

Algoritmo de Shor

Fatoração Inteira, Estimativa de Fase e Implementação em Circuitos Quânticos

Prof. Me. Bryan Kano

Prof. Dr. Routo Terada

Contexto e impacto

Algoritmo de Shor (Peter Shor, 1994):

- algoritmo quântico para **fatorar inteiros em tempo polinomial** em $\log N$;
- resolve de forma eficiente o problema de fatoração, considerado difícil classicamente;
- ameaça esquemas criptográficos baseados em fatoração, como o **RSA**.

Em hardware IBM atual (exemplo):

- demonstrações compiladas de Shor para $N = 15$ podem rodar em **poucos segundos** em processadores como o *Eagle r3*;
- isso não quebra RSA real ainda, mas mostra a **direção** do poder computacional.

Mensagem: Shor é o exemplo clássico do poder transformador da computação quântica em criptografia, segurança cibernética e matemática computacional.

Objetivos da aula

Ao final desta aula, o estudante deve ser capaz de:

- entender o problema de **fatorar** um inteiro N e sua relação com RSA;
- compreender a **ordem modular** e por que ela é o alvo quântico;
- entender o **problema de estimativa de fase quântica** (QPE);
- ver como QPE é usado para resolver o problema de **determinação de ordem**;
- seguir passo a passo a **parte quântica** do algoritmo de Shor;
- entender a parte **clássica** final que transforma a ordem r em fatores de N ;
- acompanhar um exemplo completo para $N = 15$, $a = 2$ e a ideia de implementação em Qiskit.

Roteiro da aula

- 1 Motivação, fatoração e RSA
- 2 A redução clássica: fatoração \Rightarrow ordem
- 3 Estimativa de fase quântica (QPE)
- 4 Determinação de ordem via QPE
- 5 Parte quântica de Shor: passo a passo
- 6 Exemplo: Shor para $N = 15$, $a = 2$
- 7 Execução em hardware e limitações práticas
- 8 Visão geral final

Problema de fatoração

Queremos fatorar um inteiro composto:

$$N > 1, \quad N \text{ não primo.}$$

Problema de fatoração:

Encontrar p, q tais que $N = p \cdot q, \quad 1 < p, q < N.$

Exemplo:

$$N = 15 \Rightarrow p = 3, \quad q = 5.$$

Para N com algumas dezenas de bits: factível. Para N com 2048 bits (RSA típico): nenhum algoritmo clássico conhecido resolve em tempo viável.

Ligaçāo com RSA

Esquema RSA (resumido):

- escolhemos p, q primos grandes, secretos;
- definimos $N = p \cdot q$ e tornamos N público;
- chaves e segurança dependem de ser **difícil** recuperar p, q a partir de N .

Se alguém consegue fatorar N :

- obtém p e q ;
- consegue derivar a chave secreta e quebrar o esquema.

Ponto-chave:

Shor fornece um **algoritmo quântico** que, em princípio, torna essa fatoração eficiente.

Revisão: aritmética modular

Trabalhamos no conjunto:

$$\mathbb{Z}_N = \{0, 1, 2, \dots, N - 1\}.$$

Escrevemos:

$$a \equiv b \pmod{N}$$

Leitura:

“a é congruente a b módulo N.”

Significados equivalentes:

- a e b deixam o mesmo resto na divisão por N;
- N divide ($a - b$): $N \mid (a - b)$.

Operações $+, -, \times$ são sempre feitas com resto módulo N.

Conjunto invertível \mathbb{Z}_N^*

Definimos:

$$\mathbb{Z}_N^* = \{a \in \mathbb{Z}_N : \gcd(a, N) = 1\}.$$

Leitura:

“ \mathbb{Z}_N^* é o conjunto dos números entre 0 e $N - 1$ que são coprimos com N .”

Propriedades:

- cada $a \in \mathbb{Z}_N^*$ tem inverso multiplicativo módulo N ;
- operações de multiplicação módulo N restrinvidas a \mathbb{Z}_N^* formam um **grupo**.

Definição: ordem modular

Seja $a \in \mathbb{Z}_N^*$. A **ordem de a módulo N** é:

$$r = \text{ord}_N(a) = \min\{r > 0 : a^r \equiv 1 \pmod{N}\}.$$

Leitura:

"r é o menor expoente positivo que faz a^r ter resto 1 na divisão por N."

