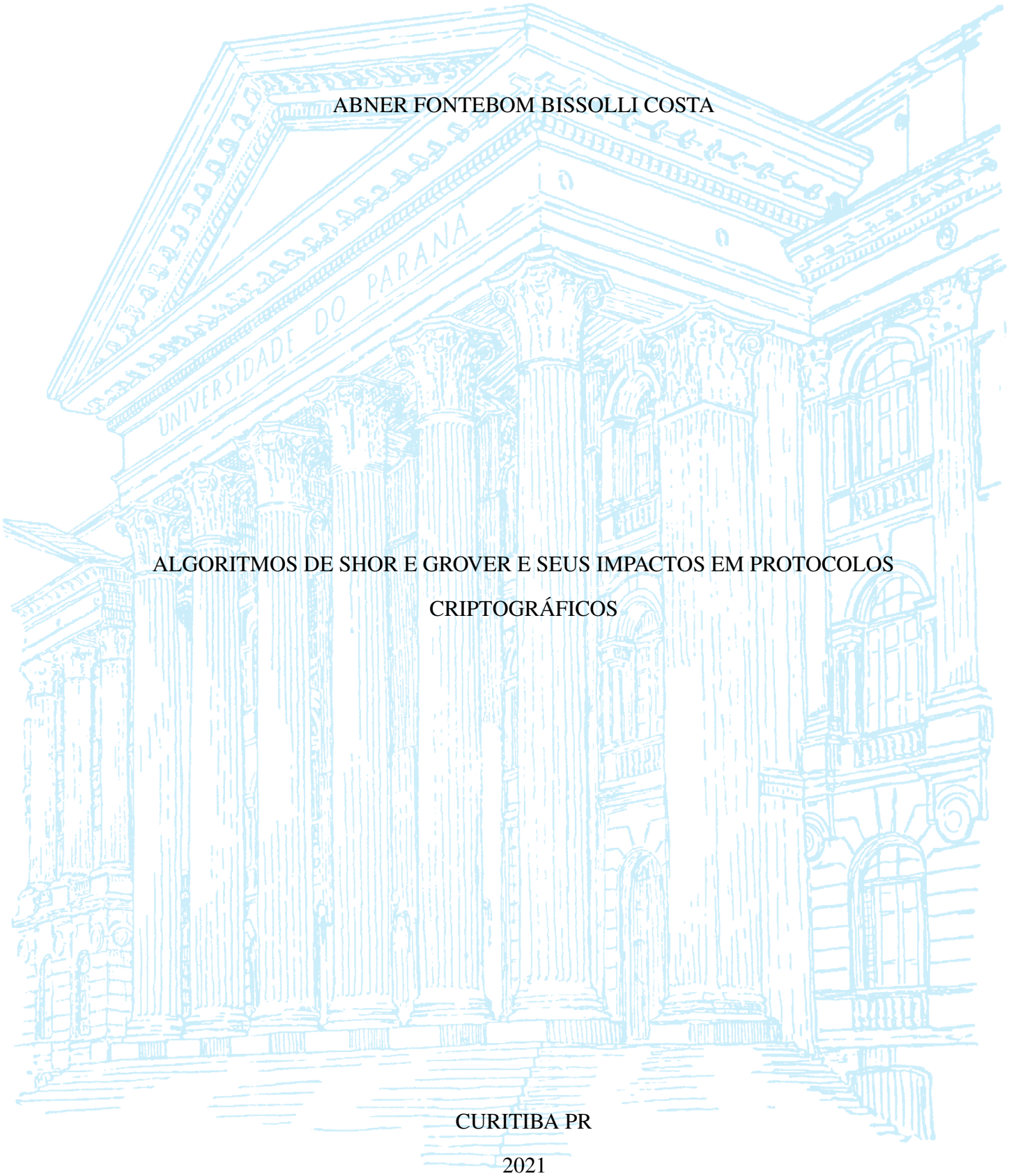
UNIVERSIDADE FEDERAL DO PARANÁ

ABNER FONTEBOM BISSOLLI COSTA



ALGORITMOS DE SHOR E GROVER E SEUS IMPACTOS EM PROTOCOLOS CRIPTOGRÁFICOS

CURITIBA PR

2021

ABNER FONTEBOM BISSOLLI COSTA

ALGORITMOS DE SHOR E GROVER E SEUS IMPACTOS EM PROTOCOLOS CRIPTOGRÁFICOS

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Dr. Murilo V. G. da Silva.

CURITIBA PR

2021

Dedico este trabalho ao meu querido pai Romildo Costa, por me incentivar desde pequeno ao estudo da matemática e ciência.

AGRADECIMENTOS

Ao professor orientador Murilo Vicente Gonçalves da Silva o qual me proporcionou o primeiro contato com computação quântica e me auxiliou ao longo do desenvolvimento de meu trabalho de graduação.

Aos professores que se dispuseram a corrigir e avaliar meu trabalho, assim como aos professores que me instruíram ao decorrer do meu curso de graduação e ao longo de toda minha jornada acadêmica.

Aos meu pais e familiares que me educaram e permitiram que eu tivesse a oportunidade e o privilégio de realizar o curso de bacharelado em ciência da computação.

A minha namorada que me auxiliou e apoiou durante todo o período que estive realizando este trabalho, e a todos os meus amigos que me apoiaram em minhas decisões.

RESUMO

Atualmente, a criptografia desempenha um papel fundamental na segurança da informação, garantindo a confidencialidade, integridade e autenticidade em comunicações via Internet. Com o avanço da internet das coisas, seu papel torna-se cada vez mais crucial para o funcionamento e estabilidade de sistemas computacionais.

Em 1980, com o avanço das descobertas científicas e da mecânica quântica, foi proposto, pelo físico Paul Benioff, um modelo da máquina de Turing que utiliza estados quânticos da matéria como sua unidade básica de medida, o qubit. Utilizando este modelo, foram desenvolvidos algoritmos capazes de resolver problemas matemáticos, considerados difíceis, de forma mais rápida que algoritmos baseados na máquina de Turing original, em alguns casos reduzindo exponencialmente o custo computacional. Estes problemas matemáticos são fundamentais para o funcionamento de sistemas criptográficos, e portanto os algoritmos quânticos capazes de solucioná-los são uma ameaça à segurança da informação.

Este presente estudo é o resultado de uma pesquisa bibliográfica sobre os principais algoritmos quânticos que impactam na criptografia atual. Nele é demonstrado de forma detalhada os algoritmos propostos por Peter Shor e Lov Grover, assim como a relações de tais algoritmos com os protocolos criptográficos RSA, AES e a troca de chaves Diffie-Hellman-Merkle.

Palavras-chave: Criptografia. Computação Quântica. Shor. Grover. Fatoração. Logaritmo Discreto. RSA. AES. Diffie-Hellman-Merkle.

ABSTRACT

Currently, cryptography plays a fundamental role in information security, ensuring confidentiality, integrity, and authenticity in Internet communications. With the advancement of the internet of things, its role becomes increasingly crucial for the functioning and stability of computer systems.

In 1980, with the advance of scientific discoveries and quantum mechanics, the physicist Paul Benioff proposed a model of the Turing machine that uses quantum states of matter as its basic unit of measurement, the qubit. Using this model were developed algorithms capable of solving mathematical problems, considered difficult, faster than algorithms based on the original Turing machine, in some cases exponentially reducing the computational cost. These mathematical problems are fundamental to the functioning of cryptographic systems, and therefore the quantum algorithms capable of solving them are a threat to information security.

This present study is the result of a literature review on the main quantum algorithms that impact on current cryptography. It shows in detail the algorithms proposed by Peter Shor and Lov Grover, as well as the relationship of such algorithms with the cryptographic protocols RSA, AES, and the exchange of Diffie-Hellman-Merkle keys.

Keywords: Cryptography. Quantum Computing. Shor. Grover. Factorization. Discrete logarithm. RSA. AES. Diffie-Hellman-Merkle.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ACRÔNIMOS

MMC	Mínimo Múltiplo Comum
MDC	Máximo Divisor Comum
DFT	Transformada Discreta de Fourier
IDFT	Transformada Discreta Inversa de Fourier
QFT	Transformada Quântica de Fourier
IQFT	Transformada Quântica Inversa de Fourier
RSA	Rivest, Shamir e Adleman
AES	*Advanced Encryption Standard*
PLD	Problema do Logaritmo Discreto
PLDP	Subproblema Problema do Logaritmo Discreto

LISTA DE SÍMBOLOS

e	Número de Euler
π	Número Pi
i	$\sqrt{-1}$
$\sum$	Símbolo matemático de uma sequência de somas
$\prod$	Símbolo matemático de uma sequência de produtos
$\otimes$	Símbolo matemático de produto tensorial e sequência de produtos tensoriais
log	Símbolo matemático de logaritmo na base 2
$\forall$	Símbolo matemático para todo
$\exists$	Símbolo matemático de existe
$\nexists$	Símbolo matemático de não existe
$\in$	Símbolo matemático de pertence
$\notin$	Símbolo matemático de não pertence
$\subset$	Símbolo matemático de está contido
$\not\subset$	Símbolo matemático de não está contido
$\mid$	Símbolo matemático de divide
$\nmid$	Símbolo matemático de não divide
$\approx$	Símbolo matemático de aproximadamente
$\mapsto$	Símbolo matemático de mapeamento
$\therefore$	Símbolo matemático de portanto
$\Rightarrow$	Símbolo matemático de implica
$\mathbb{Z}$	Conjunto dos número inteiros
$\mathbb{Z}_{>0}$	Conjunto dos número inteiros positivos
$\mathbb{Z}_{\geq 0}$	Conjunto dos número inteiros positivos com zero incluso
$\mathbb{Q}$	Conjunto dos número racionais
$\mathbb{R}$	Conjunto dos número reais
$\mathbb{C}$	Conjunto dos número complexos
$\mathbb{Z}_n^\times$	Conjunto de números inteiros menores que n e que são coprimos de n, onde $n \in \mathbb{Z}_{>0}$
$\mathbb{P}$	Função de probabilidade
α	alfa, primeira letra do alfabeto grego
β	beta, segunda letra do alfabeto grego
γ	gama, terceira letra do alfabeto grego
δ	delta, quarta letra do alfabeto grego
ϵ	epsilon, quinta letra do alfabeto grego

ζ	theta, sexta letra do alfabeto grego
θ	theta, oitava letra do alfabeto grego
λ	lambda, décima primeira letra do alfabeto grego
μ	mu, décima segunda letra do alfabeto grego
ρ	rho, décima sétima letra do alfabeto grego
ϕ	phi, vigésima primeira letra do alfabeto grego
Φ	Phi, vigésima primeira letra do alfabeto grego
φ	varphi, variante da vigésima primeira letra do alfabeto grego
ψ	psi, penúltima letra do alfabeto grego

SUMÁRIO

1 INTRODUÇÃO

A partir do desenvolvimento da máquina de Turing, proposta por Alan Turing em 1936, ocorreu uma grande revolução tecnológica, levando à criação dos primeiros computadores. Atualmente a maior parte da população mundial tem acesso à computadores e à rede mundial de computadores, tornando-se o meio de comunicação mais utilizado e transformando o mundo para sempre. Dado que informações confidenciais e valiosas são transmitidas em redes de computadores e na rede mundial de computadores, a segurança da informação transmitida nestas redes torna-se um assunto de extrema relevância para todos. Para garantir a segurança da informação, foram criados modelos criptográficos para impedir que a informação seja acessada por usuários não autorizados. Em 1980, com o avanço das descobertas científicas e da mecânica quântica, foi proposto pelo físico Paul Benioff um modelo da máquina de Turing que utiliza estados quânticos da matéria como sua unidade básica de medida, o qubit, dando origem à um novo modelo de computador, o computador quântico.

Peter Shor em 1994 e Lov Grover em 1996 propuseram algoritmos que operam em computadores quânticos. Estes algoritmos realizam ataques a protocolos criptográficos considerados seguros, ameaçando o funcionamento de tais protocolos. Com o crescente avanço do desenvolvimento dos computadores quânticos que utilizam múltiplos qubits, os algoritmos projetados para este modelo de computação se tornam cada vez mais palpáveis, fazendo com que os ataques de Shor e Grover se tornem ameaças iminentes à criptografia vigente.

O objetivo deste estudo é reunir o necessário para a compreensão do funcionamento dos algoritmos que ameaçam a confiabilidade de protocolos criptográficos considerados seguros e evidenciar suas relações com tais protocolos. Os algoritmos propostos por Shor solucionam os problemas de fatoração e do logaritmo discreto em tempo polinomial, realizando assim ataques ao protocolo criptográfico RSA e ao protocolo de troca de chaves de Diffie-Hellman-Merkle, respectivamente. Similarmente, o algoritmo de Grover reduz polinomialmente o custo computacional para resolver o problema de busca não estruturada, desta forma considera-se que este algoritmo realiza um ataque ao protocolo AES.

Neste trabalho estão detalhadas as etapas dos algoritmos em um nível de detalhe que normalmente não é encontrando em materiais voltados para graduação. Adicionalmente, é importante observar que o uso do Algoritmo de Shor no problema do Logaritmo Discreto não é tão difundido em materiais introdutórios. Até onde o autor deste texto tem conhecimento, não há material em língua portuguesa abordando tal algoritmo quântico para o problema do logaritmo discreto.

Este estudo é resultado de uma pesquisa bibliográfica, a qual reúne conteúdos presentes nos livros “Quantum Computation and Quantum Information: 10th Anniversary Edition” [NC11] e “An Introduction to Quantum Computing” [KLM07]. A partir deste material foram adaptadas descrições e análises de algoritmos quânticos, demonstrando com clareza cada uma de suas etapas, assim como adicionando outras informações que auxiliam a compreensão de tais algoritmos.

Esta monografia está organizada em seis capítulos principais, iniciando com uma breve introdução ao modelo de computação quântica, explicando seus fundamentos, o modelo matemático que o representa, assim como operações básicas da computação quântica. O capítulo seguinte aborda modelos e protocolos criptográficos, permitindo distinguir criptografia simetria de assimétrica, além da compreensão do funcionamento e vulnerabilidade dos protocolos criptográficos RSA, AES e a troca de chaves de Diffie-Hellman-Merkle. Os capítulos seguintes trazem

os algoritmos quânticos estudados, os quais possuem custo computacional significativamente inferior a algoritmos projetados para computadores clássicos.

O primeiro algoritmo apresentado é a transformação quântica de Fourier, descrita ao longo do Capitulo 4, o qual monstra-se uma ferramenta extremamente útil para muitos algoritmos quânticos sendo uma rotina essencial para o funcionamento dos algoritmos propostos por Peter Shor. O Capítulo 5 contém três algoritmos quânticos que servirão como base para a construção e entendimento dos algoritmos de Shor e Grover, explicando detalhadamente seus funcionamentos e custos computacionais. Os dois capítulos restantes, Capítulo 6 "Algoritmos de Shor"e Capítulo 7 "Algoritmos de Grover", abordam os principais algoritmos deste documento contendo os algoritmos propostos por Peter Shor e Lov Grover, respectivamente, assim como suas relações com os protocolos criptográficos apresentados anteriormente, bem como seus respectivos custos computacionais. Finalmente as considerações finais deste trabalho se encontram no Capítulo 8.

2 FUNDAMENTOS DA COMPUTAÇÃO QUÂNTICA

Com o avanço das descobertas científicas e da mecânica quântica, o físico Paul Benioff propôs em 1980 um modelo quântico da maquina de Turing [Ben80]. Neste modelo matemático, a definição da unidade básica de informação, assim como as possíveis operações que podem ser empregadas na manipulação da informação, levam em consideração as leis da física que descrevem a matéria no seu nível mais fundamental, em particular estados quânticos da matéria. Este modelo de computação recebe o nome de computação quântica e possui ferramentas que não estão presentes no modelo de computação clássica.

Este capítulo tem como propósito apresentar os fundamentos da computação quântica, para que os algoritmo quânticos apresentados nos capítulos subsequentes possam ser compreendidos. As informações apresentadas neste capítulo compreendem o necessário para o entendimento deste trabalho, no entanto informações adicionais podem ser encontrada nos livros "Quantum Computation and Quantum Information: 10th Anniversary Edition" [NC11] e "An Introduction to Quantum Computing" [KLM07].

2.1 BIT QUÂNTICO

Bit quântico, denominado de qubit $[\text{'kju.bɪt}]$, é a unidade básica da informação em computadores quânticos, análogo aos bit, o qual é a unidade básica da informação em computadores clássicos. Contudo existe uma grande diferença entre um qubit e um bit, um bit pode estar apenas em dois possíveis estados, 0 (desligado) ou 1 (ligado), enquanto um qubit pode estar em uma superposição dos estados 0 e 1, a qual gera uma infinidade de possíveis estados. Na mecânica quântica é dito que o qubit é um sistema quântico de dois estados.

Para a implementação física de um computador quântico, poderiam ser utilizados, por exemplo, o spin de um elétron, o qual possui dois estados de excitação, *spin up* e *spin down* e a polarização de um fóton, a qual pode estar nos estados de polarização vertical e polarização horizontal. A rigor, de acordo com as leis da mecânica quântica, qualquer objeto físico que possa ser interpretado como um sistema quântico de dois estados, poderia ser utilizado como substrato para implementar um qubit, uma vez que este deve respeitar as mesmas leis matemáticas. A questão de qual objeto físico é mais adequado para a implementação real de computadores quânticos é uma questão de engenharia.

Ao longo deste trabalho a palavra qubit se refere tanto ao objeto físico como à sua representação em um modelo matemático, dependendo exclusivamente do contexto no qual a palavra está inserida.

2.1.1 Notação de Dirac

A notação de Dirac, também conhecida como notação *bra-ket*, foi desenvolvida por Paul Dirac em 1939 [Dir39], para a representação de sistemas quânticos e atualmente é a representação padrão para a computação quântica. Esta notação utiliza colchetes angulares "$\langle$", "$\rangle$" e a barra vertical "|", para construir um *bra* ou um *ket*. Tanto o *bra* como o *ket* denotam vetores unitários de números complexos, podendo ser representado por matrizes linha e é escrito na forma $\langle\psi|$, enquanto o *ket* é utilizado para representar matrizes coluna e é escrito na forma $|\psi\rangle$, ambas matrizes de números complexos. Nesta notação, um estado de qubit é representado por um *ket*, ou então uma superposição linear de dois estados ortonormais, pertencentes à uma base. cada

estado de uma base é representada por um *ket*. Uma base comumente utilizada é a base $\{|0\rangle, |1\rangle\}$, chamada de base computacional, onde:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Utilizando a base $\{|0\rangle, |1\rangle\}$ é possível escrever qualquer superposição de um qubit. Seja $|\psi\rangle$ um estado arbitrário para um único qubit e $\alpha, \beta \in \mathbb{C}$, tais que $|\alpha|^2 + |\beta|^2 = 1$, então:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}.$$

Seja $\alpha, \beta \in \mathbb{C}$, então α, assim como β, pode ser escrito da seguinte forma $\alpha = a + bi$, onde $a, b \in \mathbb{R}$. De maneira similar, o conjugado de α, α^*, é escrito na forma $\alpha^* = a - bi$.

