2026

Shor's algorithm proved a significant milestone in quantum computing. Because it promises an exponential speedup over conventional algorithms for integer factorization, a problem critical to modern cryptography systems. This work covers implementation steps of Shor's algorithm and analyzes Quantum Fourier Transform (QFT) usage for extracting periodicity from quantum superpositions. Firstly, theoretical foundations of algorithm are illustrated concentrating on QFT construction and operation within the setting of the quantum circuit. Thereafter, each implementation step such as qubit optimization, modular exponentiation, and circuit design is discussed. Our findings validate the theoretical effectiveness of the QFT in solving the period-finding subroutine, while also highlighting the practical difficulties and scaling constraints presented by existing quantum hardware. The paper ends with a summary of possible advancements and future paths for error mitigation strategies and quantum algorithm design. For the benchmark case $N = 15$, (N denotes the composite integer to be factored) the implemented order-finding circuit exhibits a logical depth of approximately 10, increasing to about 20–25 layers after gate decomposition, highlighting the rapid depth scaling that constrains near-term quantum implementations.

Keywords: Quantum fourier's transform, Shor's algorithm, Cryptography, Superposition

1 Introduction

One of the most important advancements in quantum computing was Peter Shor's algorithm published in 1994[1]. An integer factorization method offers a quantum solution to this issue by breaking down a huge composite number into its prime components. Even for conventional supercomputers, this is an extremely difficult and time-consuming task while factoring very large numbers. Popular encryption algorithms like Rivest, Shamir, and Adleman (RSA)[2] are predicated on the difficulties of factoring huge numbers almost impossible with conventional computing resources. One popular method for factoring numbers in polynomial time is Shor's algorithm. Because factoring the product of two primes takes longer than polynomial time using the most popular classical algorithm, the commonly used cryptographic system e.g. RSA depends on factoring being impossible for sufficiently large numbers[2]. This paper emphasizes the quantum component of Shor's algorithm, which genuinely resolves the period finding issue. An effective period finding algorithm can be used to factor integers effectively since a factoring problem can be converted into a period finding problem in polynomial time. For the time being, it suffices to demonstrate that number factoring can be done effectively if we calculate the order r of $a^x \ mod \ N$. Here, a is the known number or a coprime of N. Prior to discussing how this could be used as a factor in this work, let us first address the period finding problem which is building block of number factorization methods[3-4]. The ability to factor large integers manifold faster than the most popular classical methods make Shor's method revolutionary. Shor's method may factor an integer N in polynomial time or $O((\log) \ N)^3$ using a quantum computer whereas conventional methods such as the General Number Field Sieve (GNFS) require sub-exponential time. The method does this by reducing the factoring problem to a period-finding problem to address it more efficiently exploiting quantum ideas like superposition and the Quantum Fourier Transform (QFT). Shor's algorithm has both conventional and quantum components. The quantum component targets calculating the order of a modular exponential function which is extremely slow in a classical system[5]. Once the period has been calculated, classical computing can be utilized to find the original number's prime factors. Even if large-scale implementation of Shor's method is currently beyond the capability of current quantum computers, it serves as a strong demonstration of quantum computation's

*Corresponding author: E-mail: gurmohan@cdac.in

possibilities[6-7]. Post-quantum cryptography techniques must be developed as soon as possible due to their significant influence on cybersecurity. Shor's algorithm is not only a crucial algorithm in quantum computing theory, but it also represents a paradigm shift in the digital security infrastructure[8-10].

2 Literature Work

2.1 Quantum Superposition and Parallelism

Quantum computing emerged as novel approach to information processing using fundamental concepts of quantum physics. The concepts of quantum superposition and parallelism make it unique from classical methods of computation. These properties allow quantum computers to solve some problems manifold faster than conventional computers[11]. Superposition is the ability of a quantum bit, or qubit, to exist in many states simultaneously. Unlike classical bits, which can only be either *0* or *1*, a qubit can be in a state that is a linear combination of both *0* and *1*. The quantum state of a qubit is represented mathematically in Eq. (1) as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \quad \ldots (1)$$

where, the complex integers α and β represent probability amplitudes of quantum state being $|0\rangle$ and $|1\rangle$; respectively with $|\alpha|^2 + |\beta|^2 = 1$. All feasible combinations of their classical values can be represented by a superposition of qubits when more than one is involved.