Exemplo: $N = 15$, $a = 2$:

$$2^1 = 2, \quad 2^2 = 4, \quad 2^3 = 8, \quad 2^4 = 16 \equiv 1 \pmod{15} \Rightarrow r = 4.$$

Intuição geral: qual é o truque do Shor?

- Problema clássico: **fatorar** um inteiro grande N .
- O algoritmo de Shor não ataca diretamente a fatoração.
- Em vez disso, ele faz uma sequência de “truques matemáticos”:
 - ➊ Troca fatoração \Rightarrow problema de **achar um período**.
 - ➋ Codifica esse período como uma **fase quântica**.
 - ➌ Usa a **Transformada de Fourier Quântica (QFT)** para extrair esse período de forma eficiente.
- Ideia central: a mecânica quântica é excelente em detectar **periodicidade escondida** em funções.

Problema original: fatorar N

- Queremos escrever:

$$N = p \cdot q,$$

onde p e q são primos (ou encontrar qualquer fator não trivial de N).

- Não se conhece algoritmo clássico em tempo polinomial em $\log N$ para fatorar N .
- A segurança de esquemas como RSA depende do fato de que, **na prática clássica**, fatorar números de 2048/4096 bits é muito caro.
- Shor percebeu: em vez de atacar a fatoração diretamente, é melhor transformá-la num problema de **ordem (período)** de uma função modular.

Truque 1: fatorar \Rightarrow achar um período

Passo 1: escolher a e definir uma função modular

- Escolhemos um inteiro a tal que

$$1 < a < N \quad \text{e} \quad \gcd(a, N) = 1.$$

- Definimos a função:

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

- Essa função é calculada “normalmente” e, no fim, tomamos o resto da divisão por N .

Propriedade-chave: $f(x)$ é periódica. Existe um inteiro $r > 0$ tal que:

$$f(x + r) \equiv f(x) \pmod{N} \quad \text{para todo } x.$$

Esse r é chamado de **ordem de a módulo N** .

Função modular e período: definição formal

Ordem de a módulo N :

$$r = \min\{ r > 0 : a^r \equiv 1 \pmod{N} \}.$$

- r é o menor expoente positivo para o qual a^r “volta para 1” no sistema módulo N .
- Isso implica:

$$f(x + r) = a^{x+r} \pmod{N} = a^x \cdot a^r \pmod{N} \equiv a^x \cdot 1 \equiv f(x) \pmod{N}.$$

- Portanto, o período de $f(x) = a^x \pmod{N}$ é exatamente a ordem r de a módulo N .

Exemplo concreto de período: $N = 15$, $a = 2$

Sejam $N = 15$ e $a = 2$. Então:

$$f(0) = 2^0 \bmod 15 = 1,$$

$$f(1) = 2^1 \bmod 15 = 2,$$

$$f(2) = 2^2 \bmod 15 = 4,$$

$$f(3) = 2^3 \bmod 15 = 8,$$

$$f(4) = 2^4 \bmod 15 = 16 \equiv 1 \pmod{15},$$

$$f(5) = 2^5 \bmod 15 = 32 \equiv 2 \pmod{15},$$

⋮

- Sequência dos valores: $1, 2, 4, 8, 1, 2, 4, 8, \dots$
- Claramente, a sequência se repete a cada 4 passos.
- Logo, o período é $r = 4$ e a ordem de 2 módulo 15 é:

Por que saber o período ajuda a fatorar N ?

Suponha que encontramos r tal que

$$a^r \equiv 1 \pmod{N}.$$

- Isso significa que N divide $a^r - 1$:

$$a^r - 1 \equiv 0 \pmod{N} \Rightarrow N \mid (a^r - 1).$$

- Se r é par, podemos escrever:

$$a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1).$$

- Como N divide o produto, há boa chance de que:

$$\gcd(a^{r/2} - 1, N) \quad \text{ou} \quad \gcd(a^{r/2} + 1, N)$$

seja um fator não trivial de N .

Em resumo: descobrir r fornece, com alta probabilidade, um caminho para fatorar N .



Por que achar o período é difícil para um computador clássico?