Na notação de Dirac, um *ket* é o trasposto conjugado de um *bra*, e vice-versa.

$$|\psi\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}; \quad \langle\psi| = \begin{bmatrix} \alpha^* & \beta^* \end{bmatrix}.$$

Na maioria dos casos, algoritmos quânticos utilizam mais de um qubit para realizar suas operações. Sejam $|\psi_1\rangle$ e $|\psi_2\rangle$ estados arbitrários de dois qubit e $|\psi\rangle$ o estado do sistema composto por estes. O estado $|\psi\rangle$ é escrito como o produto tensorial $|\psi_1\rangle \otimes |\psi_2\rangle$. Normalmente utiliza-se a notação simplificada $|\psi_1\rangle|\psi_2\rangle$, ou então $|\psi_1\psi_2\rangle$, para se representar este produto tensorial.

Faça os estados $|\psi_1\rangle$ e $|\psi_2\rangle$ da seguinte forma:

$$|\psi_1\rangle = \alpha_1|0\rangle + \beta_1|1\rangle.$$

$$|\psi_2\rangle = \alpha_2|0\rangle + \beta_2|1\rangle.$$

Então:

$$\begin{aligned} |\psi\rangle &= |\psi_1\rangle|\psi_2\rangle \\ &= (\alpha_1|0\rangle + \beta_1|1\rangle) \otimes (\alpha_2|0\rangle + \beta_2|1\rangle) \\ &= \begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix} \otimes \begin{bmatrix} \alpha_2 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} \alpha_1\alpha_2 \\ \alpha_1\beta_2 \\ \beta_1\alpha_2 \\ \beta_1\beta_2 \end{bmatrix} \\ &= \alpha_1\alpha_2|00\rangle + \alpha_1\beta_2|01\rangle + \beta_1\alpha_2|10\rangle + \beta_1\beta_2|11\rangle. \end{aligned} \tag{2.1}$$

Esta notação pode ser generalizada para descrever superposições de n qubits, onde n é um número inteiro positivo.

Seja $s \in \mathbb{Z}_{\geq 0}$, o qual pode ser escrito pelo vetor binário $[s_1 s_2 \dots s_n]$, onde $s_i \in \{0, 1\}$ e $i \in \{1, 2, \cdots, n\}$, então:

$$|s\rangle = |s_1\rangle|s_2\rangle \dots |s_n\rangle.$$

O conjunto de todos os 2^n possíveis estados $|s\rangle$ é chamado de base computacional, de forma que um estado arbitrário $|\psi\rangle$ de n qubits pode ser expresso, como mostrado a seguir:

$$|\psi\rangle^n = \sum_{s=0}^{2^n-1} a_s|s\rangle, \quad a_s \in \mathbb{C}, n \in \mathbb{Z}_{>0}, \sum_{s=0}^{2^n-1} |a_s|^2 = 1.$$

Também é comum utilizar a seguinte notação, menos carregada, para estados quânticos:

$$|\psi\rangle = \sum_s a_s|s\rangle, \quad a_s \in \mathbb{C}, \sum_s |a_s|^2 = 1.$$

2.1.2 Representação na esfera de Bloch

A esfera de Bloch nomeada em homenagem ao físico Felix Bloch, é utilizada para representar um qubit, um sistema quântico de dois níveis. Esta representação utiliza uma espaço tridimensional definido por uma esfera unitária, na qual todos os possíveis estados de um qubit podem ser descritos por meio de uma seta, a qual inicia no centro da esfera e aponta para qualquer direção.

O eixo vertical é definido como o eixo z, onde o topo deste eixo representa o estado $|0\rangle$, e sua base representa o estado $|1\rangle$. Na Figura 2.1 é possível visualizar a representação de um qubit na esfera de Bloch.

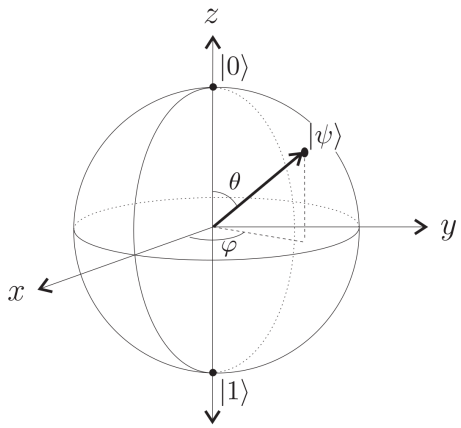


Figura 2.1: Estado de um qubit representado na esfera de Bloch [KLM07].

Utilizando a base $\{|0\rangle, |1\rangle\}$, é possível definir as bases correspondentes aos eixos x e y na esfera de Bloch.

Bases do eixo y:

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}; \quad |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

Bases do eixo x:

$$|i+\rangle = \frac{|0\rangle + i|1\rangle}{\sqrt{2}}; \quad |i-\rangle = \frac{|0\rangle - i|1\rangle}{\sqrt{2}}.$$

Na Figura 2.2 é possível visualizar como as bases dos eixos x e y se enquadram na esfera de Bloch.

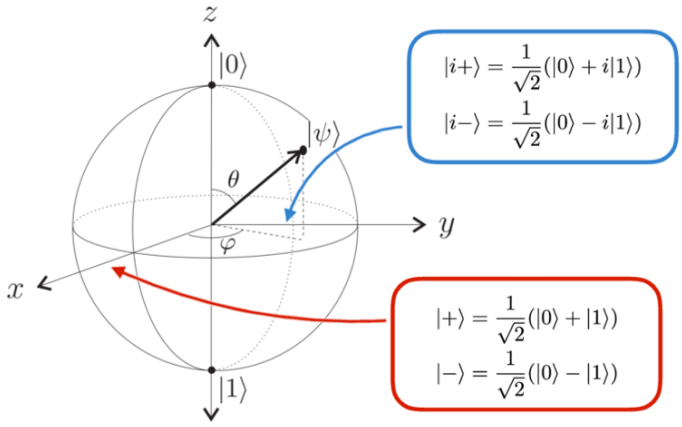


Figura 2.2: Estado de um qubit representado na esfera de Bloch, contendo as bases dos eixos x e y [KLM07].

2.2 PORTAS LÓGICAS QUÂNTICAS QUE ATUAM SOBRE UM ÚNICO QUBIT

Comumente chamadas de operadores quânticos, ou simplesmente operadores, as portas logicas quânticas são unidades lógicas de um computador quântico, sendo responsáveis por alterar o estado de um ou mais qubits.

Uma porta porta logica quântica que atua sobre um único qubit, realiza uma rotação do vetor na representação na esfera de Bloch e é representada por uma matriz de valores complexos 2×2. Seja U uma porta lógica quântica arbitrária, então:

$$U = \begin{bmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{bmatrix}.$$

A aplicação da porta logica U à um qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ é expressada da seguinte forma:

$$U|\psi\rangle = \begin{bmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{bmatrix} \cdot \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} u_{00}\alpha + u_{01}\beta \\ u_{10}\alpha + u_{11}\beta \end{bmatrix}.$$

Dado um operador U qualquer, é definido $U^\dagger$ como seu transposto conjugado complexo.

$$U = \begin{bmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{bmatrix}; \quad U^\dagger = \begin{bmatrix} u_{00}^* & u_{10}^* \\ u_{01}^* & u_{11}^* \end{bmatrix}.$$

Para um operador U qualquer ser unitário, este deve possuir a seguinte propriedade:

$$UU^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

2.2.1 Principais Operadores Quânticos

Operador de Identidade:

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{2.2}$$

Operador de Hadamard:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{2.3}$$

Operadores X, Y e Z de Puali, derivados das matrizes de Pauli:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}; \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{2.4}$$

Operador T:

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/8} \end{bmatrix}. \tag{2.5}$$

Operador mudança de fase:

$$R_\phi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix}. \tag{2.6}$$

Operador genérico U:

$$U = \begin{bmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{bmatrix}. \tag{2.7}$$

2.2.2 Outras representações de um operador

Além da forma matricial, a notação de Dirac permite descrever operadores quânticos por meio de um produto tensorial entre um *ket* e um *bra*.

Para exemplificar, considere os vetores $|\psi\rangle$ e $\langle\phi|$ a seguir:

$$|\psi\rangle = \begin{bmatrix} a \\ b \end{bmatrix}; \quad \langle\phi| = \begin{bmatrix} c & d \end{bmatrix}.$$

Utilizando esta representação, um operador U é escrito da seguinte forma:

$$\begin{aligned} U &= |\psi\rangle\langle\phi| \\ &= \begin{bmatrix} a \\ b \end{bmatrix} \otimes \begin{bmatrix} c & d \end{bmatrix} \\ &= \begin{bmatrix} ac & ad \\ bc & bd \end{bmatrix} \end{aligned} \tag{2.8}$$

2.3 PORTAS LÓGICAS QUÂNTICAS QUE ATUAM SOBRE MÚLTIPLOS QUBITS

Portas lógicas quânticas que atuam em mais de um qubit, podem ser representadas por matrizes de 2^n elementos por 2^n elementos, onde n é o quantidade de qubits em que a porta atua. Por exemplo, seja $|\psi\rangle$ uma superposição de dois qubits e U uma porta quântica que atua sobre uma superposição de dois qubits:

$$|\psi\rangle = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix}.$$

$$U = \begin{bmatrix} u_{0,0} & u_{0,1} & u_{0,2} & u_{0,3} \\ u_{1,0} & u_{1,1} & u_{1,2} & u_{1,3} \\ u_{2,0} & u_{2,1} & u_{2,2} & u_{2,3} \\ u_{3,0} & u_{3,1} & u_{3,2} & u_{3,3} \end{bmatrix}.$$

A aplicação da porta U ao estado $|\psi\rangle$ ocorre da seguinte forma:

$$U|\psi\rangle = \begin{bmatrix} u_{0,0} & u_{0,1} & u_{0,2} & u_{0,3} \\ u_{1,0} & u_{1,1} & u_{1,2} & u_{1,3} \\ u_{2,0} & u_{2,1} & u_{2,2} & u_{2,3} \\ u_{3,0} & u_{3,1} & u_{3,2} & u_{3,3} \end{bmatrix} \cdot \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} u_{0,0}\alpha_0 + u_{0,1}\alpha_1 + u_{0,2}\alpha_2 + u_{0,3}\alpha_3 \\ u_{1,0}\alpha_0 + u_{1,1}\alpha_1 + u_{1,2}\alpha_2 + u_{1,3}\alpha_3 \\ u_{2,0}\alpha_0 + u_{2,1}\alpha_1 + u_{2,2}\alpha_2 + u_{2,3}\alpha_3 \\ u_{3,0}\alpha_0 + u_{3,1}\alpha_1 + u_{3,2}\alpha_2 + u_{3,3}\alpha_3 \end{bmatrix}.$$

Um operador unitário U, arbitrário, pode ser construído utilizando um conjunto de portas quânticas que atuam sobre um único qubit e um operador que atua sobre dois qubits, sem perda de generalidade [KLM07].

2.3.1 Outras formas de escrever um operador

Em alguns casos, um operador pode ser representado por um produto tensorial de operadores, $U = A \otimes B$. Sejam A, B os seguintes operadores:

$$A = \begin{bmatrix} a_{0,0} & a_{0,1} \\ a_{1,0} & a_{1,1} \end{bmatrix}; \quad B = \begin{bmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{bmatrix}.$$

Então:

$$\begin{aligned} U &= A \otimes B \\ &= \begin{bmatrix} a_{0,0} & a_{0,1} \\ a_{1,0} & a_{1,1} \end{bmatrix} \otimes \begin{bmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{bmatrix} \\ &= \begin{bmatrix} a_{0,0} \cdot \begin{bmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{bmatrix} & a_{0,1} \cdot \begin{bmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{bmatrix} \\ a_{1,0} \cdot \begin{bmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{bmatrix} & a_{1,1} \cdot \begin{bmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{bmatrix} \end{bmatrix} \\ &= \begin{bmatrix} a_{0,0} \cdot b_{0,0} & a_{0,0} \cdot b_{0,1} & a_{0,1} \cdot b_{0,0} & a_{0,1} \cdot b_{0,1} \\ a_{0,0} \cdot b_{1,0} & a_{0,0} \cdot b_{1,1} & a_{0,1} \cdot b_{1,0} & a_{0,1} \cdot b_{1,1} \\ a_{1,0} \cdot b_{0,0} & a_{1,0} \cdot b_{0,1} & a_{1,1} \cdot b_{0,0} & a_{1,1} \cdot b_{0,1} \\ a_{1,0} \cdot b_{1,0} & a_{1,0} \cdot b_{1,1} & a_{1,1} \cdot b_{1,0} & a_{1,1} \cdot b_{1,1} \end{bmatrix}. \end{aligned} \tag{2.9}$$

Para ilustrar isso, considere os operadores X e Z de Pauli, respectivamente, e seja o operador U definido a seguir:

$$\begin{aligned} U &= X \otimes Z \\ &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \\ &= \begin{bmatrix} 0 \cdot \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} & 1 \cdot \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \\ 1 \cdot \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} & 0 \cdot \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}. \end{aligned} \tag{2.10}$$

Também é possível escrever um operador na forma:

$$U^{\otimes n} = U \otimes U^{\otimes n-1} \tag{2.11}$$

Para visualizar um exemplo, faça U como o operador de Hadamard H e $n = 2$:

$$\begin{aligned} H^{\otimes 2} &= H \otimes H \\ &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \cdot H & 1 \cdot H \\ 1 \cdot H & -1 \cdot H \end{bmatrix} \\ &= \frac{1}{\sqrt{4}} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}. \end{aligned} \tag{2.12}$$

2.3.2 Operadores de Controle

Operadores de controle são casos particulares de operadores que atuam em mais de um qubit, estes qubits também recebem o nome de qubits de entrada e são classificados entre duas categorias, "qubits de controle" e "qubits alvo". A quantidade de qubits alvo e qubits de controle de um operador é arbitrária, desde que o operador possua pelo menos um qubit alvo e um qubit de controle.

Seja U um operador qualquer que atua sobre um único qubit :

$$U = \begin{bmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{bmatrix}.$$

Seja *ct-U* um operador de controle construído utilizando o operador U, atuando sobre um par de qubits:

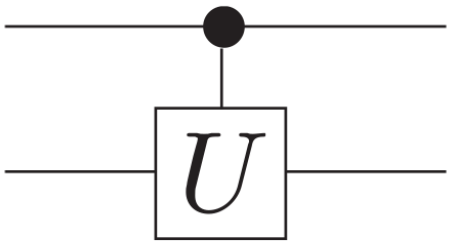


Figura 2.3: Operador *ct-U* [KLM07].

O qubit de controle é responsável por definir se o operador U será ou não aplicado ao qubit alvo. Caso o qubit de controle esteja no estado $|1\rangle$ o operador U será aplicado ao qubit alvo, caso o qubit de controle esteja no estado $|0\rangle$ o operador U não será aplicado.

É possível escrever o operador de controle *ct-U* da seguinte forma:

$$\textit{ct-U} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{bmatrix}.$$

No Teorema A.1.3 do Apêndice A, foi provado que é possível construir um operador de controle utilizando um operador U, para qualquer U que atue sobre um único qubit.

A partir dos resultados apresentados, é possível construir um operador de controle *ct-G*, dado um operador unitário G que atua sobre uma quantidade arbitrária de qubits [KLM07]. Para construir este operador de controle, basta substituir cada porta quântica U, que atua sobre um único qubit, presente no operador G, por uma porta controlada *ct-U* [KLM07].

2.4 AUTOVETORES E AUTOVALORES

Seja U um operador que atua sobre n qubits, e $|\psi\rangle^{\otimes n}$ uma superposição de n qubits. É dito que $|\psi\rangle^{\otimes n}$ é autovetor de U, quando:

$$U|\psi\rangle = e^{i\alpha}|\psi\rangle, \quad \alpha \in \mathbb{R}.$$

Caso $U|\psi\rangle = e^{i\alpha}|\psi\rangle$, onde $\alpha \in \mathbb{R}$, o estado $|\psi\rangle$ é denominado como autovetor de U e o número complexo $e^{i\alpha}$ como autovalor de U.

2.5 MEDIÇÕES

Medições são responsáveis por fornecer informações sobre o estado de um qubit, e consequentemente de um sistema de n qubits. Para ser realizada uma medição é necessário se escolher uma base, seja ela $\{|0\rangle, |1\rangle\}$, $\{|+\rangle, |-\rangle\}$ ou $\{|i+\rangle, |i-\rangle\}$, outras bases podem ser escolhidas, porém devem seguir um conjunto de regras que não será abordado. Seja $|\psi\rangle$ o estado de um único qubit, ao realizar uma medição sobre o estado $|\psi\rangle$, utilizando a base $\{|0\rangle, |1\rangle\}$, o qubit irá colapsar e se tornará $|0\rangle$ ou $|1\rangle$, caso a base escolhida seja $\{|+\rangle, |-\rangle\}$, o qubit irá colapsar e se tornará $|+\rangle$ ou $|-\rangle$.

Realizando-se um medição de um qubit, existe uma determinada probabilidade de que o mesmo colapse em cada estado da base escolhida. Como exemplo, considere a base $\{|A\rangle, |B\rangle\}$ e o estado $|\psi\rangle$, descrito da seguinte forma:

$$|\psi\rangle = \alpha|A\rangle + \beta|B\rangle.$$

Após uma medição no estado $|\psi\rangle$, utilizando a base $\{|A\rangle, |B\rangle\}$, as probabilidades de $|\psi\rangle$ colapsar em $|A\rangle$ e $|B\rangle$ são, respectivamente, p_A e p_B, onde:

$$p_A = |\alpha|^2; \quad p_B = |\beta|^2.$$

Existem outras formas de analisar e realizar uma medição, no entanto a definição apresentada nesta seção servirá para o propósito desejado. Ao referir-se à uma medição em determinado estado ou qubit, irá se referir à uma medição na base $\{|0\rangle, |1\rangle\}$.

2.6 EMARANHAMENTO DE QUBITS

Na computação quântica dois ou mais qubits podem estar em um estado chamado de emaranhado. É dito que um estado $|\phi\rangle$ é um estado emaranhado quando não é possível escrever $|\phi\rangle$ na forma $|\psi_1\rangle \otimes |\psi_2\rangle$, onde estes são qubits independentes, não importando quais estados

$|\psi_1\rangle$ e $|\psi_2\rangle$ sejam escolhidos. Para exemplificar, considere o estado $|\phi\rangle = \frac{|0\rangle|0\rangle+|1\rangle|1\rangle}{\sqrt{2}}$ e observe que não existem $|\psi_1\rangle$ e $|\psi_2\rangle$ capazes de satisfazer seguinte a equação:

$$\frac{|0\rangle|0\rangle + |1\rangle|1\rangle}{\sqrt{2}} = |\psi_1\rangle \otimes |\psi_2\rangle.$$

Ao ser aplicada uma medição em um único qubit de um estado emaranhado, o outro qubit, emaranhado com o primeiro, poderá sofrer alterações em seu estado, independentemente da distância que os separam. Suponha que, após uma medição no primeiro qubit do estado $|\phi\rangle$, seja obtido o estado $|1\rangle$. Automaticamente o estado do segundo qubit de $|\phi\rangle$ irá colapsar no estado $|1\rangle$. Este é um fenômeno importante da mecânica quântica, que abre possibilidades que não poderiam ser exploradas utilizando computação clássica.