Quantum parallelism is a consequence of this characteristic. When an operation (often a quantum gate or function) is applied to a register of qubits in superposition, the quantum computer applies the operation to all of the states in the superposition at the same time. To put it another way, a quantum computer can evaluate a function on multiple inputs simultaneously. All function outcomes are not directly obtained via this parallelism, though; rather, it enables quantum algorithms to use interference and measurement to extract global features from the system, such as periodicity or patterns[12].

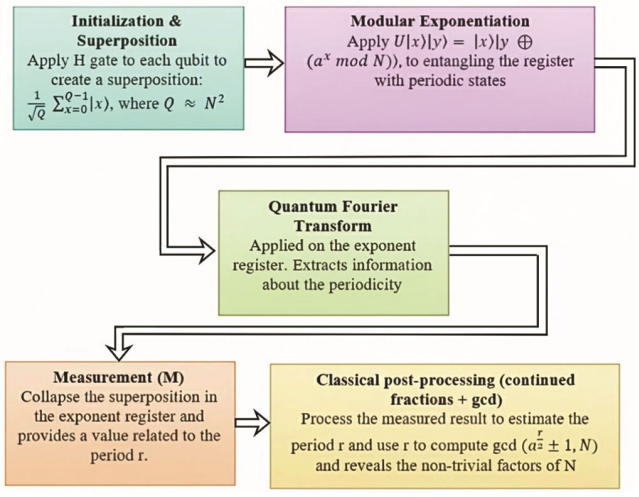

Fig. 1 — Standard sequence of steps in Shor's algorithm

For instance, exponential scalability is made possible by the simultaneous representation of $2^3 = 8$ states by 3 qubits. The key example of quantum parallelism in operation is Shor's technique for integer factorization. The approach computes $a^x mod\, N$ for every x in parallel by preparing a superposition of all potential exponents and applying quantum parallelism. The order of the function is then extracted using the Quantum Fourier Transform (QFT), which is utilized to determine the factors of N.

2.2 Shor's Algorithm

Shor's algorithm solves the challenge of identifying the prime factors of a composite integer N in polynomial time on a quantum computer, which may crack RSA encryption[1]. The approach reduces factorization to a period-finding issue and employs quantum phase estimation using the QFT. The general sequence of steps followed in Shor's algorithm is shown in Fig. 1 describing way a composite number N is factored by algorithm leveraging quantum parallelism and QFT for period finding.

2.3 Quantum Fourier Transform

The QFT provides the foundational framework for the development of Shor's algorithm. It enables the efficient determination of a function's period, a critical step in factoring large integers. The following sections provides a detailed explanation of the algorithm emphasizing the central role of the QFT[12–13].

2.3.1 Problem Configuration

The goal is to find non-trivial prime factors of a composite number N. Select a random number so that $1 < a < N$ (Here a is randomly chosen integer coprime to N) and gcd $(a, N) = 1$. If gcd $(a, N) > 1$, a factor is found immediately. Then define the function $f(x) = a^x \bmod N$. This function is periodic with period r, meaning $f\{x\} = f(x + r)$. Determining this period r is the quantum part of the algorithm[1].

2.3.2 Set up the Quantum Registers

Select two quantum registers to be used. The first register is then initialized in a superposition of all possible values of x as described in Eq. (2).

$$\frac{1}{\sqrt{Q}} \sum_{x=0}^{Q-1} | x\rangle \quad \ldots (2)$$

Here, Q is the dimension of Hilbert space, i.e. the total number of computational basis states included in the superposition.

where, Q is a power of 2 such that $Q > N^2$

The second register is initialized to $| 0\rangle$, and then modular exponentiation is applied as described by Eq. (3).

$$\frac{1}{\sqrt{Q}} \sum_{x=0}^{Q-1} | x\rangle | f(x)\rangle \quad \ldots (3)$$

This entangles the input x with its function value $f(x)$[1].

2.3.3 Measure the Second Register

The wave function collapses to superposition of only those x values that correspond to a certain function output (x_o) when the second register is measured. The first register becomes a superposition of states separated by the unknown period r due to the periodicity of the function. The first register takes the form of Eq. (4).

$$\frac{1}{\sqrt{M}} \sum_{k=0}^{M-1} | x_0 + kr\rangle \quad \ldots (4)$$

where, M is the number of terms in the superposition and $M \approx \frac{n}{r}$. Here, n is the size of the computational basis and order r is the function.