Definição direta:

$$r = \min\{ r > 0 : a^r \equiv 1 \pmod{N} \}.$$

- Modo mais direto (força bruta):

$$a^1 \bmod N, a^2 \bmod N, a^3 \bmod N, \dots$$

até encontrar o primeiro expoente que dá resto 1.

- Em casos ruins, r pode ser grande (da ordem de N).
- Para N com milhares de bits, isso é **praticamente inviável**.
- Mesmo com algoritmos clássicos avançados (Number Field Sieve, etc.), o custo cresce mais rápido do que qualquer polinômio em $\log N$.

Moral: **extrair a estrutura global de período de $f(x)$ é muito caro para algoritmos clássicos conhecidos.**

Truque 2: codificar o período em fase quântica

Shor define um operador unitário de **multiplicação modular**:

$$M_a |x\rangle = |ax \bmod N\rangle .$$

- Esse operador tem autovetores $|\psi_j\rangle$ e autovalores da forma:

$$M_a |\psi_j\rangle = e^{2\pi i \frac{j}{r}} |\psi_j\rangle ,$$

onde r é justamente a ordem de a módulo N .

- Observe: o número r aparece “escondido” no **exponente da fase**:

$$e^{2\pi i \frac{j}{r}}.$$

Ideia: transformar o problema de achar o período em um problema de estimar essa fase.

Como a fase aparece no circuito quântico

- Preparamos um registrador quântico em superposição de muitos valores de x :

$$\frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x\rangle .$$

- Implementamos um circuito que, de forma reversível, calcula

$$|x\rangle |1\rangle \longmapsto |x\rangle |a^x \bmod N\rangle .$$

- Em uma única aplicação do circuito, o computador quântico avalia $a^x \bmod N$ para todos os x da superposição.
- A periodicidade de $f(x) = a^x \bmod N$ gera uma estrutura altamente regular neste estado em superposição (uma “onda discreta” de passo r).

Essa estrutura regular é justamente o que será lido pela QFT.

Truque 3: QFT como analisador de espectro do período

A Transformada de Fourier Quântica em 2^m pontos é a operação:

$$|x\rangle \mapsto \frac{1}{\sqrt{2^m}} \sum_{y=0}^{2^m-1} e^{2\pi i xy/2^m} |y\rangle .$$

- Quando aplicamos a QFT a uma superposição periódica (com passo r), as fases $e^{2\pi i xy/2^m}$ se combinam.
- Isso gera **picos de probabilidade** em valores de y tais que

$$\frac{y}{2^m} \approx \frac{j}{r}$$

para algum inteiro j .

- Medindo o registrador após a QFT, obtemos um y que codifica a razão j/r .
- A partir de $y/2^m$, usamos **frações contínuas** para recuperar r .

Mini-truque extra: enganando a necessidade de autovetores

Em princípio, precisaríamos preparar um autovetor $|\psi_j\rangle$ de M_a .

Fato importante:

$|1\rangle$ pode ser escrito como superposição uniforme dos autovetores $|\psi_k\rangle$.

Consequências:

- Preparar o estado simples $|1\rangle$ já contém **todos os autovetores relevantes** ao mesmo tempo.
- Cada execução do algoritmo se comporta como se tivéssemos escolhido um $|\psi_k\rangle$ aleatório.
- Repetindo o experimento algumas vezes, a estatística dos resultados é suficiente para reconstruir r com alta probabilidade.

Por que a quântica ganha aqui? (complexidade)

Do ponto de vista de custo computacional:

- Parte quântica do Shor:
 - Implementa exponenciação modular em superposição.
 - Usa QFT em 2^m pontos, com custo $\mathcal{O}(m^2)$ (ou melhor).
 - Escolhendo $m \approx 2 \log N$, o custo é polinomial em $\log N$.
- Parte clássica (frações contínuas, gcd etc.) também é polinomial em $\log N$.

Comparação:

- Shor (quântico): tempo **polinomial** em $\log N$.
- Melhores algoritmos clássicos conhecidos: **subexponenciais**, mas ainda **superpolinomiais** em $\log N$.

Resumo da intuição

O truque do Shor

Transformar fatoração em um problema de achar o período de uma função modular, codificar esse período como fase em um circuito quântico, e usar a Transformada de Fourier Quântica para extrair essa fase (e portanto o período) em tempo polinomial em $\log N$.