2.7 FASE GLOBAL E RELATIVA

Seja $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. Como $\alpha, \beta \in \mathbb{C}$ e $|\alpha|^2 + |\beta|^2 = 1$, é possível escrever α e β na seguinte forma, onde θ, ϕ_1 e ϕ_2 são números reais, normalmente interpretados como ângulos:

$$\begin{aligned} \alpha &= e^{i\phi_1} \cos\left(\frac{\theta}{2}\right) \\ \beta &= e^{i\phi_2} \sin\left(\frac{\theta}{2}\right). \end{aligned} \tag{2.13}$$

Seja φ, tal que = $\phi_2 = \phi_1 + \varphi$, então:

$$\begin{aligned} \alpha &= e^{i\phi_1} \cos\left(\frac{\theta}{2}\right) \\ \beta &= e^{i(\phi_1+\varphi)} \sin\left(\frac{\theta}{2}\right). \end{aligned} \tag{2.14}$$

Como $|e^{i\phi_1}|^2 = 1$, multiplicar uma um dos coeficientes de $|\psi\rangle$ por $e^{i\phi_1}$ não afetará o resultado de uma medição do qubit $|\psi\rangle$, portanto substituindo os valores α e β da seguinte forma, o resultado de uma medição em $|\psi\rangle$ não será alterado:

$$\begin{aligned} \alpha &= \cos\left(\frac{\theta}{2}\right) \\ \beta &= e^{i\varphi} \sin\left(\frac{\theta}{2}\right). \end{aligned} \tag{2.15}$$

Denomina-se o valor $e^{i\phi_1}$ como a fase global do estado $|\psi\rangle$, enquanto $e^{i\varphi}$ é denominado de fase relativa.

3 CRIPTOGRAFIA

Neste capítulo serão introduzidos os conceitos de criptografia simétrica e assimétrica, além de uma breve apresentação do protocolo criptográfico AES [RD01], baseado no modelo simétrico, e dos protocolos RSA [RSA78] e o acordo de chaves de Diffie-Hellman-Merkle [Mer78], baseados em criptografia assimétrica.

Os protocolos RSA e o AES continuam sendo amplamente utilizados, enquanto o acordo de chaves Diffie-Hellman-Merkle foi descontinuado, porém tem inspirado a construção de outros protocolos criptográficos.

O conteúdo deste capítulo também pode ser encontrado no livro "Criptografia e Segurança de Redes: Princípios e Praticas" [SVBF14], o qual compreende, também, outros aspectos importantes para a segurança da informação.

3.1 MODELOS CRIPTOGRÁFICOS

3.1.1 Criptografia Simétrica

A criptografia simétrica baseia-se no principio de utilizar uma mesma chave k para criptografar e descriptografar uma mensagem m. Na Tabela 3.1 são estabelecidas as convenções utilizadas no restante desta subseção.

Tabela 3.1: Símbolos usados para descrever o modelo de criptografia simétrica.

k	Chave secreta
f	Função de cifragem
f^{-1}	Função de decifragem
m	Mensagem
m'	Mensagem m após ser criptografada utilizando a função f

Para realizar a cifragem num modelo criptográfico baseado em criptografia simétrica, é utilizada uma função f, em conjunto com a mensagem m e a chave k, retornando a mensagem cifrada.

$$f(m, k) = m'.$$

Para se decifrar uma mensagem num modelo criptográfico baseado em criptografia simétrica, é necessário conhecer a chave secreta k, a inversa da função f e a mensagem criptografada m'.

$$f^{-1}(m', k) = m.$$

3.1.2 Criptografia Assimétrica

Protocolos criptográficos baseados em criptografia assimétrica utilizam-se do conceito de funções de mão única, que são funções fáceis de serem computadas dada uma entrada, porém computacionalmente difíceis de inverter dada uma imagem qualquer.

É importante mencionar que, até então, não foi provada a existência de funções de mão única para computadores clássicos. No entanto existem funções para as quais não se conhece algoritmo clássico probabilístico eficiente, que seja capaz de calcular as inversas destas funções.

Na criptografia assimétrica, cada usuário gera um par de chaves para si, sendo uma chave privada, conhecida apenas pelo mesmo, e uma pública, disponível para todos os usuários. Documentos criptografados utilizando a chave pública somente podem ser descriptografados utilizando a chave privada e vice-versa.

Neste sistema criptográfico, a chave pública é gerada a partir de uma chave privada utilizando uma função de mão única. Com isso nota-se que é possível deduzir a chave privada a partir uma chave pública, porém, como a função utilizada para gerar a chave pública é uma função de mão única, ela é computacionalmente difícil de ser invertida. Protocolos criptográficos baseados em criptografia assimétrica podem ser utilizado para garantir autenticidade e inviolabilidade da informação.

Na Tabela 3.2 são estabelecidas as convenções utilizadas no restante desta subseção.

Tabela 3.2: Símbolos usados para descrever o modelo de criptografia assimétrica.

A e B	Usuários do sistema
pri_A e pub_A	Chaves privada e pública do usuário A, respectivamente
pri_B e pub_B	Chaves privada e pública do usuário B, respectivamente
f	Função de cifragem
d	Função de decifragem
g	Função autenticadora
m	Mensagem
m'	Mensagem m após ser criptografada utilizando a função f
s	Assinatura da mensagem m, utilizando a função g

Para que um usuário qualquer envie uma mensagem m para o usuário A, basta que ele utilize a função f e a chave pública pub_A para gerar a mensagem criptografada m', e a envie para A.

$$f(m, pub_A) = m'.$$

Como $f^{-1}(m', pub_A)$ é computacionalmente difícil de se calcular, então a mensagem pode ser enviada por um canal não seguro.

Para que o usuário A decifre uma mensagem m' criptografada utilizando a chave pub_A, basta que ele utilize a função d e sua chave privada pri_A para obter a mensagem decifrada.

$$d(m', pri_A) = m.$$

Para que o usuário A publique uma mensagem autenticada para todos os usuários, basta utilizar a função g, a chave privada pri_A e a mensagem m, para gerar uma assinatura s.

$$g(m, pri_A) = s.$$

Somente utilizando a chave pública pub_A é possível verificar que a assinatura s pertence ao usuário A e, desta forma, verificando que a mensagem m foi enviada pelo usuário A.

Para que um usuário B envie uma mensagem autenticada e secreta para o usuário A, basta que o usuário B autentique a mensagem utilizando sua chave privada pri_B, em seguida criptografe a mensagem autenticada utilizando a chave publica de A, pub_A, e por fim envia a mensagem para A.

3.2 RIVEST-SHAMIR-ADLEMAN - RSA

O protocolo RSA de criptografia, foi descrito inicialmente em 1978 por Ron Rivest, Adi Shamir e Leonard Adleman. Contudo Cliffor Cocks, um matemático Inglês que trabalhava para a agência de inteligência britânica Government Communications Headquarters (GCHQ), já havia desenvolvido um sistema equivalente em 1973, mas só foi revelado em 1997 por ser considerado segredo de estado.

Este algoritmo criptográfico é baseado no modelo de criptografia assimétrica. O protocolo RSA utiliza a multiplicação de dois primos distintos como função de mão única, uma vez que não existe algoritmo clássico conhecido capaz de fatorar um número muito grande de forma viável.

Neste protocolo, ambas as chaves, publica e privada, são compostas por dois números inteiros. No restante desta seção $\{N, E\}$ será a chave pública e $\{N, D\}$ a chave privada.

3.2.1 Gerando as chaves pública e privada

Dado um número primo p e um número inteiro positivo a, a função φ, denominada função totiente de Euler, atua da seguinte forma:

$$\varphi(p^a) = p^{a-1}(p-1).$$

Seja $n = p_1^{a_1} p_2^{a_2} \cdots p_l^{a_l}$, onde $l \geq 1$, tal que cada p_j é um número primo distinto e $a_j \in \mathbb{Z}_{>0}, \forall j \in \{1, 2, \cdots, l\}$. A função λ denominada função totiente de Carmichael, é definida da seguinte forma:

$$\lambda(n) = \begin{cases} \varphi(n) & \text{se } n = p^a, \text{ onde } p = 2 \text{ e } a < 3 \text{ ou } p > 3; \\ \frac{1}{2}\varphi(n) & \text{se } n = 2^a, \text{ onde } a \geq 3; \\ mmc(\lambda(p_1^{a_1}), \lambda(p_2^{a_2}), \cdots, \lambda(p_l^{a_l})) & \text{se } n = \prod_{i=0}^{l} p_i^{a_i}. \end{cases}$$

O Algoritmo 1 pode ser utilizado para gerar as chaves do protocolo RSA.

Algoritmo 1 *Geração de Chaves RSA*():

1: Sortear dois números primos P e Q, de preferência com magnitudes similares.
2: $N \leftarrow P * Q$.
3: Calcular a função totiente $\lambda(N) = mmc(P-1, Q-1)$.
4: Sortear um valor E, tal que $1 < E < \lambda(N)$ e $mdc(E, \lambda(N)) = 1$.
5: Calcular um valor D, tal que $E \cdot D \equiv 1 \pmod{\lambda(N)}$, podendo ser calculado utilizando o Algoritmo Euclidiano Estendido.
6: **return** $\{N, E\}$ e $\{N, D\}$, como chaves pública e privada, respectivamente.

3.2.2 Cifrando, decifrando e autenticando uma mensagem

Seja M uma mensagem. Para gerar a mensagem cifrada M', faz-se o seguinte:

$$M' \leftarrow M^E \pmod{N}.$$

Para decifrar a mensagem M', realiza-se a seguinte operação:

$$M \leftarrow M'^D \pmod{N}.$$

Para autenticar uma mensagem, inicialmente calcula-se o valor *hash* da mensagem. Seja H o valor *hash* da mensagem M, calculado utilizando uma função *hash* arbitrária. Em seguida é gerada uma assinatura S para a mensagem M:

$$S \leftarrow H^D \pmod{N}.$$

Para verificar uma assinatura, novamente calcula-se o valor H. Em seguida a assinatura da mensagem é utilizada para obter um valor H', que deverá ser idêntico ao valor H:

$$H' \leftarrow S^E \pmod{N}.$$

Se $H = H'$, então a assinatura é verdadeira, caso contrário a assinatura é falsa.

3.2.3 Exemplo de cifragem com números pequenos

A seguir é apresentada a Tabela 3.3, na qual é realizado um exemplo do processo de geração de chaves utilizando o Algoritmo 1, a Tabela 3.4 com as chaves obtidas e a Tabela 3.5 exemplificando o processo de cifragem.

Tabela 3.3: Exemplo de geração de chaves através do algoritmo RSA

Escolher dois primos P=61 e Q=43
Calcular o módulo: $N = P \cdot Q = 61 \cdot 43 = 3233$
Calcular o totiente: $\lambda(3233) = mmc(60, 52) = 780$
Escolher expoente E, tal que $1 < E < 780$ e $mdc(E, 780)$. $E = 17$
Calcular expoente D, tal que $D \cdot E \equiv 1 \pmod{780}$. $D = 413$

Tabela 3.4: Exemplo de chaves geradas através do algoritmo RSA

Chave pública: $\{N = 3233, E = 17\}$
Chave privada: $\{N = 3233, D = 413\}$

Tabela 3.5: Exemplo de encriptação através do algoritmo RSA

Mensagem: "A"	M = 65
Cifragem ($M \rightarrow M'$)	$M' \equiv M^E \pmod{N}$, $2790 \equiv 65^{17} \pmod{3233}$
Mensagem cifrada:	$M' = 2790$
Decifragem ($M' \rightarrow M$)	$M \equiv M'^D \pmod{N}$, $65 \equiv 2790^{423} \pmod{3233}$
Mensagem decifrada:	$M = 65$

3.2.4 Dificuldade de decifrar uma mensagem cifrada utilizando o protocolo RSA

Note que, para qualquer valor D, satisfazendo $D \cdot E \equiv 1 \pmod{\lambda(N)}$, a chave $\{N, D\}$ pode ser utilizada como chave privada do par de chaves em questão. Note também que D pode ser obtido a partir da chave pública $\{N, E\}$. No entanto para calcular o resultado da função $\lambda(N)$, é necessário encontrar os valores de P e Q. Portanto a dificuldade de descriptografar, sem a chave privada, uma mensagem criptografada utilizando o protocolo RSA, se resume na dificuldade de fatorar N.

Atualmente o custo computacional do melhor algoritmo clássico probabilístico, esta limitado por um período de tempo super-polinomial, regido por $e^{O((\log N)^{\frac{1}{3}})(\log\log N)^{\frac{2}{3}})}$, onde N é o número a se fatorar [LLMP93].

3.3 TROCA DE CHAVES DIFFIE-HELLMAN-MERKLE

O algoritmo de troca de chaves Diffie-Hellman-Merkle, foi desenvolvido por Whitfield Diffie e Martin Hellman e publicado em 1976. Seu conceito foi inicialmente proposto por Malcolm Williamson, o qual era funcionário do Government Communications Headquarters (GCHQ) no Reino Unido, porém foi mantido em segredo por ser considerado uma questão de segurança nacional, sendo liberado à público somente em 1997. Martin Hellman sugeriu, em 2002, que o algoritmo fosse chamado de troca de chave de Diffie-Hellman-Merkle, até então conhecido por troca de chaves de Diffie-Hellman, em reconhecimento às contribuições de Ralph Merkle para a invenção da criptografia de chave pública.

Este protocolo criptográfico baseia-se no modelo de criptografia assimétrica, onde a função de mão única é a potenciação em aritmética modular e a dificuldade de decifrar o algoritmo está relacionada com o problema do logaritmo discreto, o qual não possui, em computação clássica, algoritmo conhecido capaz de resolvê-lo de forma viável para números de grandes magnitudes. Este protocolo tem como objetivo principal estabelecer uma chave compartilhada com outro indivíduo, mesmo que o canal utilizado seja inseguro. Tal chave poderá ser utilizada em algum algoritmo mais robusto, baseado em criptografia simétrica.

Neste protocolo a chave pública é composta por um número primo P, um gerador G do subgrupo $\mathbb{Z}_p^\times$ e um valor $A \in \mathbb{Z}_p^\times$. Enquanto a chave privada é composta pelos mesmo valores P e G em conjunto com um inteiro α, tal que $A \equiv G^\alpha \pmod{P}$.

3.3.1 Geração de Chaves Diffie-Hellman-Merkle

O Algoritmo 2 é utilizado para gerar um par de chaves de Diffie-Hellman-Merkle.

Algoritmo 2 *Geração de Chaves Diffie-Hellman-Merkle*():

1: Sortear um número primo P, de preferencia da ordem de 2^{1024}.
2: Encontrar um gerador G do subgrupo $\mathbb{Z}_p^\times$.
3: Escolher aleatoriamente um inteiro α.
4: $A \leftarrow G^\alpha \pmod{p}$
5: **return** $\{P, G, A\}$ e $\{P, G, \alpha\}$, como chaves pública e privada, respectivamente.

3.3.2 Gerando uma chave compartilhada

O principal objetivo do acordo de chaves Diffie-Hellman-Merkle é gerar valor K, através de um canal não seguro, o qual será conhecido somente pelos usuários, digamos, $Alice$ e Bob, que desejam se comunicar.

Para isso, o usuário $Alice$ gera um par chaves, $\{P, G, A\}$ e $\{P, G, \alpha\}$, e disponibiliza sua chave pública, $\{P, G, A\}$ para para Bob.

Em seguida Bob utiliza os valores P e G e gera um novo par de chaves, $\{P, G, B\}$ e $\{P, G, \beta\}$, e disponibiliza sua chave pública, $\{P, G, B\}$ para para $Alice$.

Finalmente $Alice$ e Bob geram o valor k da seguinte forma:

Como $Alice$ conhece os valores α, B, P:

$$\begin{aligned} k &= B^{\alpha} \pmod{P} \\ &= (G^{\beta})^{\alpha} \pmod{P} \\ &= G^{\beta\alpha} \pmod{P}. \end{aligned} \tag{3.1}$$

Como Bob conhece os valores β, A, P:

$$\begin{aligned} k &= A^{\beta} \pmod{P} \\ &= (G^{\alpha})^{\beta} \pmod{P} \\ &= G^{\alpha\beta} \pmod{P}. \end{aligned} \tag{3.2}$$

Com isso, tanto $Alice$ como Bob possuem uma mesma chave secreta k.

3.3.3 Dificuldade de deduzir a chave secreta a partir das chaves públicas

Note que, para encontrar valor k basta encontrar α ou β. Ou seja, dados os valores P, G, A, B, se deseja encontrar α ou β, de modo a satisfazer ao menos uma das equações a seguir.

$$\begin{aligned} A &\equiv G^{\alpha} \pmod{P}; \\ B &\equiv G^{\beta} \pmod{P}. \end{aligned} \tag{3.3}$$

Atualmente o custo computacional do melhor algoritmo clássico probabilístico, esta limitado por um período de tempo super-polinomial, regido por $e^{O((\log P)^{\frac{1}{3}})(\log\log P)^{\frac{2}{3}})}$ [LLMP93].

3.4 ADVANCED ENCRYPTION STANDARD (AES)

O protocolo criptográfico AES, desenvolvido por Vincent Rijmen e Joan Daemen e baseado em criptografia simétrica, foi adotado pelo governo dos Estados Unidos da América em 1998, após um concurso do NIST (National Institute of Standarts and Technology), o qual tinha como objetivo definir um novo protocolo criptográfico simétrico, que iria substituir o DES (Data Encryption Standard) e o 3DES, os quais já apresentavam várias vulnerabilidades.

Tal protocolo utiliza uma função f, conhecida e rápida de se calcular, em conjunto com uma chave secreta k para cifrar uma mensagem m. Para decifrar uma mensagem, o algoritmo utiliza a função f^{-1}, a qual pode ser calculada, em tempo viável, realizando a inversão da função f. A chave secreta k pode ser composta de 128, 192 ou 256 bits, ou seja 2^{128}, 2^{192} ou 2^{256} possíveis chaves, respectivamente.

O algoritmo AES utiliza do principio de cifragem em blocos. Onde a chave e a mensagem são codificadas em blocos de bytes, sendo o tamanho de cada bloco fixado de acordo com a quantidade de bits que compõe a chave. Chaves de 128, 192 e 256 bits, serão codificadas em blocos de dimensões 4×4, 4×6 e 4×8, respectivamente, onde cada elemento do bloco corresponde a um byte.