2.3.4 Apply the Quantum Fourier Transform (QFT)

The QFT is applied to the first register. The periodic superposition is mapped by the QFT to a state with high probability amplitudes at multiples of Q/r. In terms of mathematics, QFT transforms as described in Eq. (5).

$$\sum_{j=0}^{r-1} e^{2\pi i s j / r} | j\rangle \; = \rightarrow k \approx Q.\frac{s}{r} \quad \ldots (5)$$

The measurement of first register after the QFT gives a value k which is close to some integer multiple of $\frac{Q}{r}$.

Figure 2 illustrates a 5-qubit quantum register undergoing a QFT followed by a bit-reversal (swap) operation. Each qubit labeled q_0 through q_4 is initialized in the $|0\rangle$ state with the QFT applied beginning from the lowest-order qubit (q_4). The QFT circuit maps quantum basis states onto their corresponding frequency components by introducing relative phase shifts through Hadamard and controlled-phase rotation gates. Finally, qubit swaps are performed to reverse the bit order. QFT plays a pivotal role in algorithms such as Shor's, where it enables the extraction of periodicity from quantum states[14-18]. The simulation is performed using 1024 shots under ideal conditions, without incorporating any noise model.

2.3.5 Post-processing in the classical sense (continued fraction expansion)

From the measured value k, the ratio k/Q is computed and approximated as s/r, where s and r are integers, using the continued fraction method[19-21]. Use continued function expansion to find a fraction close to $\frac{k}{Q}$. The denominator of this fraction is the candidate for the period r. Once a candidate r is obtained, verify that r is even and that it fulfills condition $a^{r/2} \not\equiv -1 \pmod N$. If both conditions hold, compute $\gcd\left(a^{r/2}\right) \pm 1, N$. These are non-trivial factors of N. If either condition fails, the standard procedure is to repeat the algorithm with a different random base a or to consider an alternative convergent from the continued-fraction expansion.

3 Materials and Methods

3.1 Period Finding in Shor's Algorithm

The core classical problem that Shor's algorithm solves is period finding for the function in Eq. (6).

$$f(x) = a^x \; mod \; N \quad \ldots (6)$$

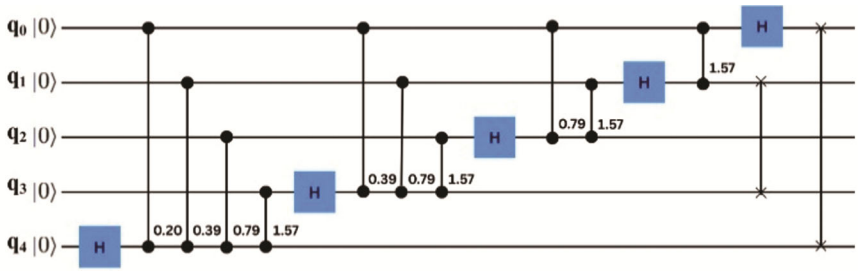

Fig. 2 — 5-qubit QFT circuit is constructed using a systematic sequence of Hadamard gates and controlled phase-rotation gates to realize the desired transformation. The circuit is implemented and simulated using the Qiskit framework with the Aer Simulator backend to verify its functionality

where, integers a and N both are positive, there is no factor in common between them, and a is less than N. The smallest non-zero integer is the period or order (r), which means that $a^x \, mod \, N = 1$. The *order* (or *period*) r of a modulo N is the smallest positive integer satisfying $a^r \equiv 1 \, (mod \, N)$. Finding this period r is the central objective of the quantum subroutine: once r is known, the factorization of N can often be obtained by classical post-processing.

Figure 3 illustrates an example plot of $f(x) = a^x \, mod N$. The plotted markers indicate integer x values. The connecting lines simply help with visual perception and are not meant to represent intermediate (non-integer) states. The following conditions represent three different scenarios arise in finding period r

3.1.1 Condition I

If the value a is less than N but not equals to zero ($1 < a < N$) then the function $f(x)$ shown in Fig. 3 is periodic with some positive period r. Consider an example with parameters: N = 46, a = 3, where a is less then and not equal to 1 ($a < N$) and $\gcd(a, N) = 1$ (here no common factors).