- Em seguida, na aula principal, entramos:
 - na construção detalhada do circuito;
 - na conexão com a Estimativa de Fase Quântica (QPE);
 - e no exemplo completo de fatorar $N = 15$ (ou outro) com Shor.

Redução: fatoração \Rightarrow encontrar ordem

Ideia central de Shor (parte clássica):

Se conseguimos encontrar a ordem r de um a módulo N , então muitas vezes conseguimos fatorar N .

Passos:

- ① Escolher a aleatório com $1 < a < N$.
- ② Calcular $d = \gcd(a, N)$:
 - se $d > 1$, já achamos um fator;
 - se $d = 1$, prosseguir.
- ③ Encontrar a ordem r de a módulo N (é aqui que entra o computador quântico).
- ④ Usar r para extrair fatores via gcd.

Por que a ordem leva à fatoração? (1/3)

Suponha que encontramos r tal que:

$$a^r \equiv 1 \pmod{N}.$$

Isso implica:

$$a^r - 1 \equiv 0 \pmod{N} \quad \Rightarrow \quad N \mid (a^r - 1).$$

Se r é par, podemos escrever:

$$a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1).$$

Pergunta natural:

- “Por que esse fatoramento ajuda a achar fatores de N ?”

Por que a ordem leva à fatoração? (2/3)

Da igualdade:

$$N \mid (a^{r/2} - 1)(a^{r/2} + 1),$$

temos que N divide o produto de dois números.

Se acontecer que:

$$a^{r/2} \not\equiv \pm 1 \pmod{N},$$

então:

- nenhum dos fatores $(a^{r/2} - 1)$ e $(a^{r/2} + 1)$ é múltiplo de N ;
- mas o produto deles é múltiplo de N ;
- logo, cada um carrega uma parte dos fatores de N .

Então procuramos os **maiores divisores comuns** com N .

Por que a ordem leva à fatoração? (3/3)

Definimos:

$$p_1 = \gcd(a^{r/2} - 1, N), \quad p_2 = \gcd(a^{r/2} + 1, N).$$

Interpretação:

- p_1 é o maior número que divide simultaneamente $a^{r/2} - 1$ e N ;
- p_2 é o maior número que divide simultaneamente $a^{r/2} + 1$ e N .

Com boa probabilidade:

$$1 < p_1, p_2 < N,$$

ou seja, encontramos fatores **não triviais** de N .

Exemplo completo: $N = 15$, $a = 2$

Já sabemos que para $N = 15$, $a = 2$, a ordem é $r = 4$.

$$2^4 \equiv 1 \pmod{15}.$$

Como r é par, $r/2 = 2$:

$$2^{r/2} = 2^2 = 4 \not\equiv -1 \pmod{15}, \quad (-1 \equiv 14 \pmod{15}).$$

Então:

$$p_1 = \gcd(4 - 1, 15) = \gcd(3, 15) = 3,$$

$$p_2 = \gcd(4 + 1, 15) = \gcd(5, 15) = 5.$$

Achamos os fatores 3 e 5.

Problema de estimativa de fase

Temos:

- um operador unitário U (representado por um circuito quântico);
- um estado $|\psi\rangle$ tal que:

$$U|\psi\rangle = e^{2\pi i \theta} |\psi\rangle,$$

ou seja, $|\psi\rangle$ é um **autovetor** de U ;

- queremos estimar o número real $\theta \in [0, 1]$.

Leitura da equação:

“Aplicar U em $|\psi\rangle$ apenas multiplica o estado por uma fase $e^{2\pi i \theta}$, sem mudar a direção do vetor no espaço de Hilbert.”

Objetivo numérico da QPE

Queremos obter uma aproximação discreta de θ .

- Escolhemos um número de bits m para a precisão.
- QPE nos dá um inteiro y com:

$$\theta \approx \frac{y}{2^m}, \quad y \in \{0, 1, \dots, 2^m - 1\}.$$

Ou seja, ao medir m qubits, esperamos que o resultado binário de y represente uma aproximação da fase θ .

Pergunta natural:

- “Como o circuito faz essa aproximação aparecer nos m qubits?”