Seja m_i uma parte da mensagem e k uma chave, ambas compostas de 128 bits. Seja $\{a_0, a_2, \cdots a_{15}\}$ a codificação da mensagem m em bytes, e $\{b_0, b_2, \cdots b_{15}\}$ a codificação da chave k em bytes. Então suas representações na forma de blocos serão:

$$m_i = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 & a_7 \\ a_8 & a_9 & a_{10} & a_{11} \\ a_{12} & a_{13} & a_{14} & a_{15} \end{bmatrix};$$

$$k = \begin{bmatrix} b_0 & b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 & b_7 \\ b_8 & b_9 & b_{10} & b_{11} \\ b_{12} & b_{13} & b_{14} & b_{15} \end{bmatrix}.$$

3.4.1 Cifragem

Para cifrar um bloco m_i da mensagem original, utilizando uma chave secreta k, pode se utilizar o Algoritmo 3. O valor de n corresponde à 10, 12 ou 14, para chaves de tamanho 128, 192 e 256, respectivamente.

Algoritmo 3 *Cifragem AES*(m_i, k, n):

1: $estado \leftarrow m_i$
2: $chave \leftarrow$ *KeyExpansion(k)*
3: $estado \leftarrow$ *AddRoundKey(estado, chave[0,3]).*
4: **for** $i \in \{1, \cdots, n-1\}$ **do**
5: $estado \leftarrow$ *SubBytes(estado).*
6: $estado \leftarrow$ *ShiftRows(estado).*
7: $estado \leftarrow$ *MixColumns(estado).*
8: $estado \leftarrow$ *AddRoundKey(estado, chave[4*i,4*i+3]).*
9: **end for**
10: $estado \leftarrow$ *SubBytes(estado).*
11: $estado \leftarrow$ *ShiftRows(estado).*
12: $c_i \leftarrow$ *AddRoundKey(estado, chave[4*n,4*n+3]).*
13: **return** c_i.

Para decifrar um bloco cifrado c_i, utilizando uma chave secreta k, podemos utilizar o Algoritmo 4. O valor de n corresponde à 10, 12 ou 14, para chaves de tamanho 128, 192 e 256, respectivamente.

Algoritmo 4 *Decifragem AES*(c_i, k):

1: $estado \leftarrow c_i$.
2: $chave \leftarrow$ *KeyExpansion(k).*
3: $estado \leftarrow$ *AddRoundKey(estado, chave[4*n,4*n+3]).*
4: **for** $i \in \{n-1, \cdots, 1\}$ **do**
5: $estado \leftarrow$ *InvShiftRows(estado).*
6: $estado \leftarrow$ *InvSubBytes(estado).*
7: $estado \leftarrow$ *AddRoundKey(estado, chave[4*i,4*i+3]).*
8: $estado \leftarrow$ *InvMixColumns(estado).*
9: **end for**
10: $estado \leftarrow$ *InvShiftRows(estado).*
11: $estado \leftarrow$ *InvSubBytes(estado).*
12: $m_i \leftarrow$ *AddRoundKey(estado, chave[0,3]).*
13: **return** m_i.

Neste trabalho não serão descritas as funções *KeyExpansion* e *AddRoundKey*, assim como as funções *SubBytes*, *ShiftRows*, *MixColumns* e suas inversas *InvSubBytes*, *InvShiftRows*, *InvMixColumns*, pois não serão relevantes para as análises propostas. Porém podem ser encontradas na publicação original referente ao algoritmo AES [RD01].

3.4.2 Dificuldade de decifrar um bloco cifrado, sem o conhecimento da chave secreta

Para decifrar um bloco c_i, que foi cifrado utilizando do protocolo AES, devemos encontrar uma chave específica k, entre 2^{128}, 2^{192} ou 2^{256} possíveis chaves, para chaves de 128, 192 e 256 bits, respectivamente.

Note que o Algoritmo 4 pode ser computado em tempo viável, de outro modo o protocolo AES como um todo seria inviável. Para encontrar uma determinada chave k de n bits, realizando um ataque de força bruta, pode ser necessário executar este algoritmo 2^n, inviabilizando esta forma ataque.

Tabela 3.6: Custo computacional do melhor algoritmo clássico para decifrar o protocolo AES

AES-128	Tempo $2^{126.13}$	Espaço 2^{56}
AES-196	Tempo $2^{189.91}$	Espaço 2^{48}
AES-256	Tempo $2^{254.27}$	Espaço 2^{40}

A Tabela 3.6 descreve o custo computacional de tempo e espaço, simultaneamente, necessários para decifrar o protocolo AES. Estes custos computacionais são referentes ao atual melhor algoritmo clássico que realiza esta tarefa, onde são utilizada chaves de 128, 192 e 256 bits, respectivamente [TW15].

4 TRANSFORMAÇÃO QUÂNTICA DE FOURIER

Neste capítulo será introduzida a transformação quântica de Fourier, baseada na transformada de Fourier, a qual é amplamente utilizada na matemática e na computação. Em poucas palavras, a transformada de Fourier é uma transformação matemática que decompõe funções baseadas em uma determinada entrada em funções baseadas na frequência desta entrada.

Uma vez que a transformada de Fourier atua sobre um conjunto de dados contínuos e na prática, em computação, lida-se com dados discretos, é desejada uma versão da transformada de Fourier que seja capaz de atuar sobre um conjunto de dados discretos. A transformada discreta de Fourier, a qual será apresentada neste capítulo, cumpre este papel, tornando-se uma ferramenta de extrema importância para a computação.

Na computação quântica, é possível construir um circuito quântico que atua de forma similar, mas não idêntica, à transformada discreta de Fourier. Enquanto a transformada discreta de Fourier retorna um conjunto de valores, este circuito apenas transforma os estados de um conjunto de qubits, onde uma medição sobre estes estados retornará uma estimativa para os valores destes, não retornando os valores em si. No entanto este circuito quântico possui um custo computacional exponencialmente menor se comparado ao custo computacional da transformada discreta de Fourier, implementada em computadores clássicos, apresentando esta vantagem em relação ao algoritmo clássico. Sendo uma peça chave em muitos algoritmos quânticos, sua análise é de extrema importância.

As análises presentes neste capítulo foram adaptadas do livro "Quantum Computation and Quantum Information: 10th Anniversary Edition" [NC11], adicionando novas informações para facilitar o entendimento de determinadas passagens.

4.1 TRANSFORMADA DISCRETA DE FOURIER

Nesta seção será abordada a transformada discreta de Fourier.

Definição 4.1.1. *A transformada discreta de Fourier* ($\mathcal{DFT}$) *transforma uma sequência de N números complexos* $\{x_j\} := x_0, x_1, \cdots, x_{N-1}$ *em uma nova sequência de números complexos* $\{X_k\} := X_0, X_1, \cdots, X_{N-1}$*, segundo a seguinte regra:*

$$\{x_j\} \xrightarrow{\mathcal{DFT}} \{X_k\} : \quad X_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{i2\pi kj/N}; \tag{4.1}$$

Definição 4.1.2. *A transformada discreta inversa de Fourier* ($\mathcal{IDFT}$) *transforma uma sequência de N números complexos* $\{X_k\} = X_0, X_1, \cdots, X_{N-1}$ *em uma nova sequência de números complexos* $\{x_j\} = x_0, x_1, \cdots, x_{N-1}$*, segundo a seguinte regra:*

$$\{X_j\} \xrightarrow{\mathcal{IDFT}} \{x_j\} : \quad x_j = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X_k e^{-i2\pi kj/N}. \tag{4.2}$$

Para a definição da transformada discreta de Fourier e sua inversa, o sinal do expoente $e^{i2\pi kj/N}$ e o fator de normalização $\frac{1}{\sqrt{N}}$, em frente aos somatórios nas Definições 4.1.1 e 4.1.2 podem variar dependendo da definição empregada, porém os sinais dos expoentes da $\mathcal{DFT}$ e da $\mathcal{IDFT}$ devem ser opostos e a multiplicação dos fatores de normalização deve ser igual a $\frac{1}{N}$.

Por convenção serão utilizados os valores apresentados nos ambientes de definição 4.1.1 e 4.1.2.

A seguir será reescrita a transformada discreta de Fourier, na forma de uma matriz que atua sobre um vetor de N números complexos $\{x_j\} := x_0, x_1, \cdots, x_{N-1}$.

Seja $\zeta = e^{2\pi i/N}$ e considere que as sequências $\{x_i\}$ e $\{X_k\}$ das Definições 4.1.1 e 4.1.2 são dados por vetores coluna. A $\mathcal{DFT}$ pode ser expressa como uma matriz M tal que, multiplicando $\{x_i\}$ por M, se obtém $\{X_k\}$. Segue a definição de M:

$$M = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \zeta & \zeta^2 & \zeta^3 & \cdots & \zeta^{(N-1)} \\ 1 & \zeta^2 & \zeta^4 & \zeta^6 & \cdots & \zeta^{2(N-1)} \\ 1 & \zeta^3 & \zeta^6 & \zeta^9 & \cdots & \zeta^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta^{(N-1)} & \zeta^{2(N-1)} & \zeta^{3(N-1)} & \cdots & \zeta^{(N-1)^2} \end{bmatrix}.$$

Com isso é possível observar que o custo computacional de multiplicar a matriz M por um vetor de N elementos é $\Theta(N^2)$, e portanto o custo computacional a implementação padrão da transformada discreta de Fourier é $\Theta(N^2)$. Contudo existe uma forma de realizá-la com custo computacional de $\Theta(N \log(N))$ denominada de transformada discreta rápida de Fourier $\mathcal{FFT}$.

Ao final deste capítulo estes resultados serão relevantes para que se possa comparar o custo computacional de realizar a transformada discreta de Fourier, utilizando computação clássica, com o custo computacional de realizar a transformação quântica de Fourier, utilizada em computadores quânticos.

4.2 TRANSFORMAÇÃO QUÂNTICA DE FOURIER

Analogamente à transformada discreta de Fourier utilizada nos computadores clássicos, a transformação quântica de Fourier ($\mathcal{QFT}$), proposta por Don Coppersmith em 1994 [Cop94], atua sobre os coeficientes de um estado quântico, realizando uma transformação unitária. Ou seja, aplicando a $\mathcal{QFT}$ sobre um estado $|\psi_0\rangle = \sum_j x_j |j\rangle$, obtém-se um estado $|\psi_1\rangle = \sum_j X_j |j\rangle$. O foco central desta seção é descrever como implementar um circuito quântico que realize esta tarefa. Seja:

$$N = 2^n, \quad n \in \mathbb{Z}_{>0}.$$

Sejam $|x\rangle^n$ e $|X\rangle^n$ estados quânticos arbitrários, ambos de n qubits, definidos a seguir:

$$|x\rangle^n = \sum_{j=0}^{N-1} x_j |j\rangle, \quad \forall j \in \{0, 1 \cdots, N-1\};$$

$$|X\rangle^n = \sum_{k=0}^{N-1} X_k |k\rangle, \quad \forall k \in \{0, 1 \cdots, N-1\}.$$

Definição 4.2.1. *A transformação quântica de Fourier ($\mathcal{QFT}$) é uma transformação unitária que transforma um estado quântico de n qubits $|x\rangle^n$, em um outro estado quântico de n qubits $|X\rangle^n$, segundo a seguinte regra:*

$$|x\rangle^n \xrightarrow{\mathcal{QFT}} |X\rangle^n : \quad X_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{i2\pi kj/N}. \tag{4.3}$$

4.2.1 Implementação da transformação quântica de Fourier

Nesta subseção será apresentada a construção de um circuito quântico capaz de realizar a transformação quântica de Fourier, utilizando apenas $\frac{n(n-1)+2}{2}$ portas quânticas. Este processo será dividido em duas etapas. A primeira etapa será reescrever a operação em uma forma equivalente, de forma a ser mais facilmente implementada por um circuito. A segunda etapa será construir o circuito capaz de realizar a transformação.

4.2.1.1 Reescrevendo a QFT

Para a implementação da transformação quântica de Fourier é necessário implementar um circuito capaz de mapear qualquer estado $|x\rangle^n$ arbitrário para um estado $|X\rangle^n$ correspondente. Construindo um circuito capaz de realizar o mapeamento 4.4, já será suficiente, uma vez que neste mapeamento estabelece como a transformação atua para cada estado $|j\rangle$ da base computacional.

$$|j\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{i2\pi kj/N} |k\rangle, \quad \forall j, k \in \{0, 1 \cdots, N-1\}. \tag{4.4}$$

Portanto o objetivo será obter um circuito quântico capaz de realizar esta transformação. Durante o restante deste capítulo serão adotadas as seguintes notações, que serão úteis para lidar com j tanto nas situações em que este deve ser tratado como um número inteiro, assim como nas situações que se deseja lidar com j como uma palavra de n bits:

$$j \equiv [j_1 j_2 j_3 \cdots j_n] = \sum_{l=1}^{n} j_l 2^{n-l}, \quad \forall j_l \in \{0, 1\} \quad \forall l \in \mathbb{Z}_{>0}. \tag{4.5}$$

$$[0.j_1 j_2 j_3 \cdots j_n] = \sum_{l=1}^{n} j_l 2^{-l}, \quad \forall j_l \in \{0, 1\} \quad \forall l \in \mathbb{Z}_{>0}. \tag{4.6}$$

Com isso, tem-se que:

$$\begin{aligned} |j\rangle &= |[j_1 j_2 j_3 \cdots j_n]\rangle, \quad \forall j \in \{0, 1 \cdots, N-1\}; \\ |k\rangle &= |[k_1 k_2 k_3 \cdots k_n]\rangle, \quad \forall k \in \{0, 1 \cdots, N-1\}. \end{aligned} \tag{4.7}$$

Portanto a transformação quântica de Fourier deverá realizar a seguinte transformação:

$$|[j_1 j_2 j_3 \cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \sum_{[k_1 k_2 k_3 \cdots k_n]=0}^{2^n-1} e^{2\pi i [k_1 k_2 k_3 \cdots k_n][j_1 j_2 j_3 \cdots j_n]/2^n} |[k_1 k_2 k_3 \cdots k_n]\rangle. \tag{4.8}$$

Reescrevendo o somatório em n somatórios menores com apenas duas parcelas cada, um somatório para cada bit do número $k = [k_1k_2k_3\cdots k_n]$ de entrada, a seguinte equação é obtida:

$$|[j_1j_2j_3\cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} e^{2\pi i[k_1k_2k_3\cdots k_n][j_1j_2j_3\cdots j_n]/2^n} |[k_1k_2k_3\cdots k_n]\rangle. \quad (4.9)$$

Ao reescrever o n-qubit $e^{2\pi i[k_1k_2k_3\cdots k_n][j_1j_2j_3\cdots j_n]/2^n}|[k_1k_2k_3\cdots k_n]\rangle$ na forma de um produto tensorial de n termos, o seguinte resultado é obtido:

$$\begin{aligned} |[j_1j_2j_3\cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} \Bigg(& e^{2\pi i[j_1j_2j_3\cdots j_n]k_1 2^{n-1}/2^n} |k_1\rangle \otimes \\ & e^{2\pi i[j_1j_2j_3\cdots j_n]k_2 2^{n-2}/2^n} |k_2\rangle \otimes \cdots . \\ & e^{2\pi i[j_1j_2j_3\cdots j_n]k_{n-1} 2^{1}/2^n} |k_{n-1}\rangle \otimes \\ & e^{2\pi i[j_1j_2j_3\cdots j_n]k_n 2^{0}/2^n} |k_n\rangle \Bigg) \end{aligned} \quad (4.10)$$

Reescrevendo o conjunto de produtos tensoriais:

$$|[j_1j_2j_3\cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} \left(\bigotimes_{l=1}^{n} e^{2\pi i[j_1j_2j_3\cdots j_n]k_l 2^{n-l}/2^n} |k_l\rangle \right). \quad (4.11)$$

Como $\frac{2^{n-l}}{2^n} = 2^{-l}$ a equação pode ser simplificada:

$$|[j_1j_2j_3\cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} \left(\bigotimes_{l=1}^{n} e^{2\pi i[j_1j_2j_3\cdots j_n]k_l 2^{-l}} |k_l\rangle \right). \quad (4.12)$$

Note que é possível reescrever a transformação na seguinte forma:

$$|[j_1j_2j_3\cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left(\sum_{k_l=0}^{1} e^{2\pi i[j_1j_2j_3\cdots j_n]k_l 2^{-l}} |k_l\rangle \right). \quad (4.13)$$

Escrevendo o somatório de (4.13) de maneira explícita:

$$|[j_1j_2j_3\cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left(|0\rangle + e^{2\pi i[j_1j_2j_3\cdots j_n]2^{-l}} |1\rangle \right). \quad (4.14)$$

Conforme (4.5), substituindo a palavra binária $[j_1j_2j_3\cdots j_n]$, a qual representa um número inteiro, por um somatório:

$$|[j_1j_2j_3\cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left(|0\rangle + e^{2\pi i \sum_{h=1}^{n}(j_h 2^{n-h})2^{-l}} |1\rangle \right). \quad (4.15)$$

Como 2^n, dentro do somatório $\sum_{h=1}^{n}(j_h 2^{n-h})$, não depende de h, é possível mover 2^n para fora do somatório:

$$|[j_1 j_2 j_3 \cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left(|0\rangle + e^{2\pi i \sum_{h=1}^{n}(j_h 2^{-h}) 2^{n-l}} |1\rangle \right). \tag{4.16}$$

A partir de (4.6), $\sum_{h=1}^{n}(j_h 2^{-h})$ equivale a $[0.j_1 j_2 j_3 \cdots j_n]$, portanto:

$$|[j_1 j_2 j_3 \cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left(|0\rangle + e^{2\pi i [0.j_1 j_2 j_3 \cdots j_n] 2^{n-l}} |1\rangle \right). \tag{4.17}$$

Fazendo $n - l = m$:

$$|[j_1 j_2 j_3 \cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \bigotimes_{m=n-1}^{0} \left(|0\rangle + e^{2\pi i [0.j_1 j_2 j_3 \cdots j_n] 2^{m}} |1\rangle \right). \tag{4.18}$$

Com isso, realizando a multiplicação $[0.j_1 j_2 j_3 \cdots j_n]2^m$, obtém-se:

$$|[j_1 j_2 j_3 \cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \bigotimes_{m=n-1}^{0} \left(|0\rangle + e^{2\pi i [j_1 \cdots j_m . j_{m+1} \cdots j_n]} |1\rangle \right). \tag{4.19}$$

Como $e^{2\pi i} = 1$, então $e^{2\pi i k} = 1, \forall k \in \mathbb{Z}_{>0}$, portanto ignorar a parte inteira, $[j_1 \cdots j_m]$, do valor $[j_1 \cdots j_m . j_{m+1} \cdots j_n]$, não alterará o resultado da transformação. Desta forma é possível observar que a seguinte equação é verdadeira:

$$|[j_1 j_2 j_3 \cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \bigotimes_{m=n-1}^{0} \left(|0\rangle + e^{2\pi i [0.j_{m+1} \cdots j_n]} |1\rangle \right). \tag{4.20}$$

Expandindo o conjunto de produtos tensoriais:

$$|[j_1 j_2 j_3 \cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{2^n}} \left(\left(|0\rangle + e^{2\pi i [0.j_n]} |1\rangle \right) \otimes \cdots \otimes \left(|0\rangle + e^{2\pi i [0.j_1 j_2 j_3 \cdots j_n]} |1\rangle \right) \right). \tag{4.21}$$

Finalmente, o resultado final é obtido distribuindo $\frac{1}{\sqrt{2^n}}$ entre os n qubits:

$$|[j_1 j_2 j_3 \cdots j_n]\rangle \xrightarrow{\mathcal{QFT}} \left(\frac{|0\rangle + e^{2\pi i [0.j_n]} |1\rangle}{\sqrt{2}} \right) \otimes \cdots \otimes \left(\frac{|0\rangle + e^{2\pi i [0.j_1 j_2 j_3 \cdots j_n]} |1\rangle}{\sqrt{2}} \right). \tag{4.22}$$

4.2.1.2 Construindo o circuito

Nesta etapa da seção, o objetivo será partir de um estado inicial $|j\rangle = |[j_1 j_2 j_3 \cdots j_n]\rangle$ e, plicando usa sequência de operadores, obter um resultado equivalente a (4.22).