The function $3^x \, mod \, 46$ is a periodic with the period $r = 11$, meaning $3^{x+11} \, mod \, 46 = \, 3^x mod \, 46$ for all x

Now quantitively;
At $x = 0, 3^0 \, mod \, 46 = 1$.
At $x = 1, 3^1 \, mod \, 46 = 3$.
At $x = 11, 3^{11} \, mod \, 46 = 1$. (It indicates start of the cycle)

It is observed that here every 11 steps, the values fluctuate between 0 and 40, with lows near 0 to 5 and peaks around 35 to 40.

3.1.2 Condition II

If $\gcd(a, N) > 1$, then integer a already shares a nontrivial factor with N. In this case the algorithm immediately yields that common factor (and factorization proceeds classically), so the order-finding subroutine is unnecessary. In other words, a common divisor of a and N provides a direct certificate of compositeness. Then the output has not any value of *r,* that is r = 0, as shown in Fig. 4.

3.1.3 Condition III

If the selected base a is greater than the modulus N, it is first reduced modulo N and replaced by $a mod N$. This reduction ensures that the analysis is carried out within the valid range $1 < a < N$. The periodic behavior of the function depends solely on this reduced residue.

$$f(x) = a^x mod N \quad \ldots (7)$$

Consequently, when $a > N$, the original value of $a$has no influence on the outcome; instead, the existence of a valid period r depends entirely on the reduced value $a mod N$ and on whether $\gcd(a, N) = 1$.

If the reduced value $a mod N$ is not coprime with N, the function fails to exhibit periodic behavior, and no valid order exists. In such cases, the period is taken as $r = 0$, as illustrated in Fig. 5.

3.2 Decoding Shor's Algorithm

Shor's algorithm uses quantum phase estimation (QPE) on the unitary operator $U|y\rangle = |ay \, mod \, N\rangle$ encapsulating modular exponentiation. To understand the algorithm, consider an eigenstate (U). If it is start from

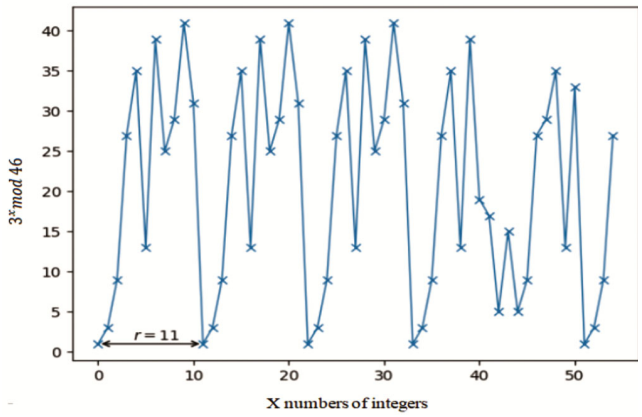

Fig. 3 — Illustration of the periodic function $f(x) = a^x mod N$ for $N = 46$ and $a = 3$. The function exhibits a clear periodic pattern with a period $r = 11$, where $f(x + 11) = f(x)$ for all x

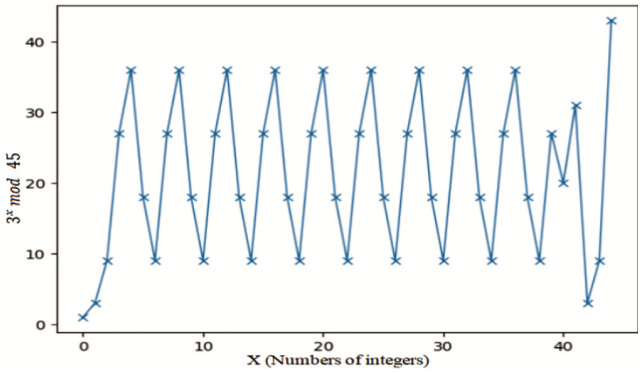

Fig. 4 — Plot of the modular exponential function $f(x) = a^x mod N$ when $\gcd(a, N) > 1$. In this case, a shares a nontrivial common factor with N, providing an immediate classical factor of N. The function does not exhibit periodicity, indicating that the order r is undefined ($r = 0$)

computational basis state $|1\rangle$, then every subsequent application of U multiplies the state of our register by $a\ (mod\ N)$. After r applications, the state is reached $|1\rangle$ once more. For instance, when $N = 46$ and $a = 3$.