Círculo padrão de QPE (visão geral)

Estrutura:

- **Registrador de controle:** m qubits, iniciados em $|0\rangle^{\otimes m}$.
- **Registrador alvo:** n qubits, iniciados em $|\psi\rangle$.

Passos:

- ① Aplicar Hadamard H em cada qubit de controle \Rightarrow superposição uniforme.
- ② Para cada qubit de controle j (do menos significativo ao mais significativo):

aplicar U^{2^j} controlado por esse qubit.

- ③ Aplicar QFT inversa (QFT^{-1}) no registrador de controle.
- ④ Medir os m qubits de controle \Rightarrow resultado y .

Leitura da ação dos unitários controlados

Quando aplicamos U^{2^j} controlado, estamos fazendo:

$$\begin{aligned} |0\rangle |\psi\rangle &\mapsto |0\rangle |\psi\rangle, \\ |1\rangle |\psi\rangle &\mapsto |1\rangle U^{2^j} |\psi\rangle. \end{aligned}$$

Como $U |\psi\rangle = e^{2\pi i \theta} |\psi\rangle$:

$$U^{2^j} |\psi\rangle = e^{2\pi i \theta 2^j} |\psi\rangle.$$

Logo:

$$|1\rangle |\psi\rangle \mapsto e^{2\pi i \theta 2^j} |1\rangle |\psi\rangle.$$

Ou seja:

“Cada qubit de controle acumula uma fase proporcional a $2^j \theta$ quando está em 1.”

Por que a QFT inversa aparece?

Depois dos unitários controlados, o registrador de controle está em uma superposição de estados da forma:

$$\frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} e^{2\pi i \theta x} |x\rangle.$$

Isso é uma superposição com **estrutura periódica de fase**.

A QFT inversa QFT^{-1} sobre os m qubits transforma essa distribuição de fases em uma distribuição de **amplitudes concentradas** em um valor y próximo de $2^m\theta$.

Intuição:

- QFT é o “radar” de periodicidade;
- ela “lê” a frequência escondida na fase e devolve um valor aproximado da fase em forma de número binário.

Problema de determinação de ordem

Relembrando:

- Dado N e $a \in \mathbb{Z}_N^*$, queremos a ordem r :

$$r = \min\{r > 0 : a^r \equiv 1 \pmod{N}\}.$$

Ligaçāo com QPE:

Se construirmos um operador unitário U cujo autovalor dependa de $1/r$, então QPE pode estimar $1/r$ e, daí, recuperar r .

Definindo o operador M_a

No espaço cujos estados básicos correspondem a \mathbb{Z}_N , definimos:

$$M_a |x\rangle = |ax \bmod N\rangle, \quad x \in \mathbb{Z}_N.$$

Leitura:

“ M_a pega o estado que representa o número x e o manda para o estado que representa $(ax \bmod N)$.”

Propriedades:

- M_a é uma **permutação** dos estados básicos;
- toda permutação é representada por uma matriz unitária;
- logo, M_a é um operador unitário: pode ser usado em circuito quântico.

Autovetores de M_a e conexão com r

Para um dado $a \in \mathbb{Z}_N^*$ com ordem r , existem vetores próprios de M_a da forma:

$$|\psi_j\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \omega_r^{-jk} |a^k \bmod N\rangle, \quad j = 0, 1, \dots, r-1,$$

onde

$$\omega_r = e^{2\pi i/r}.$$

E vale:

$$M_a |\psi_j\rangle = \omega_r^j |\psi_j\rangle = e^{2\pi ij/r} |\psi_j\rangle.$$

Leitura:

“Cada $|\psi_j\rangle$ é um autovetor de M_a , com autovalor $e^{2\pi ij/r}$. ”

Logo, a fase θ_j associada é:

Escolha do estado inicial $|1\rangle$

Queremos aplicar QPE usando $U = M_a$.

Idealmente, precisaríamos de um autovetor $|\psi_j\rangle$ como entrada. Mas preparar $|\psi_j\rangle$ diretamente é difícil.

Observação:

$$|1\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} |\psi_j\rangle .$$

Interpretação:

- o estado computacional $|1\rangle$ é uma superposição **uniforme** de todos os autovetores $|\psi_j\rangle$;
- isso significa que, ao rodar QPE com $|1\rangle$ como estado alvo, é como se, em cada execução, tivéssemos “escolhido” aleatoriamente um j entre 0 e $r - 1$.