Considere o seguinte operador, que é uma variante do operador definido em (2.6):

$$R_\phi = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i / 2^\phi} \end{bmatrix}. \tag{4.23}$$

Para o restante desta seção, será utilizada a variante do operador de mudança de fase (4.23). Utilizando um operador de controle *ct-*R_ϕ, construído a partir do operador (4.23), em conjunto com o operador de Hadamard (2.3), será possível construir o circuito desejado.

Seja o estado inicial:

$$|j\rangle = |[j_1 j_2 j_3 \cdots j_n]\rangle.$$

Aplica-se a porta de Hadamard no primeiro qubit j_1 obtendo o estado:

$$\begin{aligned} &\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)|[j_2 j_3 \cdots j_n]\rangle, \quad \text{se} \quad j_1 = 0; \\ &\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)|[j_2 j_3 \cdots j_n]\rangle, \quad \text{se} \quad j_1 = 1. \end{aligned} \tag{4.24}$$

Como:

$$\begin{aligned} e^{2\pi i[0.0]} &= e^0 = 1; \\ e^{2\pi i[0.1]} &= e^{\pi i} = -1. \end{aligned} \tag{4.25}$$

É possível reescrever a atuação da porta de Hadamard ao primeiro qubit j_1 da seguinte forma:

$$\left(\frac{|0\rangle + e^{2\pi i[0.j_1]}|1\rangle}{\sqrt{2}}\right)|[j_2 j_3 \cdots j_n]\rangle.$$

Aplicando o operador de controle *ct-*R_2, utilizando o qubit j_2 como qubit de controle, obtém-se:

$$\left(\frac{|0\rangle + e^{2\pi i[0.j_1 j_2]}|1\rangle}{\sqrt{2}}\right)|[j_2 j_3 \cdots j_n]\rangle.$$

Aplicando os operadores de controle *ct-*R_3, *ct-*R_4, $\cdots$ *ct-*R_n, utilizando os qubits $j_3, \cdots j_n$ como qubits de controle, respectivamente, o seguinte estado é obtido:

$$\left(\frac{|0\rangle + e^{2\pi i[0.j_1 j_2 j_3 \cdots j_n]}|1\rangle}{\sqrt{2}}\right)|[j_2 j_3 \cdots j_n]\rangle.$$

A partir disso aplica-se a porta de Haddamard no segundo qubit, obtendo:

$$\left(\frac{|0\rangle + e^{2\pi i[0.j_1 j_2 j_3 \cdots j_n]}|1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle + e^{2\pi i[0.j_2]}|1\rangle}{\sqrt{2}}\right)|[j_3 \cdots j_n]\rangle.$$

A seguir, os operadores de controle *ct-*R_2, *ct-*R_3, $\cdots$ *ct-*R_{n-1} são aplicados, utilizando os qubits $j_3, \cdots j_n$ como qubits de controle, respectivamente. Resultando no seguinte estado:

$$\left(\frac{|0\rangle + e^{2\pi i[0.j_1 j_2 j_3 \cdots j_n]}|1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle + e^{2\pi i[0.j_2 j_3 \cdots j_n]}|1\rangle}{\sqrt{2}}\right)|[j_3 \cdots j_n]\rangle.$$

Continua-se este procedimento até o n-ésimo qubit obtendo o estado final:

$$\left(\frac{|0\rangle + e^{2\pi i[0.j_1 j_2 j_3 \cdots j_n]}|1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle + e^{2\pi i[0.j_2 j_3 \cdots j_n]}|1\rangle}{\sqrt{2}}\right) \cdots \left(\frac{|0\rangle + e^{2\pi i[0.j_n]}|1\rangle}{\sqrt{2}}\right).$$

A Figura 4.1 representa a implementação do circuito da transformação quântica de Fourier.

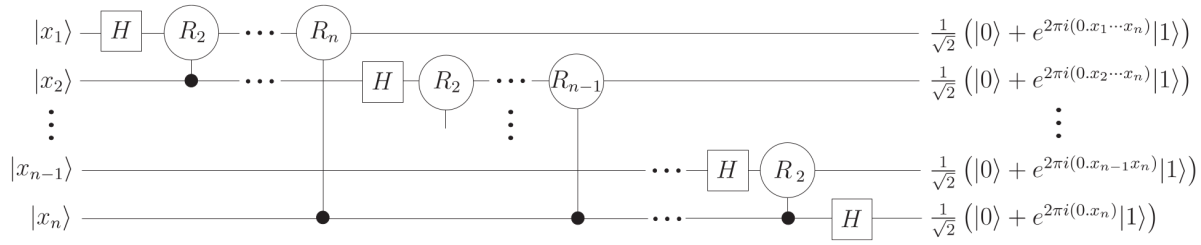


Figura 4.1: Circuito utilizado na implementação da transformação quântica de Fourier, antes a permutação em ordem reversa dos qubits de saída [KLM07].

O último passo da construção do circuito é inverter a ordem de significância dos qubits resultantes, para que o resultado final esteja precisamente correto.

4.2.2 transformação quântica inversa de Fourier

Dada a definição da transformação quântica inversa de Fourier

Definição 4.2.2. *A transformação quântica inversa de Fourier ($\mathcal{IQFT}$, $\mathcal{QFT}^\dagger$) transforma um estado quântico de n qubits $|X\rangle^n$, em um outro estado quântico de n qubits $|x\rangle^n$, a qual é definida por:*

$$|X\rangle^n \xrightarrow{\mathcal{QFT}^\dagger} |x\rangle^n : \quad x_j = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X_k e^{-i2\pi kj/N}. \tag{4.26}$$

O objetivo desta subseção será demonstrar que a transformação quântica inversa de Fourier ($\mathcal{IQFT}$) é equivalente à $\mathcal{QFT}^\dagger$, o transposto conjugado do operador $\mathcal{QFT}$, provando que a $\mathcal{QFT}$ realiza uma transformação unitária, ou seja:

$$\mathcal{QFT}\mathcal{QFT}^\dagger = \mathcal{I}.$$

Seja a transformação quântica de Fourier:

$$|j\rangle \xrightarrow{\mathcal{QFT}} \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{i2\pi kj/N}|k\rangle, \quad \forall j \in \{0, 1 \cdots, N-1\}.$$

A qual pode ser representada pelo seguinte operador:

$$QFT = \frac{1}{\sqrt{N}} \sum_{k,j=0}^{N-1} e^{i2\pi kj/N}|k\rangle\langle j|.$$

Similarmente, dada a transformação quântica inversa de Fourier:

$$|k'\rangle \xrightarrow{\mathcal{QFT}^\dagger} \frac{1}{\sqrt{N}} \sum_{j'=0}^{N-1} e^{-i2\pi k'j'/N}|j'\rangle, \quad \forall k' \in \{0, 1 \cdots, N-1\}.$$

É possível representá-la pelo seguinte operador:

$$QFT^\dagger = \frac{1}{\sqrt{N}} \sum_{k',j'=0}^{N-1} e^{-i2\pi k'j'/N}|j'\rangle\langle k'|.$$

Portanto:

$$\begin{aligned} QFTQFT^{\dagger} &= \frac{1}{\sqrt{N}} \sum_{k,j=0}^{N-1} e^{i2\pi kj/N} |k\rangle\langle j| \frac{1}{\sqrt{N}} \sum_{k',j'=0}^{N-1} e^{-i2\pi k'j'/N} |j'\rangle\langle k'| \\ &= \frac{1}{N} \sum_{k,k',j,j'=0}^{N-1} e^{i2\pi kj/N} |k\rangle\langle j| e^{-i2\pi k'j'/N} |j'\rangle\langle k'| \\ &= \frac{1}{N} \sum_{k,k',j,j'=0}^{N-1} e^{i2\pi(kj-k'j')/N} |k\rangle\langle j|j'\rangle\langle k'|. \end{aligned} \tag{4.27}$$

Dado que, se $j \neq j'$ então $\langle j|j'\rangle = 0$ e se $j = j'$ então $\langle j|j'\rangle = 1$, portanto:

$$\sum_{j,j'=0}^{N-1} \langle j|j'\rangle = \sum_{j=0}^{N-1} \langle j|j\rangle = \sum_{j=0}^{N-1} 1.$$

Substituindo em (4.27):

$$\begin{aligned} QFTQFT^{\dagger} &= \frac{1}{N} \sum_{k,k',j,j'=0}^{N-1} e^{i2\pi(kj-k'j')/N} |k\rangle\langle j|j'\rangle\langle k'| \\ &= \frac{1}{N} \sum_{k,k',j=0}^{N-1} e^{i2\pi(kj-k'j)/N} |k\rangle\langle j|j\rangle\langle k'| \\ &= \frac{1}{N} \sum_{k,k',j=0}^{N-1} e^{i2\pi(kj-k'j)/N} |k\rangle\langle k'| \\ &= \frac{1}{N} \sum_{k,k',j=0}^{N-1} e^{i2\pi j(k-k')/N} |k\rangle\langle k'| \\ &= \frac{1}{N} \sum_{j=0}^{N-1} \sum_{k,k'}^{N-1} e^{i2\pi j(k-k')/N} |k\rangle\langle k'|. \end{aligned} \tag{4.28}$$

Dado que, se $k \neq k'$ então $|k\rangle\langle k'| = 0_{N,N}$, uma matriz de N por N elementos preenchida por zeros, e se $k = k'$ então $|k\rangle\langle k'| = |k\rangle\langle k|$, portanto:

$$\sum_{k,k'=0}^{N-1} |k\rangle\langle k'| = \sum_{k=0}^{N-1} |k\rangle\langle k|.$$

Substituindo em (4.28), obtém-se o seguinte:

$$\begin{aligned} QFTQFT^{\dagger} &= \frac{1}{N}\sum_{j=0}^{N-1}\sum_{k,k'}^{N-1} e^{i2\pi j(k-k')/N}|k\rangle\langle k'| \\ &= \frac{1}{N}\sum_{j=0}^{N-1}\sum_{k=0}^{N-1} e^{i2\pi j(k-k)/N}|k\rangle\langle k| \\ &= \frac{1}{N}\sum_{j=0}^{N-1}\sum_{k=0}^{N-1} |k\rangle\langle k| \\ &= \frac{1}{N}\sum_{j=0}^{N-1} I \\ &= I. \end{aligned} \tag{4.29}$$

Portanto a transformação quântica de Fourier é unitária. Utilizando-se deste fato, é possível inferir que a $\mathcal{QFT}^{\dagger}$ pode ser implementada, construindo a $\mathcal{QFT}$ de "trás para frente" e substituindo os operador R_{ϕ} por:

$$R_{\phi} = \begin{bmatrix} 1 & 0 \\ 0 & e^{-2\pi i/2^{\phi}} \end{bmatrix}.$$

4.2.3 Custo Computacional

Analisando o circuito da $\mathcal{QFT}$, é possível notar que são aplicadas n portas de Hadamard, e $(n-1)+(n-2)+(\cdots)+(1)+(0) = \frac{(n-1)*(n-2)}{2}$ operadores de controle *ct*-R_{ϕ}, totalizando $\frac{n^2-n+2}{2}$ portas quânticas.

Como $N = 2^n$, conclui-se que o custo computacional da $\mathcal{QFT}$, assim como da $\mathcal{QFT}^{\dagger}$, é $\Theta(\log^2(N))$.

4.3 COMPARANDO CUSTOS COMPUTACIONAIS

Até então foi possível compreender que a implementação padrão da transformada discreta de Fourier, em computadores clássicos, possui custo computacional de $\Theta(N^2)$ enquanto sua equivalente para computadores quânticos, pode ser implementada utilizando $\log^2(N)$ portas quânticas, possuindo assim custo computacional $\Theta(\log^2(N))$.

Em computadores clássicos, a forma mais eficiente de implementação da transformada discreta de Fourier, até então, é a transformada rápida de Fourier, a qual tem custo computacional $\Theta(N\log(N))$.

Analogamente à transformada rápida de Fourier, existe um algoritmo quântico capaz de implementar a transformação quântica de Fourier de forma mais eficiente, se comparado com a implementação apresentada. Tal algoritmo possui custo computacional $\Theta(\log(N)\log\log(N))$ [HH00].

5 ALGORITMOS QUÂNTICOS

Para o melhor entendimento dos algoritmo quânticos presentes neste documento, este capítulo tem como propósito explicar algoritmos quânticos que são rotinas básicas usadas nos algoritmos subsequentes. Neste capítulo serão apresentados os algoritmos *Phase Kickback*, Algoritmo Quântico para Estimação de Fase e o Algoritmo Quântico para Solucionar o Problema da Ordem.

Este capítulo foi construído a partir do conteúdo presente nos livros "An Introduction to Quantum Computing" [KLM07] e "Quantum Computation and Quantum Information: 10th Anniversary Edition" [NC11]. Os algoritmos quânticos apresentados recebem uma análise matemática e computacional um pouco mais elaborada que a presente nos materiais de apoio, facilitando a compreensão de tais algoritmos.

5.1 PHASE KICK-BACK

Nesta seção será apresentada a descrição de um fenômeno chamado de *phase-kickback*. Este fenômeno é um pouco contra-intuitivo, mas é uma etapa central em vários algoritmos quânticos.

Seja U um operador e $|\psi\rangle$ um autovetor de U, tal que:

$$U|\psi\rangle = e^{i\alpha}|\psi\rangle, \quad \alpha \in [0, 2\pi[.$$

A seguir será analisando o resultado da aplicação de um operador de controle *ct-U*. Mais precisamente, será possível observar que, aplicando o operador, a fase global $e^{i\alpha}$ é transferida para o qubit de controle [CEMM98] [DJ92].

Pela definição de U e $|\psi\rangle$ e pela definição dos operadores de controle, note que:

$$ct\text{-}U|0\rangle|\psi\rangle = |0\rangle|\psi\rangle. \tag{5.1}$$

$$\begin{aligned} ct\text{-}U\,|1\rangle\,|\psi\rangle &= |1\rangle\,U\,|\psi\rangle \\ &= |1\rangle\,e^{2\pi iw}\,|\psi\rangle \\ &= e^{2\pi iw}\,|1\rangle\,|\psi\rangle. \end{aligned} \tag{5.2}$$

A partir de (5.1) e (5.2), note que se o qubit de controle estiver no estado $|+\rangle$, o seguinte resultado será obtido:

$$ct\text{-}U\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)|\psi\rangle = \left(\frac{|0\rangle + e^{i\alpha}|1\rangle}{\sqrt{2}}\right)|\psi\rangle.$$

5.2 ALGORITMO QUÂNTICO PARA ESTIMAÇÃO DE FASE

Nesta seção será discutido o algoritmo quântico para a estimação da fase, assim como seu custo computacional. Este algoritmo utiliza o fenômeno de "Phase Kick-Back", o qual foi abordado na seção anterior.

Este algoritmo foi inicialmente proposto por Alexei Yurievich Kitaev [Kit95] e pode ser encontrado em "Quantum Algorithms Revised" [CEMM98].

O algoritmo quântico de fatoração e o algoritmo quântico para resolver o problema do logaritmo discreto, ambos propostos por Peter Shor [Sho99], utilizam-se de variantes do algoritmo quântico para a estimação da fase.

A seguir é apresentada a definição formal do problema que o Algoritmo de Estimação de Fase resolve.

Problema de Estimação de Fase:
Entrada: Um determinado operador U, um autovetor de $|\psi\rangle$, de U, com autovalor $e^{2\pi iw}$ correspondente,
Problema: Obter uma boa estimativa para w.