$U|1\rangle = |3\rangle$
$U^2|1\rangle = |9\rangle$
$U^3|1\rangle = |27\rangle$ ⋮
$U^{r-1}|1\rangle = |31\rangle$
$U^r|1\rangle = |1\rangle$

Figure 6 illustrates how a quantum register is affected when a quantum unitary operation U is applied repeatedly. It is an essential component of the quantum period finding subroutine, a crucial step in Shor's technique for factoring big numbers. Here, the order 11 is indicated by the pattern repeating every r = 11 steps. This is most likely a simulation of the Shor's Algorithm's modular exponentiation circuit.

4 Results

Figure 7 depicts block diagram of Shor's algorithm illustrating the main functional components of the quantum circuit. The process consists of four major stages: quantum register initialization, controlled modular multiplications, application of the inverse QFT, and final measurement. Together, these blocks implement the quantum period-finding subroutine essential for efficient integer factorization.

The main component of Shor's algorithm for factoring a number 15 using base 7 is represented by this quantum circuit. The functions of each component are broken down as follows:

4.1 Initialization

In Fig. 7 of circuit implementation, the top three qubits $q_0, q_1,$ and q_2 play the role of the control register and are initialized using Hadamard gates (H) to create a uniform superposition of all possible exponent states $|x\rangle$. The qubit q_3 is initialized to the state $|1\rangle$ by applying an X gate, setting its initial value to one for modular multiplication. The remaining qubits i.e. q_4 to q_6 act as ancilla qubits for supporting the modular exponentiation process during the execution of the algorithm.

4.2 Modular Exponentiation

The controlled unitary operations in the circuit are designed to compute $7^x\ mod\ 15$, where x represents the binary value encoded in the control qubits. Each modular multiplication block corresponds to a specific power of the base and is conditionally applied depending on the state of the control register. Specifically, the circuit implements controlled modular multiplications of $7^1 mod 15$, $7^2 mod 15$, and $7^4 mod 15$, thereby simulating the modular exponential function $f(x) = a^x mod N$ with parameters $a = 7$ and $N = 15$. These controlled operations collectively enable the period-finding subroutine playing critical operation in Shor's algorithm.

4.3 Inverse Quantum Fourier Transform (IQFT)

After the function $7^x mod 15$ is encoded into the quantum state, the inverse QFT is performed on the control register to extract the order of the function through quantum interference. This transformation converts the phase information accumulated during the modular exponentiation into a measurable form, enabling the identification of the order r, satisfying the relation $a^r mod N = 1$. Calculation of this period is the main objective of the quantum subroutine in Shor's algorithm. It provides the essential information required for efficient integer factorization.

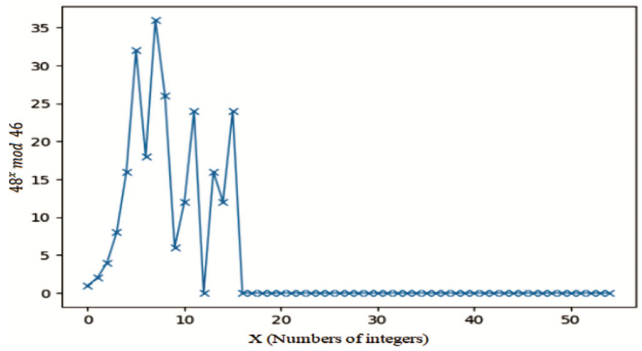

Fig. 5 — Plot of $f(x) = a^x mod N$ for $a > N$. After reduction of $a$modulo N, no periodic behavior is observed, indicating $r = 0$

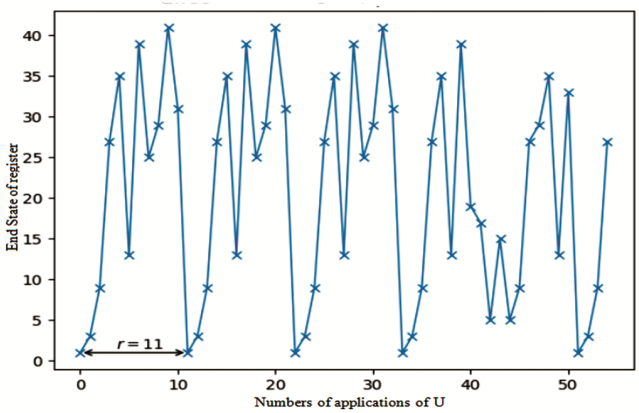

Fig. 6 — Simulation of repeated applications of the unitary operator U in Shor's algorithm, showing periodic evolution of the quantum register with period $r = 11$