Resultado da QPE com entrada $|1\rangle$

Rodando QPE com:

$$U = M_a, \quad |\psi\rangle = |1\rangle,$$

a saída (no registrador de controle) é um valor aproximado de:

$$\theta = \frac{j}{r},$$

para algum j uniformemente escolhido em $\{0, 1, \dots, r - 1\}$.

Ou seja, a medida dos m qubits superiores produz:

$$\theta \approx \frac{y}{2^m} \approx \frac{j}{r}.$$

A partir de θ , tentamos reconstruir $\frac{j}{r}$ usando **frações contínuas** e extrair r .

Arquitetura do circuito quântico de Shor

Registradores:

- **Registrador de controle:** m qubits, para estimar a fase (QPE/QFT);
- **Registrador alvo:** n qubits, para armazenar valores em \mathbb{Z}_N .

Estado inicial:

$$|0\rangle^{\otimes m} |1\rangle .$$

Passos principais:

- ① Hadamards em todos os qubits de controle.
- ② Aplicar expoentes modulares controlados: $M_a^{2^k}$.
- ③ Aplicar QFT inversa no registrador de controle.
- ④ Medir o registrador de controle \Rightarrow obter aproximação de $\theta = j/r$.

Expoente x em binário e exponenciação modular

Escrevemos o expoente x em binário:

$$x = x_0 2^0 + x_1 2^1 + \cdots + x_{m-1} 2^{m-1},$$

com $x_j \in \{0, 1\}$.

A potência modular é:

$$a^x \bmod N = a^{x_0 2^0} a^{x_1 2^1} \cdots a^{x_{m-1} 2^{m-1}} \bmod N.$$

Cada fator $a^{2^j} \bmod N$ é pré-calculado classicamente, e implementamos a transformação:

$$|y\rangle \mapsto |(y \cdot a^{2^j}) \bmod N\rangle,$$

controlada pelo bit x_j .

Resumo da evolução quântica

Em termos de estados:

- ➊ Inicial:

$$|\psi_0\rangle = |0\rangle^{\otimes m} |1\rangle.$$

- ➋ Após Hadamards nos m qubits de controle:

$$|\psi_1\rangle = \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x\rangle |1\rangle.$$

- ➌ Após exposições modulares controladas:

$$|\psi_2\rangle = \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x\rangle |a^x \bmod N\rangle.$$

- ➍ Interpretando: a função $f(x) = a^x \bmod N$ está aplicada a todos os x em superposição.

Periodicidade em x

Sabemos que a ordem r satisfaz:

$$a^{x+r} \equiv a^x \pmod{N}.$$

Logo:

$$f(x+r) = f(x).$$

Consequência:

- os valores $f(x)$ se repetem com período r ;
- o segundo registrador guarda uma função periódica de x ;
- queremos “descobrir” esse período r usando interferência e QFT.

Medição do registrador alvo

Quando medimos o registrador alvo (segundo registrador):

- obtemos um valor $y_0 = a^{x_0} \bmod N$ para algum x_0 ;
- o primeiro registrador colapsa para uma superposição dos x que satisfazem:

$$f(x) = a^x \bmod N = y_0.$$

Devido à periodicidade:

$$x = x_0 + kr, \quad k = 0, 1, \dots$$

Então o estado do primeiro registrador vira (até normalização):

$$|\phi\rangle = \sum_k |x_0 + kr\rangle.$$

Aplicando QFT no primeiro registrador

Aplicamos F_{2^m} (QFT) no primeiro registrador:

$$F_{2^m} |\phi\rangle = \frac{1}{\sqrt{M2^m}} \sum_{y=0}^{2^m-1} e^{2\pi i x_0 y / 2^m} \left(\sum_{k=0}^{M-1} e^{2\pi i k r y / 2^m} \right) |y\rangle,$$

onde M é o número de termos na soma em k .

Ponto central: a soma

$$S(y) = \sum_{k=0}^{M-1} e^{2\pi i k r y / 2^m}$$

fica grande quando $\frac{y}{2^m}$ é próximo de $\frac{j}{r}$, para algum inteiro j .