5.2.1 Intuição do Algoritmo

Considere um operador unitário U, o qual atua sobre n qubits, com autovetor $|\psi\rangle$ e autovalor $e^{2\pi iw}$ correspondentes:

$$U|\psi\rangle = e^{2\pi iw}|\psi\rangle.$$

Dado um operador de controle *ct-U*, o qual possui um único qubit de controle, é possível observar, a partir dos resultados da Seção 5.1, que o operador atua da seguinte forma:

$$\begin{aligned} ct\text{-}U\,|1\rangle\,|\psi\rangle &= |1\rangle\,U\,|\psi\rangle \\ &= |1\rangle\,e^{2\pi iw}\,|\psi\rangle \\ &= e^{2\pi iw}\,|1\rangle\,|\psi\rangle. \end{aligned} \tag{5.3}$$

$$ct\text{-}U\,|0\rangle\,|\psi\rangle = |0\rangle\,|\psi\rangle.$$

Caso o qubit de controle esteja em uma superposição arbitrária $\alpha|0\rangle + \beta|1\rangle$, o seguinte resultado pode ser observado:

$$ct\text{-}U\,(\alpha|0\rangle + \beta|1\rangle)\,|\psi\rangle = (\alpha|0\rangle + e^{2\pi iw}\beta|1\rangle)\,|\psi\rangle.$$

Faça $w = w_{int} + w_{frac}$, onde w_{int} é a parte inteira de w e w_{frac} sua parte fracionária. Note que:

$$e^{2\pi iw} = e^{2\pi iw_{int}} \cdot e^{2\pi iw_{frac}}$$

Dado que w_{int} é um número inteiro, $e^{2\pi iw_{int}} = 1$ e com isso:

$$e^{2\pi iw} = e^{2\pi iw_{frac}}$$

Portanto será estabelecido que $0 \leq w < 1$. Também será estabelecido que w pode ser descrito, através da notação (4.6), utilizando n bits. Caso não seja possível representá-lo desta maneira, o Teorema 5.2.1 aponta que o Algoritmo Quântico para Estimação de Fase é capaz de obter uma boa estimativa para o valor de w. Utilizando (4.6), escreva w da seguinte forma:

$$w = [0.w_1w_2w_3\cdots w_n], \quad w_k \in \{0,1\}, \forall k \in \{1,2,\cdots,n\}.$$

O objetivo desta seção será implementar um circuito que, dados U e $|\psi\rangle$, é capaz de gerar o estado (5.4) e, a partir dele, estimar w, porém, antes de tal circuito ser apresentado, será

mostrado nesta seção, como tal estado $|\phi\rangle$ pode ser usado para se obter a estimativa desejada para w.

$$|\phi\rangle = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} e^{2\pi i w j} |j\rangle. \tag{5.4}$$

Dado $j \in \{0, 1, \cdots, 2^n - 1\}$, onde $n \in \mathbb{Z}_{>0}$, a notação (4.5) permite expressá-lo na forma $j = [j_1 j_2 j_3 \cdots j_n], \quad j_l \in \{0, 1\}, \forall l \in \{1, 2, \cdots, n\}$, e desenvolvendo a sequência de passos as seguir, é reescrever o estado $|\phi\rangle$ da seguinte maneira:

$$\begin{aligned}
|\phi\rangle &= \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} e^{2\pi i j w} |j\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{[j_1 j_2 \cdots j_n]=0}^{2^n-1} e^{2\pi i [j_1 j_2 \cdots j_n] w} |[j_1 j_2 \cdots j_n]\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{j_1=0}^{1} \cdots \sum_{j_n=0}^{1} e^{2\pi i [j_1 j_2 \cdots j_n] w} |[j_1 j_2 \cdots j_n]\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{j_1=0}^{1} \cdots \sum_{j_n=0}^{1} \bigotimes_{l=1}^{n} e^{2\pi i j_l w} |j_l\rangle \\
&= \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \sum_{j_l=0}^{1} e^{2\pi i j_l 2^{n-l} w} |j_l\rangle \\
&= \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left(|0\rangle + e^{2\pi i 2^{n-l} w} |1\rangle \right) \\
&= \bigotimes_{l=1}^{n} \frac{|0\rangle + e^{2\pi i 2^{n-l} w} |1\rangle}{\sqrt{2}}.
\end{aligned} \tag{5.5}$$

Abrindo o produto tensorial:

$$|\phi\rangle = \left(\frac{|0\rangle + e^{2\pi i (2^{n-1} w)} |1\rangle}{\sqrt{2}} \right) \otimes \left(\frac{|0\rangle + e^{2\pi i (2^{n-2} w)} |1\rangle}{\sqrt{2}} \right) \otimes \cdots \otimes \left(\frac{|0\rangle + e^{2\pi i (w)} |1\rangle}{\sqrt{2}} \right). \tag{5.6}$$

Como $e^{2\pi i k} = 1, \forall k \in \mathbb{Z}_{>0}$, é possível reescrever o estado $|\phi\rangle$ na seguinte forma:

$$|\phi\rangle = \left(\frac{|0\rangle + e^{2\pi i [0.w_n]} |1\rangle}{\sqrt{2}} \right) \otimes \left(\frac{|0\rangle + e^{2\pi i [0.w_{n-1} w_n]} |1\rangle}{\sqrt{2}} \right) \otimes \cdots \otimes \left(\frac{|0\rangle + e^{2\pi i [0.w_1 w_2 \cdots w_n]} |1\rangle}{\sqrt{2}} \right). \tag{5.7}$$

A partir de (5.7), é possível notar que aplicando a transformação quântica inversa de Fourier ($\mathcal{QFT}^\dagger$) sobre o estado $|\phi\rangle$, será possível obter o estado $|w_1 w_2 w_3 \cdots w_n\rangle$, e obtendo assim uma estimativa para a fase w.

Teorema 5.2.1. *Dado um operador U e um autovetor $|\psi\rangle$ de U, com autovalor $e^{2\pi i w}$ correspondente, onde $0 \leq w < 1$ e não pode ser representado com n bits.*
Então a probabilidade do algoritmo quântico para estimação de fase, ao aplicar a transformação quântica inversa de Fourier ($\mathcal{QFT}^{\dagger}$), gerar a melhor aproximação para w utilizando n bits é de, pelo menos, $4/\pi^2 = 0.405\ldots$.

Note que se w for um número irracional ou se $w = [0.w_1 w_2 w_3 \ldots w_n w_{n+1} \ldots w_m]$, onde $m \in \mathbb{Z}_{>0}, m > n$, será obtido uma estimativa para w e não seu valor exato.

A demonstração do Teorema 5.2.1 pode ser encontrada na Seção A.2 do Apêndice A, assim como em "Quantum Algorithms Revised" [CEMM98]. Aumentando o número de qubits do circuito em $O(\log \frac{1}{\epsilon})$, é possível construir um circuito capaz de gerar a melhor aproximação para w, com probabilidade de sucesso de $1 - \epsilon$, onde $0 < \epsilon \leq 1$, conforme demonstrado em "Quantum Algorithms Revised" [CEMM98].

5.2.2 Construindo o circuito

Nesta subseção será apresentado o circuito para solucionar o problema de estimação de fase. Na Subseção 5.2.1, foi visto que, uma vez que o estado $|\phi\rangle$ em (5.4) é obtido, é possível obter tal estimativa. A presente seção será iniciada, mostrando como obter $|\phi\rangle$ a partir de U e $|\psi\rangle$. Em seguida será apresentado o circuito completo para o problema.

Note que se $|\psi\rangle$ é autovetor de U com autovalor correspondente $e^{2\pi i w}$, $|\psi\rangle$ também é autovetor de U^2 com autovalor correspondente $(e^{2\pi i w})^2 = e^{2(2\pi i w)}$. Utilizando-se deste fato, é possível inferir que $|\psi\rangle$ é um autovetor de U^x com autovalor correspondente $e^{x 2\pi i w}$, $\forall x \in \mathbb{Z}_{>0}$.

Portanto, implementando o operador $ct\text{-}U^k$, colocando o qubit de controle no estado $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$ e o qubit alvo como $|\psi\rangle$, o seguinte resultado é obtido:

$$ct\text{-}U^{2^k}\left(\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)|\psi\rangle\right) = \left(\frac{|0\rangle + e^{2\pi i (2^k w)}|1\rangle}{\sqrt{2}}\right)|\psi\rangle$$

A Figura 5.1 contém o circuito capaz de gerar o estado (5.7), utilizando n operadores de controle $ct\text{-}U^{2^k}, \forall k \in \{1, 2, \cdots, n\}$, cada qual possuindo um qubit controle no estado $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$, e utilizando os qubits do estado $|\psi\rangle$ como qubits alvo.

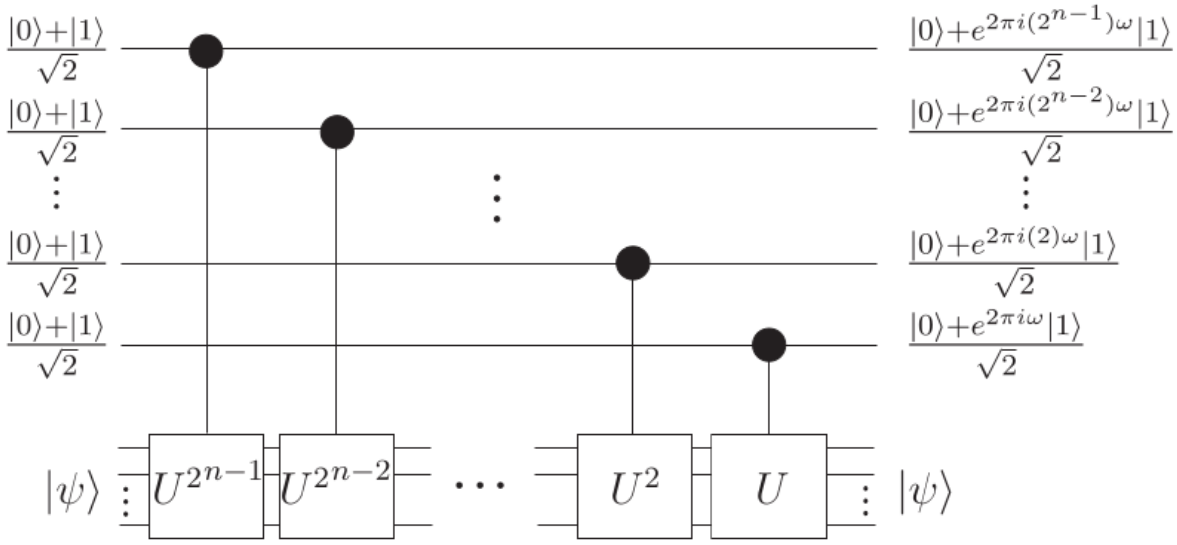


Figura 5.1: Primeira etapa da construção do circuito quântico para estimação da fase [KLM07].

Logo, é possível criar o circuito capaz de obter uma boa estimativa para o autovalor w, a partir de operador U e um estado $|\psi\rangle$ com autovalor $e^{2\pi iw}$ correspondente.

A Figura 5.2 adiciona a $\mathcal{QFT}^\dagger$ ao circuito descrito na Figura 5.1.

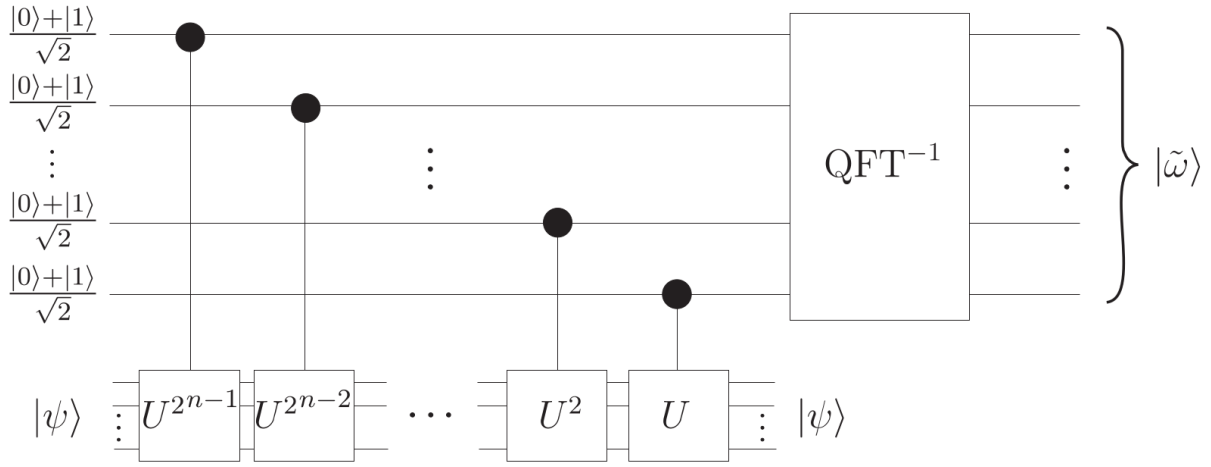


Figura 5.2: Segunda etapa da construção do circuito quântico para estimação da fase [KLM07].

É desejado que os qubits de controle estejam no estado:

$$\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \otimes \cdots \otimes \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right).$$

Portanto inicia-se n qubits de controle no estado $|0\rangle$, em seguida se aplica um operador de Hadamard sobre cada um destes qubits.

Outra alternativa é utilizar uma transformação quântica de Fourier $\mathcal{QFT}$ sobre os n qubits de controle, os quais estão no estado $|0\rangle^n$, uma vez que a seguinte igualdade é verdadeira:

$$\mathcal{QFT}\,|0\rangle^n = H^{\otimes n}\,|0\rangle^n.$$

Finalmente, o circuito completo, capaz de estimar w, é apresentado na Figura 5.3.

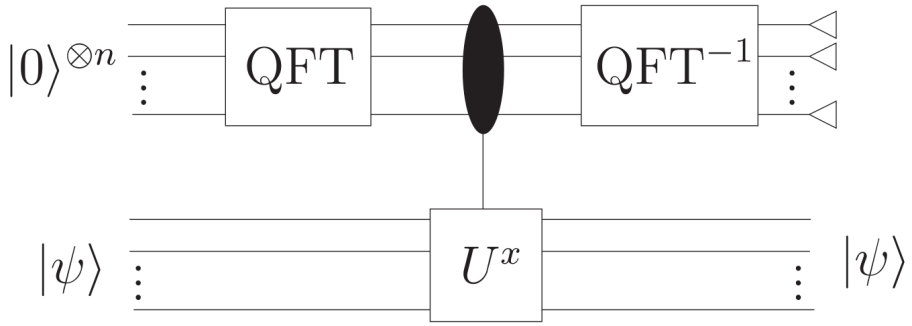


Figura 5.3: Circuito quântico para estimação da fase [KLM07].

Onde U^x representa uma sequência de n operadores $ct\text{-}U^{2^k}$, cada qual utilizando o k-ésimo qubit como qubit de controle $\forall k \in \{1, 2, \cdots, n\}$.

5.2.3 Algoritmo e Complexidade Computacional

Nesta subseção será apresentado o Algoritmo 5, onde m é o número de qubits necessários para representar o estado $|\psi\rangle$ e n é o numero de qubits de controle, ou seja o número de casas após a virgula que desejamos descobrir. Ainda nesta subseção será realizada uma análise do custo computacional deste algoritmo.

Algoritmo 5 *Estimação de Fase*$(U, |\psi\rangle^m, n)$:

1: Inicia um registrador no estado $|0\rangle^n|\psi\rangle^m$.
2: Aplica a $\mathcal{QFT}$ sobre os n primeiros qubits, $|0\rangle^n$, gerando o estado $\frac{1}{\sqrt{2^n}}\sum_{j=0}^{2^n-1}|j\rangle|\psi\rangle^m$.
3: Aplica a sequência de operadores de controle U^x, gerando o estado $\frac{1}{\sqrt{2^n}}\sum_{j=0}^{2^n-1}e^{2\pi i w j}|j\rangle|\psi\rangle^m$.
4: Aplica a $\mathcal{QFT}^\dagger$ sobre o estado $\sum_{j=0}^{2^n-1}e^{2\pi i w j}|j\rangle$, obtendo o estado $|y\rangle|\psi\rangle^m$.
5: Realiza uma medição no estado $|y\rangle$, obtendo um valor y.
6: **return** $\frac{y}{2^n} = \tilde{w}$, como uma estimação para w.

Dado que o operador U é arbitrário, suponha que o custo computacional de U_x seja:

$$O(f(m,n)) \tag{5.8}$$

A Tabela 5.1 a seguir, contém uma análise do custo computacional das linhas mais relevantes do Algoritmo 5, onde $N = 2^n$.

Tabela 5.1: Análise do custo computacional das linhas mais importantes do Algoritmo 5.

Linha	Custo computacional em Tempo	Referência
2	$O((log(N))^2)$	Anánálise da QFT
3	$O(log(N)f(m,n))$	(5.8)
4	$O((log(N))^2)$	Análise da QFT
Total	$O(log(N))^2 + log(N)f(m,n))$	Linhas 2 e 3

Analisando o Algoritmo 5, pode ser afirmado que a sua probabilidade de sucesso é:

$$\frac{4}{\pi^2} = 0.405\ldots.$$

5.3 ALGORITMO QUÂNTICO PARA SOLUÇÃO DO PROBLEMA DA ORDEM

Nesta seção será abordado o algoritmo quântico utilizado para resolver o problema da ordem, definido a seguir, assim como seu custo computacional. Este algoritmo foi inicialmente proposto por Peter Shor [Sho99] e é similar ao algoritmo de estimação de fase em muitos aspectos, desempenhando um papel muito importante nos algoritmo de fatoração e no algoritmo que soluciona o problema do logaritmo discreto.

Definição 5.3.1. *Dado dois números $a, N \in \mathbb{Z}_{>0}$, tais que $a < N$, a ordem de a módulo N, é o menor inteiro positivo r que satisfaz a seguinte equação:*

$$a^r \equiv 1 \pmod{N}. \tag{5.9}$$

A seguir é apresentada a definição formal do problema da ordem.

Problema da Ordem:
Entrada: Dois números $a, N \in \mathbb{Z}_{>0}$, tais que $MDC(a, N) = 1$ e $a < N$.
Problema: Obter o menor inteiro positivo r tal que $a^r \equiv 1 \pmod{N}$.

5.3.1 Intuição do Algoritmo

Considere o operador U_a, o qual pode ser implementado em tempo polinomial [BCDP96]:

$$U_a|s\rangle = |sa \bmod N\rangle, \quad s, a, N \in \mathbb{Z}_{>0}, s < N.$$

Para solucionar o problema da ordem, será construído um circuito similar ao circuito utilizado para solucionar o problema da estimação da fase, onde o operador U utilizado no algoritmo de estimação de fase é substituído pelo operador U_a.

Nesta seção será utilizado o Lema 5.3.1 para provar que um determinado estado $|u_k\rangle$ é autovetor de U_a, com autovalor $e^{2\pi i w}$, onde $w = \frac{k}{r}$. Desta forma o Algoritmo Quântico para Estimação de Fase, é capaz de estimar $\frac{k}{r}$. Em sequência é utilizando o Lema 5.3.2, para demonstrar que $\sum_{k=0}^{r-1} |u_k\rangle = |1\rangle$, e com isso o algoritmo estimará um valor $\frac{k}{r}$, onde $k \in \{0, 1, \cdots, r-1\}$ é escolhido uniformemente ao acaso após uma medição.

Atente-se ao Lema 5.3.1, o qual será útil posteriormente.