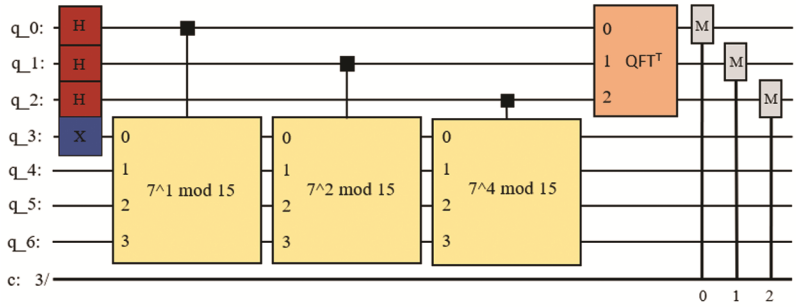

Fig. 7 — Block-level representation of Shor's algorithm showing the key stages register initialization, controlled modular multiplications, inverse QFT, and measurement. The algorithmic implementation and simulation has been carried out using the Qiskit framework with the Aer Simulator backend

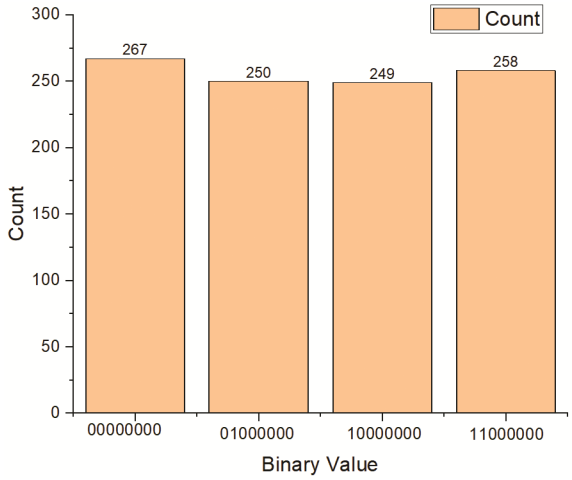

Fig. 8 — Modular exponential function $f(x) = 7^x mod 15$ showing periodicity with period $r = 4$ enabling the factorization of $N = 15$ in Shor's algorithm through computation of gcd $(a^{r/2} \pm 1, N)$

4.4 Measurement

The control qubits are measured, giving a classical outcome related to the period r, which is used in classical post-processing to find the factors of 15. This circuit is used to find the order r of 7 modulo 15, once r is found (say $r = 4$), Shor's algorithm uses $\gcd(7^{r/2}) \pm 1, 15$ to find nontrivial factors of 15 (in this case 3 and 5). The simulation results obtained from circuits in Fig. 7 are shown in Fig. 8.

The 3-qubit measurement outcomes are shown using zero-padded 8-bit binary labels for visualization purposes.

The results confirm that the function $f(x) = 7^x \ mod \ 15$ contains $r = 4$. In order to support factor 15; this value must be determined precisely in Shor's method. The Fig. 8 results confirm that the modular exponential function $f(x) = 7^x mod 15$ shows a order $r = 4$. It means $7^{x+4} mod 15 = 7^x mod 15$ for all integer values of x. Computing the period is the main goal assigned to quantum subroutine in Shor's algorithm. The period r provides the essential information needed to factorize $N = 15$ efficiently. Once the value $r = 4$ is obtained, the subsequent classical post-processing step through computation of gcd $(a^{r/2} \pm 1, N)$ produces the non-trivial factors of 15.

Table 1 — Resource metrics of the quantum circuit

Metric	Value
Qubits	5
Hadamard gates	5
Controlled-phase gates	10
SWAP gates	2
Total logical gates	17
CNOT-equivalent gates	≈16
Logical circuit depth	≈10
Hardware depth (IBM devices)	≈20–25

4.5 Efficiency of the Algorithm

According to Fig. 8, Shor's algorithm has a high efficiency for the tiny instance of $N = 15$ with a order $r = 4$. The circuit efficiently estimates the period by utilizing QPE and a small number of quantum gates and qubits. The practical efficiency of Shor's algorithm depends on the performance of quantum hardware and the effectiveness of error correction aspects not addressed in this paper. Theoretically, the algorithm exhibits polynomial scaling with $\log N$, delivering an exponential speedup over classical factoring methods for large integers. The main resource metrics of the quantum circuit, such as the number of qubits, the distribution of gates, and the total circuit complexity are compiled in Table 1.