Medida de y e aproximação de j/r

Devido à forma de $S(y)$:

- a probabilidade de medir cada y é proporcional a $|S(y)|^2$;
- $|S(y)|^2$ é grande quando:

$$\frac{y}{2^m} \approx \frac{j}{r},$$

para algum $j \in \{0, \dots, r - 1\}$.

Logo, a medida nos dá um y tal que:

$$\frac{y}{2^m} \approx \frac{j}{r}.$$

A partir disso, usamos matemática clássica (frações contínuas) para recuperar r .

Escolha de parâmetros: $N = 15$, $a = 2$

Queremos fatorar:

$$N = 15.$$

Primeiro passo clássico:

$$\gcd(2, 15) = 1 \Rightarrow 2 \in \mathbb{Z}_{15}^*.$$

Objetivo quântico:

- encontrar a ordem r de $a = 2$ módulo 15;
- sabemos analiticamente que $r = 4$, mas o circuito deve “descobrir” isso.

No tutorial Qiskit:

- usam-se 4 qubits para representar números de 0 a 15 (Registrador alvo);
- usam-se 8 qubits no Registrador de controle para estimar a fase com precisão.

Operador M_2 para $N = 15$ (visão conceitual)

Definimos:

$$M_2 |x\rangle = |2x \bmod 15\rangle, \quad x = 0, \dots, 14.$$

Isso é uma **permutação** dos 15 estados $|0\rangle, \dots, |14\rangle$.

Exemplo de ação:

$$M_2 |0\rangle = |0\rangle,$$

$$M_2 |1\rangle = |2\rangle,$$

$$M_2 |2\rangle = |4\rangle,$$

$$M_2 |4\rangle = |8\rangle,$$

$$M_2 |8\rangle = |1\rangle,$$

mostrando um ciclo $(1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 1)$.

Interpretação: M_2 apenas “embaralha” os estados da base, sem criar superposição por si só.

Implementação de M_2 com SWAPs

Como $N = 15$ é pequeno:

- podemos inspecionar a permutação de M_2 e perceber que ela é construída por algumas trocas (SWAPs) de qubits;
- por exemplo, o tutorial Qiskit usa 4 qubits e constrói M_2 com uma sequência de portas SWAP que realiza exatamente a permutação desejada;
- a versão controlada de M_2 (para QPE) é obtida promovendo o circuito a **porta controlada**.

Ponto didático:

- o objetivo aqui é entender o **que** M_2 faz; o **como** detalhado (porta a porta) é delegado ao transpiler do Qiskit em implementações práticas.

Operadores $M_{2^{2k}}$ (exponenciação modular)

Para QPE, precisamos de potências U^{2^k} .

No contexto de Shor:

$$U = M_a, \quad U^{2^k} = M_a^{2^k} = M_{a^{2^k} \bmod N}.$$

Para $N = 15$, $a = 2$:

$$b_k = a^{2^k} \bmod 15.$$

Exemplo numérico (como no tutorial):

$$b_0 = 2, \quad b_1 = 4, \quad b_2 = 1, \quad b_3 = 1, \dots$$

Leitura:

- só precisamos implementar M_2 , M_4 e M_1 ;
- M_1 é a identidade (não faz nada);
- M_4 também é uma permutação que pode ser construída com SWAPs.

Saída típica em simulador ideal

Em um simulador ideal com 1024 shots, o tutorial obtém algo como:

- bitstrings mais prováveis no registrador de controle:

00000000, 01000000, 10000000, 11000000.

Convertendo para decimal:

0, 64, 128, 192.

Convertendo para fases:

$$\frac{0}{256} = 0, \quad \frac{64}{256} = 0,25, \quad \frac{128}{256} = 0,5, \quad \frac{192}{256} = 0,75.$$

Esses valores são aproximações de:

$$0, \quad \frac{1}{4}, \quad \frac{2}{4}, \quad \frac{3}{4}.$$

Reconstruindo r com frações contínuas

Sabemos que cada fase é da forma:

$$\theta = \frac{k}{r}.$$

Usamos frações contínuas para aproximar cada fase por uma fração racional com denominador $\leq N$:

$$0 \rightarrow \frac{0}{1}, \quad 0,25 \rightarrow \frac{1}{4}, \quad 0,5 \rightarrow \frac{1}{2}, \quad 0,75 \rightarrow \frac{3}{4}.$$

Os denominadores candidatos a r são:

$$1, 4, 2, 4.$$

Com alta probabilidade, o valor correto é $r = 4$.