Lema 5.3.1. *Sejam $a, N \in \mathbb{Z}_{>0}$, tais que $mdc(a, N) = 1$. Seja r a ordem de a módulo N e $k \in \{0, 1, \cdots, N-1\}$. Então, $\forall t \in \{1, 2, \cdots, r-1\}$:*

$$\sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+t)} |a^{s+t} \bmod N\rangle = \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^s \bmod N\rangle. \tag{5.10}$$

Demonstração.

$$\begin{aligned}\sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+t)} |a^{s+t} \bmod N\rangle &= \sum_{s=0}^{r-1-t} e^{-2\pi i \frac{k}{r}(s+t)} |a^{s+t} \bmod N\rangle + \sum_{s=r-t}^{r-1} e^{-2\pi i \frac{k}{r}(s+t)} |a^{s+t} \bmod N\rangle \\ &= \sum_{s=t}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^s \bmod N\rangle + \sum_{s=r-t}^{r-1} e^{-2\pi i \frac{k}{r}(s+t)} |a^{s+t} \bmod N\rangle.\end{aligned} \tag{5.11}$$

Reescrevendo o segundo somatório:

$$\sum_{s=r-t}^{r-1} e^{-2\pi i \frac{k}{r}(s+t)} |a^{s+t} \bmod N\rangle = \sum_{s=0}^{r-1-(r-t)} e^{-2\pi i \frac{k}{r}(s+t+(r-t))} |a^{s+t+(r-t)} \bmod N\rangle. \tag{5.12}$$

Portanto:

$$\begin{aligned}
&\sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+t)} |a^{s+t} \bmod N\rangle \\
&= \sum_{s=t}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle + \sum_{s=0}^{r-1-(r-t)} e^{-2\pi i \frac{k}{r}(s+t+(r-t))} |a^{s+t+(r-t)} \bmod N\rangle \\
&= \sum_{s=t}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle + \sum_{s=0}^{t-1} e^{-2\pi i \frac{k}{r}(s+r)} |a^{s+r} \bmod N\rangle \\
&= \sum_{s=t}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle + \sum_{s=0}^{t-1} e^{-2\pi i \frac{k}{r}(s+r)} |a^{s} a^{r} \bmod N\rangle.
\end{aligned} \tag{5.13}$$

Como $a^r \equiv 1 \pmod N$:

$$\sum_{s=0}^{t-1} e^{-2\pi i \frac{k}{r}(s+r)} |a^{s} a^{r} \bmod N\rangle = \sum_{s=0}^{t-1} e^{-2\pi i \frac{k}{r}(s+r)} |a^{s} \bmod N\rangle. \tag{5.14}$$

Substituindo o resultado de (5.14) em (5.13):

$$\begin{aligned}
&\sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+t)} |a^{s+t} \bmod N\rangle \\
&= \sum_{s=t}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle + \sum_{s=0}^{t-1} e^{-2\pi i \frac{k}{r}(s+r)} |a^{s} \bmod N\rangle \\
&= \sum_{s=t}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle + \sum_{s=0}^{t-1} \left(e^{-2\pi i \frac{k}{r}(s)} e^{-2\pi i \frac{k}{r}(r)} \right) |a^{s} \bmod N\rangle \\
&= \sum_{s=t}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle + \sum_{s=0}^{t-1} \left(e^{-2\pi i \frac{k}{r}(s)} e^{-2\pi i k} \right) |a^{s} \bmod N\rangle.
\end{aligned} \tag{5.15}$$

Como $e^{-2\pi ik} = (e^{2\pi i})^{-k} = (1)^{-k} = 1$, então:

$$\begin{aligned}
&\sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+t)} |a^{s+t} \bmod N\rangle \\
&= \sum_{s=t}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle + \sum_{s=0}^{t-1} \left(e^{-2\pi i \frac{k}{r}(s)} 1 \right) |a^{s} \bmod N\rangle \\
&= \sum_{s=t}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle + \sum_{s=0}^{t-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle \\
&= \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle.
\end{aligned} \tag{5.16}$$

□

Agora considere o estado:

$$|u_k\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |a^{s} \bmod N\rangle. \tag{5.17}$$

É possível verificar que o estado $|u_k\rangle$ é um autovetor de U_a, com autovalor de $e^{2\pi i \frac{k}{r}}$, da seguinte forma:

$$\begin{aligned}
U_a|u_k\rangle &= U_a \left(\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |a^{s} \bmod N\rangle \right) \\
&= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} U_a |a^{s} \bmod N\rangle \\
&= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |a^{s+1} \bmod N\rangle \\
&= e^{2\pi i \frac{k}{r}} e^{-2\pi i \frac{k}{r}} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |a^{s+1} \bmod N\rangle \\
&= e^{2\pi i \frac{k}{r}} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+1)} |a^{s+1} \bmod N\rangle.
\end{aligned} \tag{5.18}$$

Como, pelo Lema 5.3.1:

$$\sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+1)} |a^{s+1} \bmod N\rangle = \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |a^{s} \bmod N\rangle. \tag{5.19}$$

Substituindo em (5.18):

$$\begin{aligned} U_a|u_k\rangle &= e^{2\pi i\frac{k}{r}}\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1} e^{-2\pi i\frac{k}{r}(s+1)}|a^{s+1} \bmod N\rangle \\ &= e^{2\pi i\frac{k}{r}}\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1} e^{-2\pi i\frac{k}{r}(s)}|a^{s} \bmod N\rangle \\ &= e^{2\pi i\frac{k}{r}}|u_k\rangle. \end{aligned} \tag{5.20}$$

Portanto $|u_k\rangle$ é um autovetor de U_a com autovalor de $e^{2\pi i\frac{k}{r}}$.

Utilizando operador U_a e o estado $|u_k\rangle$, como entradas para o algoritmo de estimação de fase, este retornará o estado (5.21), onde n determina a precisão do algoritmo. Isto ocorre pois $e^{2\pi i\frac{k}{r}}$ é o autovalor de $|u_k\rangle$ em relação à U_b, ou seja, o valor w estimado pelo Algoritmo 5 é $\frac{k}{r}$.

$$|0\rangle^n|u_k\rangle \longmapsto |\widetilde{k/r}\rangle|u_k\rangle, \quad k \in \{0, 1, \cdots, r-1\}. \tag{5.21}$$

Porém, como r não é conhecido, não é possível preparar o estado $|u_k\rangle$ e portando é necessário encontrar uma outra forma para que seja obtido um resultado semelhante.

Lema 5.3.2. *Dado dois números $x, m \in \mathbb{Z}_{>0}$, tais que $mdc(x, m) = 1$. Faça r a ordem de x módulo m. Dado um estado quântico $|u_k\rangle$, tal que:*

$$|u_k\rangle = \frac{1}{\sqrt{r}}\sum_{s=0}^{r-1} e^{-2\pi i\frac{k}{r}s}|x^s \bmod m\rangle. \tag{5.22}$$

Então:

$$\frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}|u_k\rangle = |1\rangle. \tag{5.23}$$

Demonstração. Inicialmente, pela definição de $|u_k\rangle$, note que:

$$\begin{aligned} \frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}|u_k\rangle &= \frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1} e^{-2\pi i\frac{k}{r}s}|x^s \bmod m\rangle \\ &= \frac{1}{\sqrt{r}}\frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}\sum_{s=0}^{r-1} e^{-2\pi i\frac{k}{r}s}|x^s \bmod m\rangle. \end{aligned} \tag{5.24}$$

Dado que, para um estado qualquer, a soma dos quadrados de seus coeficientes deve ser igual a 1. Ou seja, para um estado qualquer $|\phi\rangle = \sum_s \alpha_s|s\rangle$:

$$\sum_s |\alpha_s|^2 = 1. \tag{5.25}$$

Fazendo:

$$\frac{1}{\sqrt{r}}\frac{1}{\sqrt{r}}\sum_{k=0}^{r-1} e^{-2\pi i\frac{k}{r}s} = \alpha_s, \quad \forall s \in \{0, 1\cdots, r-1\}. \tag{5.26}$$

Então:

$$\frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}|u_k\rangle = \sum_{s=0}^{r-1}\alpha_s|x^s \bmod m\rangle. \tag{5.27}$$

É possível notar que $|x^s \bmod m\rangle = |1\rangle$ se, e somente se, $s \equiv 0 \pmod r$. Como $s \in \{0, 1, \cdots, r-1\}$, o único valor de s que satisfaz a equação $s \equiv 0 \pmod r$ é $s = 0$. Portanto:

$$|1\rangle = \alpha_0|x^0 \bmod m\rangle. \tag{5.28}$$

Utilizando (5.25), é possível inferir que:

$$\sum_{s=0}^{r-1}|\alpha_s|^2 = 1. \tag{5.29}$$

Fazendo $s = 0$ em (5.26):

$$\begin{aligned}\alpha_0 &= \frac{1}{\sqrt{r}}\frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}e^{-2\pi i\frac{k}{r}0}\\ &= \frac{1}{r}\sum_{k=0}^{r-1}(1)\\ &= 1.\end{aligned} \tag{5.30}$$

A partir de (5.30) e (5.29):

$$\begin{aligned}&\sum_{s=0}^{r-1}|\alpha_s|^2 = 1\\ &|\alpha_0|^2 + \sum_{s=1}^{r-1}|\alpha_s|^2 = 1\\ &|1|^2 + \sum_{s=1}^{r-1}|\alpha_s|^2 = 1\\ &\sum_{s=1}^{r-1}|\alpha_s|^2 = 0.\end{aligned} \tag{5.31}$$

A partir de (5.31), é possível concluir que:

$$\alpha_s = 0, \quad \forall s \in \{1, 2\cdots, r-1\}.$$

Portanto:

$$\begin{aligned}\sum_{s=0}^{r-1}\alpha_s|x^s \bmod m\rangle &= \alpha_0|x^0 \bmod m\rangle + \sum_{s=1}^{r-1}\alpha_s|x^s \bmod m\rangle\\ &= \alpha_0|x^0 \bmod m\rangle + \sum_{s=1}^{r-1}0|x^s \bmod m\rangle\\ &= \alpha_0|x^0 \bmod m\rangle.\end{aligned} \tag{5.32}$$

Substituindo o resultado (5.32) em (5.27):

$$\frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}|u_k\rangle = \alpha_0|x^0 \bmod m\rangle \tag{5.33}$$

Portanto, a partir de (5.28) e (5.32), conclui-se que:

$$\frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}|u_k\rangle = |1\rangle \tag{5.34}$$

□

Pelo Lema 5.3.2:

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

Seja n um número inteiro positivo arbitrário, o qual determina a precisão do algoritmo e m a quantidade de qubits suficiente para a aplicação do operador U_a. Aplicando o algoritmo de estimação de fase no estado (5.35):

$$\begin{aligned}|0\rangle^n|1\rangle^m &= |0\rangle\left(\frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}|u_k\rangle\right)\\ &= \frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}|0\rangle|u_k\rangle.\end{aligned} \tag{5.35}$$

A seguinte superposição é obtida:

$$\frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}|\widetilde{k/r}\rangle|u_k\rangle.$$

Analisando o primeiro registrador, é possível observar uma superposição de iguais probabilidades dos estados $|\widetilde{k/r}\rangle|u_k\rangle$ para $k \in \{0, 1, \cdots, r-1\}$. Ignorando o segundo registrador e aplicando uma medição no primeiro registrador é obtido um valor x, sendo $\frac{x}{2^n}$ um valor estimado para $\frac{k}{r}$, com $k \in \{0, 1, \cdots, r-1\}$.

Lema 5.3.3. *Dado um operador U e um autovetor $|\psi\rangle$ de U, com autovalor $e^{2\pi i w}$ correspondendo, onde $0 \leq w < 1$ e não pode ser representado com n bits.*
Então, com probabilidade $8/\pi^2$, o algoritmo de estimação de fase irá produzir um valor $\hat{x} \in \{0, \cdots, 2^n - 1\}$, de modo a satisfazer:

$$\left|\frac{\hat{x}}{2^n} - w\right| \leq \frac{1}{2^n} \tag{5.36}$$

Utilizando o Lema 5.3.3 apresentado na Seção A.2 do Apêndice A, é possível inferir que, com probabilidade $8/\pi^2$, o algoritmo irá produzir um valor $x \in \{0, \cdots, 2^n - 1\}$, tal que:

$$\left|\frac{x}{2^n} - \frac{k}{r}\right| \leq \frac{1}{2^n}. \tag{5.37}$$

Sendo n o número de qubits de controle, é possível escolher n suficientemente grande, tal que $2^n \geq 2r^2$, algo que pode ser garantido utilizando o valor fornecido N.

Lema 5.3.4. *Sejam $a, N \in \mathbb{Z}_{>0}$, tais que $mdc(a, N) = 1$ e $a < N$. Seja r a ordem de a módulo N. Então $r < N$.*

A partir do Teorema 5.3.4, provado em B.2.10 no Apêndice B:

$$N > r \tag{5.38}$$

Portanto $2N^2 > 2r^2$. Escolhendo n, tal que $n = \lceil \log(2N^2) \rceil$:

$$2^n \geq 2N^2 > 2r^2. \tag{5.39}$$

A partir disso, é possível inferir que o valor x, resultante da medição, satisfaz (5.40), onde $k \in \{0, 1, \cdots, r-1\}$.

$$\left| \frac{x}{2^n} - \frac{k}{r} \right| \leq \frac{1}{2^n} \leq \frac{1}{2r^2}. \tag{5.40}$$

O Algoritmo 6, surge para enunciar o algoritmo de frações continuas, apenas para que o leitor compreenda sua rotina. Tal algoritmo possui custo computacional $O(n)$ [KLM07].

Algoritmo 6 *Frações Contínuas*(w, n):

1: $k', r' \leftarrow$ valores para k' e r' que satisfazem $\left| w - \frac{k'}{r'} \right| \leq \frac{1}{2^n}$ e que $mdc(k', r') = 1$.
2: **return** k', r'.

Como (5.40), o algoritmo de frações contínuas possuirá um único retorno possível para k' e r', o qual satisfaz $\frac{k'}{r'} = \frac{k}{r}$ [KLM07]. Portanto, utilizando o algoritmo de frações continuas, é possível inferir os valores k' e r', tais que $\frac{k'}{r'} = \frac{k}{r}$ e $mdc(k', r') = 1$.

Caso $a^{r'} \not\equiv 1 \pmod{N}$, então, possivelmente, durante a medição o estado colapsou em um valor de x, tal que:

$$\left| \frac{x}{2^n} - \frac{k}{r} \right| > \frac{1}{2^n}. \tag{5.41}$$

Outra possibilidade é que os valores k e r possuam um fator comum, isto é $mdc(k, r) \neq 1$, neste caso r e r' serão diferentes.

Executando o algoritmo duas vezes, são obtidos os valores $\{k', r'\}$ e $\{k'', r''\}$. Então, com probabilidade $6/\pi^2$, $mmc(r', r'') = r$, o qual pode ser computado em $O((\log(r))^2)$ [KLM07]. Existem outras maneiras de contornar este problema, as quais podem ser encontradas em "Quantum Computation and Quantum Information: 10th Anniversary Edition" [NC11].

5.3.2 Construindo o Circuito

A construção do circuito é semelhante à construção do circuito utilizado para a estimação da fase, basta substituir os operadores de controle *ct-U* utilizados por operadores de controle *ct-*U_a e utilizar o estado $|\psi\rangle$ como $|1\rangle$. O circuito construído pode ser observado na Figura 5.4.

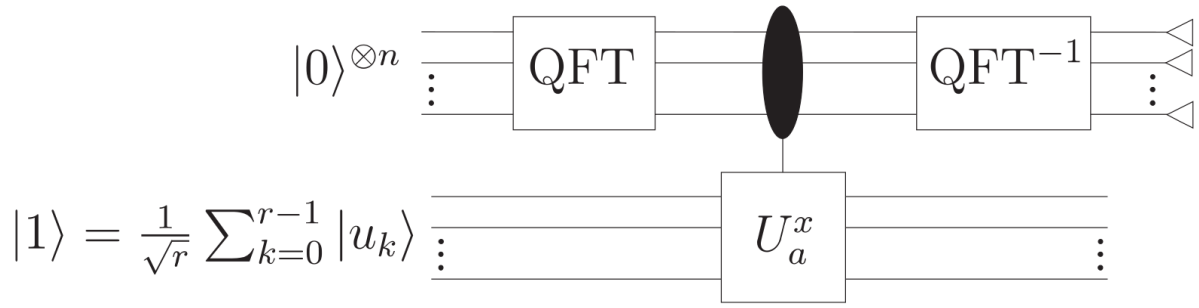


Figura 5.4: Circuito quântico para solucionar o problema da ordem [KLM07].

5.3.3 Algoritmo e Complexidade Computacional

Nesta subseção será apresentado um algoritmo $Ordem(a, N, n, U_a)$, capaz de solucionar o problema da ordem, assim como uma análise de seu custo computacional.

Algoritmo 7 $Ordem(a, N, U_a)$:

1: $n \leftarrow \lceil \log(2N^2) \rceil$.
2: Inicia um registrador no estado $|0\rangle^n|1\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |0\rangle^n|u_k\rangle$.
3: Aplica a $\mathcal{QFT}$ sobre os n primeiros qubits, $|0\rangle^n$, gerando o estado $\frac{1}{\sqrt{r2^n}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} |j\rangle|u_k\rangle$.
4: Aplica a sequência de operadores de controle U_a^x, gerando o estado $\frac{1}{\sqrt{r2^n}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} e^{2\pi i \frac{k}{r} j} |j\rangle|u_k\rangle$.
5: Aplica a $\mathcal{QFT}^\dagger$ sobre o estado $\frac{1}{\sqrt{r2^n}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} e^{2\pi i \frac{k}{r} j} |j\rangle$, obtendo o estado $\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\widetilde{k/r}\rangle|u_k\rangle$.
6: Realiza uma medição no estado $\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\widetilde{k/r}\rangle$, obtendo um valor x', onde, para algum $k \in \{0, 1, \cdots, r-1\}$, $\frac{x'}{2^n} = \widetilde{\frac{k}{r}}$.
7: $k', r' \leftarrow$ *Frações Continuas*$\left(\frac{x'}{2^n}, n\right)$.
8: Repetir os passos $1 - 5$, obtendo um valor x''
9: $k'', r'' \leftarrow$ *Frações Continuas*$\left(\frac{x''}{2^n}, n\right)$.
10: $r \leftarrow mmc(r', r'')$
11: **return** r.

Dado que o custo computacional para realizar operações de multiplicação modular, $a^j \bmod N$, é $O(\log(N) \log\log(N) \log\log\log(N))$ [BCDP96], é possível definir o custo computacional de implementar o operador U_a^x, como o custo computacional de implementar n operações de multiplicação modular [KLM07], ou seja:

$$O(n \log(N) \log\log(N) \log\log\log(N)) \tag{5.42}$$

Utilizando a Linha 1 do Algoritmo 7:

$$n \leftarrow \lceil \log(2N^2) \rceil \therefore O(n) = O(\log(N)). \tag{5.43}$$

Portanto o custo computacional de implementar o operador U_a^x é:

$$O((\log(N))^2 \log\log(N) \log\log\log(N)) \tag{5.44}$$

Tabela 5.2: Análise do custo computacional das linhas mais importantes do Algoritmo 7.

Linha	Custo computacional em Tempo	Algoritmo Quântico
3	$O((\log(N))^2)$	Análise da QFT
4	$O((\log(N))^2 \log\log(N) \log\log\log(N))$	(5.44)
4	$O((\log(N))^2)$	Análise da QFT
7	$O(\log(N))$	(5.43) [KLM07]
9	$O(\log(N))$	(5.43) [KLM07]
10	$O((\log(N))^2)$	(5.38) [KLM07]
Total	$O((\log(N))^2 \log\log(N) \log\log\log(N))$	Linha 3

Analisando o Algoritmo 7, pode ser afirmado que a sua probabilidade de sucesso é:

$$\frac{8}{\pi^2}\frac{8}{\pi^2}\frac{6}{\pi^2} = \frac{384}{\pi^6} = 0.399\ldots.$$

6 ALGORITMOS DE SHOR

Em 1994, Peter Shor propôs dois algoritmos quânticos capazes de resolver problemas extremamente relevantes para a computação. Um destes algoritmos é capaz de realizar a fatoração de um número inteiro em tempo polinomial enquanto o segundo resolve o problema do logaritmo discreto, também em tempo polinomial [Sho94].