4.6 Challenges and Future Directions

There are several obstacles to implementing Shor's algorithm with the QFT including high error rates,

limited qubit count, and deep circuit requirements that are difficult for modern quantum hardware to meet. Modular exponentiation and precise QFT implementation need a lot of resources and are prone to error propagation. Developing error mitigation strategies, building hardware-efficient circuits, and improving QFT through approximations are all part of future progress. There are encouraging avenues for testing on sophisticated quantum systems and hybrid quantum-classical models.

5 Conclusion

The principal quantum mechanism underpinning Shor's algorithm is the QFT which transforms quantum states into the frequency domain to extract the order of the modular function $f(x) = a^x mod N$. By calculating this period, the algorithm enables efficient factorization of large integers which go beyond existing classical algorithms. Experimental demonstrations so far proved the factorization of small numbers $(N = 15)$ just validating the fundamental concepts of the algorithm. The exponential speedup promised by Shor's algorithm underscores the profound potential of quantum computing specifically in cryptography challenging the security assumptions of existing classical encryption schemes such as RSA. Although Shor's technique is groundbreaking in principle, qubit defects, scaling problems, and hardware constraints pose significant practical obstacles.

References

1 Shor P W, Proc 35th Annu Symp Found Comput Sci, 1 (1994) 124.
2 Rivest R L, Shamir A & Adleman L, Commun ACM, 21 (2) (1978) 120.
3 Yu Kitaev A, "Quantum measurements and the Abelian stabilizer problem," arXiv preprint quant-ph/9511026 (1995).
4 Coppersmith D, "An approximate Fourier transform useful in quantum factoring," arXiv preprint quant-ph/0201067 (2002).
5 Dewitte L, Roland J & Eulitz F, Comput Fluids, 295 (2025) 106619.
6 Willsch D, Willsch M, Jin F, De Raedt H & Michielsen K, "Large-scale simulation of Shor's quantum factoring algorithm," arXiv preprint arXiv:2308.05047 (2023).
7 Skosana U & Tame M, Sci Rep, 11 (2021) 16599.
8 Bavdekar R, Chopde E, Bhatia A, Tiwari K, Daniel S J & Atul, "Post quantum cryptography: techniques, challenges, standardization, and directions for future research," arXiv preprint arXiv:2202.02826 (2022). DOI: 10.48550/arXiv. 2202.02826.
9 Barzen J & Leymann F, "Post-quantum security: origin, fundamentals, and adoption," arXiv preprint arXiv:2405. 11885 (2024). DOI: 10.17352/tcsit.000089.
10 Markidis S, "What is quantum parallelism, anyhow?" arXiv preprint arXiv:2405.07222 (2024).
11 11 Wu S, Zhang Y & Li J, Phys Rev Res, 7 (1) (2025) 013177.
12 Coppersmith D, "An approximate Fourier transform useful in quantum factoring," arXiv preprint quant-ph/0201067 (2002), DOI: 10.48550/arXiv.quant-ph/0201067
13 Nielsen M A & Chuang I L, Quant Comp Quant Inform, Cambridge University Press, Cambridge (2010).
14 Barenco A, Bennett C H, Cleve R, DiVincenzo D P, Margolus N, Shor P, Sleator T, Smolin J A & Weinfurter H, Phys Rev A, 52 (1995) 3457.
15 Cleve R & Watrous J, "Proc 41st Annu Symp Found Comput Sci, 454 (2000) 526.
16 Vandersypen L M K, Steffen M, Breyta G, Yannoni C S, Sherwood M H & Chuang I L, *Nature*, 414 (2001) 883.
17 Griffiths R B & Niu C-S, *Phys Rev Lett*, 76 (1996) 3228.
18 Beauregard S, *Quant Inf Comput*, 3 (2003) 175.
19 Caves C M, Fuchs C A & Schack R, *J Math Phys*, 43 (2002) 4537.
20 Hallgren S, *J ACM*, 49 (2002) 1.
21 Ekert A & Jozsa R, *Rev Mod Phys*, 68 (1996) 733.