Conectando de volta à fatoração

Uma vez estimado $r = 4$:

- verificamos se r é par (sim);
- calculamos:

$$x = a^{r/2} \bmod N = 2^2 \bmod 15 = 4;$$

- calculamos:

$$\gcd(x - 1, N) = \gcd(3, 15) = 3, \quad \gcd(x + 1, N) = \gcd(5, 15) = 5.$$

Resultado:

$$15 = 3 \cdot 5.$$

Essa última etapa é totalmente clássica.

Execução em hardware real (visão conceitual)

Em hardware real (ex.: `ibm_marrakesh`):

- o circuito é **transpilado** para o conjunto nativo de portas do backend;
- o **transpiler** tenta minimizar:
 - a profundidade de portas de 2 qubits;
 - o número total de portas;
 - erros induzidos pela arquitetura física.
- técnicas adicionais:
 - **dynamical decoupling** (sequências de pulsos para mitigar decoerência);
 - **gate twirling** (randomização de portas para transformar erros coerentes em erros mais simples de tratar estatisticamente).

Em resultados reais:

- vemos os mesmos bitstrings dominantes que no simulador;
- mas com vazamento de probabilidade para outros resultados, devido a ruído.

Pós-processamento com dados de hardware

Estratégia:

- rodar o circuito várias vezes (shots);
- coletar a distribuição de bitstrings medidos;
- focar nos bitstrings mais frequentes (acima de um certo limiar);
- converter cada bitstring:
 - de binário para decimal;
 - de decimal para fase $y/2^m$;
 - de fase para fração $\frac{k}{r}$ via frações contínuas;
- extrair candidatos a r ;
- usar r para tentar fatorar N (como fizemos com $N = 15$).

Limitações de escala

Criptografia RSA típica:

- tamanhos de chave: 2048 a 4096 bits;
- fatorar N de 2048 bits com Shor exigiria:
 - **milhões** de qubits físicos (contando correção de erros);
 - profundidades de circuito da ordem de **bilhões** de portas.

Situação atual:

- hardware NISQ (Noisy Intermediate-Scale Quantum) ainda está muito longe desse regime;
- demos como $N = 15, 21, 35$ são **provas de conceito**, não ataques práticos.

Mas:

A estrutura de Shor mostra como um computador quântico escalável poderia, em princípio, quebrar RSA.

Resumo conceitual de Shor

Parte clássica:

- escolher a ;
- calcular $\gcd(a, N)$;
- se $\gcd(a, N) = 1$, depender de achar a ordem r de a módulo N ;
- de r extrair fatores via $\gcd(a^{r/2} \pm 1, N)$.

Parte quântica:

- usar QPE com $U = M_a$;
- estimar fases da forma $\theta = k/r$;
- usar frações contínuas para obter r .

Conexão: Fatoração \Rightarrow ordem \Rightarrow estimativa de fase.

Mensagens-chave da aula

- O algoritmo de Shor **não** “adivinha” fatores: ele transforma fatoração em um problema de **periodicidade**.
- A função periódica é $f(x) = a^x \text{ mod } N$, cujo período é a ordem r .
- O computador quântico:
 - prepara superposições de muitos x ao mesmo tempo;
 - codifica a periodicidade em **fase quântica**;
 - usa QFT para transformar essa fase em picos mensuráveis.
- O computador clássico:
 - faz frações contínuas para recuperar r ;
 - usa gcd para encontrar fatores de N .

Para estudar mais

Sugestões de aprofundamento:

- curso *Fundamentals of Quantum Algorithms* (John Watrous);
- documentação do Qiskit sobre:
 - Shor;
 - QPE;
 - QFT;
- artigos de demonstração do algoritmo de Shor em hardware real (NMR, fótons, íons aprisionados, IBM Quantum).

O principal ganho de Shor:

usar interferência quântica para descobrir padrões numéricos (ordem) que são inacessíveis em tempo viável classicamente.