Neste capítulo serão utilizados os conceitos desenvolvidos nos capítulos anteriores para construir os algoritmos propostos por Peter Shor, explicando em detalhes o funcionamento de tais algoritmos.

6.1 FATORAÇÃO DE UM NÚMERO INTEIRO

Esta seção será dedicada a analisar o algoritmo proposto por Peter Shor para fatorar um número inteiro em tempo polinomial. Este algoritmo utiliza-se de uma redução probabilística capaz de reduzir o problema da fatoração ao problema da ordem em tempo polinomial. Desta forma, é possível utilizar o Algoritmo 7 para solucionar o problema.

Inicialmente o problema da fatoração será definido e reduzido gradualmente em problemas menos complexos, o ponto crucial para a redução do algoritmo é a utilização do Teorema 6.1.3, o qual permite completar a redução ao problema da ordem.

Esta seção reúne conteúdos de ambos os livros "An Introduction to Quantum Computing" [KLM07] e "Quantum Computation and Quantum Information: 10th Anniversary Edition" [NC11], resultando em uma análise que engloba aspectos distintos do problema e do algoritmo. Esta análise foi moldada para que se possa compreender de forma ampla o algoritmo proposto por Shor para fatoração, e para que as passagens matemáticas aconteçam de forma clara para o leitor.

A seguir é apresentada a definição do problema da fatoração.

Problema da fatoração:
Entrada: Um número $N \in \mathbb{Z}_{>0}$.
Problema: Obter os valores inteiros $p_1, p_2, \cdots, p_l, a_1, a_2, \cdots, a_l$, onde p_i são primos distintos $\forall i \in \{1, 2, \cdots, l\}$, $l \in \mathbb{Z}_{>0}$ e $N = p_1^{a_1} p_2^{a_2} \cdots p_l^{a_l}$.

6.1.1 Redução ao Problema da Ordem

Nesta subseção será demonstrado que é possível solucionar o problema da fatoração transformando-o no problema da ordem.

Uma vez que o teste de primariedade é um problema que admite algoritmo polinomial [AKS04] e a solução para $N = 1$ é trivial, N será tratado como um número composto.

Note que se 2 é um divisor de N, é possível reduzir o problema para encontrar os divisores de $\frac{N}{2}$. Este processo pode ser repetido até que 2 não seja um divisor do número desejado, portanto o problema pode ser reduzido à fatoração de um número ímpar.

Problema da fatoração de número ímpar:
Entrada: Um número ímpar composto $N \in \mathbb{Z}_{>0}$
Problema: Obter os valores inteiros $p_1, p_2, \cdots, p_l, a_1, a_2, \cdots, a_l$, onde p_i são primos distintos $\forall i \in \{1, 2, \cdots, l\}$, $l \in \mathbb{Z}_{>0}$ e $N = p_1^{a_1} p_2^{a_2} \cdots p_l^{a_l}$.

Caso N seja uma potência perfeita, ou seja $N = p_1^{a_1}$, o custo computacional de fatorá-lo torna-se essencialmente linear [Ber98]. Portanto o problema é reduzido a fatorar um número inteiro positivo ímpar que não é uma potência perfeita. Sendo assim pode se afirmar que:

$$N = N_1 \cdot N_2, \quad 1 < N_1, N_2 < N$$

O problema é reduzido à encontrar valores N_1 e N_2, satisfazendo a equação acima. Caso N_1 ou N_2 não sejam primos nem uma potência perfeita, basta utilizar o mesmo algoritmo utilizado para determinar N_1 e N_2 a partir de N.

Dado que o teste de primariedade é um problema que admite algoritmo polinomial, assim como determinar se um número é potência perfeita, como mencionado anteriormente, o problema pode ser reduzido ao problema de encontrar um divisor não trivial de um número inteiro positivo impar, que não é uma potência perfeita.

Problema de encontrar um divisor não trivial de um número inteiro positivo impar que não é uma potência perfeita:
Entrada: Um número impar composto $N \in \mathbb{Z}_{>0}$ que possui pelo menos dois fatores primos distintos.
Problema: Obter dois valores $N_1, N_2 \in \mathbb{Z}_{>0}$, tais que $N = N_1 \cdot N_2$, sendo $1 < N_1, N_2 < N$.

Teorema 6.1.1. *Seja $N \in \mathbb{Z}_{>0}$ um número ímpar composto que possui pelo menos dois fatores primos distintos. Seja $x \in \{2, 3, \cdots, N-2\}$ de modo a satisfazer simultaneamente as seguintes equações:*

$$x^2 \equiv 1 \pmod{N} \tag{6.1}$$

$$x \not\equiv \pm 1 \pmod{N} \tag{6.2}$$

Utilizando x é possível determinar dois inteiros N_1, N_2, tais que $N = N_1 \cdot N_2$ onde $1 < N_1, N_2 < N$.

Demonstração. Dado que $x^2 \equiv 1 \pmod{N}$ então $N \mid (x+1)(x-1)$, pois:

$$\begin{aligned}
&x^2 \equiv 1 \pmod{N};\\
&x^2 = 1 + kN, \quad k \in \mathbb{Z}_{>0};\\
&x^2 - 1 = kN, \quad k \in \mathbb{Z}_{>0};\\
&\frac{x^2-1}{N} = k, \quad k \in \mathbb{Z}_{>0};\\
&\frac{(x+1)(x-1)}{N} = k, \quad k \in \mathbb{Z}_{>0}.
\end{aligned} \tag{6.3}$$

Dado que $x \in \{2, 3, \cdots, N-2\}$, note que $(x-1) < (x+1) < N$. Então existem valores para N_1 e N_2 tais que:

$$N_1 \cdot N_2 = N, \quad 1 < N_1, N_2 < N. \tag{6.4}$$

$$\frac{(x+1)}{N_1}\frac{(x-1)}{N_2} = k, \quad k \in \mathbb{Z}_{>0}. \tag{6.5}$$

$$\frac{(x+1)}{N_1} = k_1, \quad k_1 \in \mathbb{Z}_{>0}. \tag{6.6}$$

$$\frac{(x-1)}{N_2} = k_2, \quad k_2 \in \mathbb{Z}_{>0}. \tag{6.7}$$

$$k = k_1 \cdot k_2, \tag{6.8}$$

Utilizando (6.4), (6.6) e (6.7), observe que:

$$\begin{aligned} mdc(N,(x+1)) &= N_1; \\ mdc(N,(x-1)) &= N_2. \end{aligned} \tag{6.9}$$

□

O Algoritmo 8 é responsável por calcular o máximo divisor comum (mdc). Este algoritmo recebe como entrada dois valores $a, b \in \mathbb{Z}_{>0}$, tais que $a > b$, e possui custo computacional de $O(\log(b))$ [Mol08].

Algoritmo 8 *Algoritmo de Euclides*(a, b):

1: $a \leftarrow a \bmod b$.
2: **if** $a = 0$ **then**
3: **return** b.
4: **end if**
5: **return** *Algoritmo de Euclides*(b, a).

O Teorema 6.1.1, em conjunto com o Algoritmo 8, implica que o problema de encontrar um divisor não trivial de um número inteiro positivo impar que não é uma potência perfeita, pode ser reduzido polinomialmente ao problema de encontrar x.

Problema de encontrar x:
Entrada: Um número impar composto $N \in \mathbb{Z}_{>0}$ que possui pelo menos dois fatores primos distintos.
Problema: Encontrar um inteiro $x \in \{2, 3, \cdots, N-2\}$ que satisfaça simultaneamente as equações:

$$x^2 \equiv 1 \pmod{N} \tag{6.10}$$

$$x \not\equiv \pm 1 \pmod{N} \tag{6.11}$$

Dado que o objetivo é reduzir ao problema da ordem, relembre

Problema da Ordem:
Entrada: Dois números inteiros y, N, tais que $mdc(y, N) = 1$ e $y < N$.
Problema: Obter o menor inteiro positivo r tal que $y^r \equiv 1 \pmod{N}$

Teorema 6.1.2. *(Teorema de Euler) Seja y um inteiro coprimo de N. Então $y^{\varphi(N)} \equiv 1 \pmod{N}$*

A partir do Teorema de Euler, demonstrado em B.2.5 é possível inferir que, se $mdc(y, N) = 1$ e $1 \leq y < N$, então existe um r, tal que $y^r \equiv 1 \pmod{N}$. Portanto, utilizando um algoritmo para encontrar a ordem de y módulo N, sempre será obtido um resultado.

Teorema 6.1.3. *Seja $N = p_1^{a_1} p_2^{a_2} \cdots p_l^{a_l}$, onde $l > 1$, p_j são números primos ímpares distintos e $a_j \in \mathbb{Z}_{>0}$, $\forall j \in \{1, 2, \cdots, l\}$.*
Seja $y \in \mathbb{Z}_N^{\times}$ escolhido uniformemente ao acaso e r a ordem de y módulo N.
Então, com probabilidade de, pelo menos, $1 - \frac{1}{2^{l-1}}$, as seguintes equações serão satisfeitas:

$$r \equiv 0 \pmod{2} \tag{6.12}$$

$$y^{\frac{r}{2}} \not\equiv -1 \pmod{N} \tag{6.13}$$

A demonstração do Teorema 6.1.3 pode ser encontrada na Seção B.3 do Apêndice B.

Dado que o Teorema 6.1.3 possui alta probabilidade de sucesso, será definido que o y satisfaz as equações do mesmo.

Utilizando um algoritmo para encontrar a ordem r de y módulo N, é possível encontrar x, capaz de solucionar o problema de encontrar x, fazendo $x \leftarrow y^{\frac{r}{2}} \bmod N$.

Com isso é possível afirmar que o problema de fatorar um número inteiro pode ser reduzido, probabilisticamente e em tempo polinomial, ao problema da ordem.

6.1.2 Algoritmo e Complexidade Computacional

Nesta subseção será apresentado o Algoritmo 9, denominado *Fatora*(N), para encontrar um divisor de N, denominado N_1. Para encontrar outros divisores basta realizar *Fatora*(N/N_1). Para encontrar os divisores de N_1 basta realizar *Fatora*(N_1). Caso N seja primo, o algoritmo retornará N. A etapa principal deste algoritmo é *Algoritmo de Shor*(N), descrito no Algoritmo 10. Ainda nessa subseção será realizada uma análise do custo computacional de ambos os algoritmos.

Algoritmo 9 *Fatora*(N):

```
1: if N ≡ 0 (mod 2) then
2:     return 2
3: else if N é primo then
4:     return N
5: else if N é potência perfeita then
6:     Encontra (p, a), tais que p^a = N
7:     return p
8: else
9:     N_1 ← Algoritmo de Shor(N)
10:    return (N_1)
11: end if
```

Algoritmo 10 *Algoritmo de Shor*(N):

1: **while** Verdade **do**
2: Escolha uniformemente ao acaso um valor $y \in \mathbb{Z}_N^\times$
3: **if** $mdc(y, N) \neq 1$ **then**
4: **return** $mdc(y, N)$
5: **else**
6: $r \leftarrow$ ordem de y módulo N
7: **if** $r \equiv 0 \pmod 2$ **and** $y^{\frac{r}{2}} \not\equiv \pm 1 \pmod N$ **then**
8: $x \leftarrow y^{\frac{r}{2}}$
9: **if** $mdc((x+1), N) \neq 1$ **then**
10: **return** $mdc((x+1), N)$
11: **else if** $mdc((x-1), N) \neq 1$ **then**
12: **return** $mdc((x-1), N)$
13: **else**
14: **return** ERRO
15: **end if**
16: **end if**
17: **end if**
18: **end while**

Analisando o Algoritmo 10 é possível notar que o algoritmo sairá do laço, com probabilidade de, pelo menos, $\frac{1}{2}$. Portanto não será levado em conta possíveis repetições das etapas que ocorrem dentro do laço. Dito isso, será apresentado o custo computacional das linhas mais relevantes do Algoritmo 9 na Tabela 6.1 e das linhas mais relevantes do Algoritmo 10 na Tabela 6.2.

Tabela 6.1: Análise do custo computacional das linhas mais importantes do Algoritmo 9.

Linha	Custo computacional em Tempo	Referência
3	Não é um empecilho para o algoritmo	[AKS04]
5	Não é um empecilho para o algoritmo	[Ber98]
9	$O((\log(N))^2 \log\log(N) \log\log\log(N))$	Análise do Algoritmo 10
Total	$O((\log(N))^2 \log\log(N) \log\log\log(N))$	Linha 5

Tabela 6.2: Análise do custo computacional das linhas mais importantes do Algoritmo 10

Linha	Custo computacional em Tempo	Referência
3	$O(\log(N))$	[Mol08]
6	$O((\log(N))^2 \log\log(N) \log\log\log(N))$	Subseção 5.3.3
7	$O((\log(N)) \log\log(N) \log\log\log(N))$	[BCDP96]
9	$O(\log(N))$	[Mol08]
11	$O(\log(N))$	[Mol08]
Total	$O((\log(N))^2 \log\log(N) \log\log\log(N))$	Linha 2

Logo, a complexidade de tempo do algoritmo é polinomial em n, o tamanho da entrada.

6.1.3 Algoritmo de Shor e o protocolo RSA

Como apresentado na Seção 3.2, a dificuldade de decodificar uma mensagem codificada utilizando o protocolo criptográfico RSA se resume na dificuldade de fatorar um número inteiro.

Nesta seção foi possível observar que o algoritmo quântico proposto por Peter Shor é capaz de fatorar um número inteiro utilizando $O((\log(N))^2 \log\log(N) \log\log\log(N))$ operações.

Portanto o algoritmo de Shor realiza um ataque crítico ao protocolo RSA, tornando-o inseguro.

6.1.4 Exemplo

Nesta subseção será demonstrado um exemplo do algoritmo de Shor, fatorando o número 15 utilizando o Algoritmo 9. Para este exemplo suponha que $y = 7$. Este exemplo foi adaptado do livro "Quantum Computation and Quantum Information: 10th Anniversary Edition" [NC11].

Note que 15 é ímpar, não é um número primo, não é uma potência perfeita e que $mdc(15, 7) = 1$. Portanto não pertence ao conjunto de casos "fáceis". Dito isso, o exemplo segue com a análise do algoritmo usado para resolver o problema da ordem para os valores $a = 7$ e $N = 15$.

Inicialmente aplica-se a $\mathcal{QFT}$ sobre o estado $|0\rangle$:

$$\begin{aligned} |0\rangle|1\rangle \xrightarrow{\mathcal{QFT}^{\otimes n}} & \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} |k\rangle|1\rangle \\ & = \frac{1}{\sqrt{2^n}} \left(|0\rangle + |1\rangle + |2\rangle + \cdots + |2^n - 1\rangle\right)|1\rangle \end{aligned} \tag{6.14}$$

Em seguida aplica-se uma sequência de operadores U_a, definidos anteriormente:

$$U_y^k |k\rangle|1\rangle = |k\rangle|y^k \pmod N\rangle \tag{6.15}$$

Deixando o segundo registrador no estado:

$$\begin{aligned} & \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} |k\rangle|y^k \pmod N\rangle \\ & = \frac{1}{\sqrt{2^n}} \Big[|0\rangle|y^0 \pmod N\rangle + |1\rangle|y^1 \pmod N\rangle + \cdots + |2^n - 1\rangle|y^{2^n-1} \pmod N\rangle \Big] \end{aligned} \tag{6.16}$$

Substituindo $N = 15$ e $y = 7$:

$$\begin{aligned}
&\frac{1}{\sqrt{2^n}}\Big[|0\rangle|1 \pmod{15}\rangle + |1\rangle|7 \pmod{15}\rangle + |2\rangle|49 \pmod{15}\rangle + \cdots\Big] \\
&= \frac{1}{\sqrt{2^n}}\Big[|0\rangle|1\rangle + |1\rangle|7\rangle + |2\rangle|4\rangle + |3\rangle|13\rangle + |4\rangle|1\rangle + |5\rangle|7\rangle + |6\rangle|4\rangle + |7\rangle|13\rangle + \cdots\Big] \\
&= \frac{1}{\sqrt{2^n}}\Big[|0\rangle|1\rangle + |1\rangle|7\rangle + |2\rangle|4\rangle + |3\rangle|13\rangle + \\
&\qquad |4\rangle|1\rangle + |5\rangle|7\rangle + |6\rangle|4\rangle + |7\rangle|13\rangle + \cdots\Big]
\end{aligned} \tag{6.17}$$

Neste etapa, o algoritmo para encontrar a ordem aplica a transformação quântica inversa de Fourier ($\mathcal{QFT}^\dagger$) no primeiro registrador para encontrar $|\widetilde{k/r}\rangle$, porém serão realizadas algumas análises antes de aplicar a $\mathcal{QFT}^\dagger$.

Note que o segundo registrador está em uma superposição dos estados $|1\rangle$, $|4\rangle$, $|7\rangle$ ou $|13\rangle$.
O caso geral é $|y^0 \pmod N\rangle$, $|y^1 \pmod N\rangle$, $\cdots$, $|y^{r-1} \pmod N\rangle$.
Então é possível reescrever o estado (6.17) na forma:

$$\begin{aligned}
= \frac{1}{\sqrt{2^n}}\Big[&\big(|0\rangle + |4\rangle + |8\rangle + |12\rangle + \cdots\big)|1\rangle + \\
&\big(|1\rangle + |5\rangle + |9\rangle + |13\rangle + \cdots\big)|7\rangle + \\
&\big(|2\rangle + |6\rangle + |10\rangle + |14\rangle + \cdots\big)|4\rangle + \\
&\big(|3\rangle + |7\rangle + |11\rangle + |15\rangle + \cdots\big)|13\rangle\Big]
\end{aligned} \tag{6.18}$$

Com o propósito de análise, neste exemplo aplica-se uma medição no segundo registrador, antes de aplicar a $\mathcal{QFT}^{\dagger}$ sobre primeiro registrador.
O resultado da medição será $|1\rangle$, $|4\rangle$, $|7\rangle$ ou $|13\rangle$. Suponha que o resultado da medição seja $|4\rangle$. Caso o resultado obtido fosse $|1\rangle$, $|7\rangle$ ou $|13\rangle$ a análise necessitaria de alguns ajustes, porém o resultado final seria equivalente.
Após a medição, o primeiro registrador estará no estado:

$$\sqrt{\frac{4}{2^n}}\Big[|2\rangle + |6\rangle + |10\rangle + |14\rangle + \cdots\Big] \tag{6.19}$$

A seguir é aplicada a $\mathcal{QFT}^{\dagger}$ ao primeiro registrador, deixando-o no estado:

$$\frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1}\sqrt{\frac{4}{2^n}}\Big[e^{-2\pi ik2/2^n} + e^{-2\pi ik6/2^n} + e^{-2\pi ik10/2^n} + e^{-2\pi ik14/2^n} + \cdots e^{-2\pi ik2(2^{n-1}-1)/2^n}\Big] \tag{6.20}$$

O qual é equivalente a:

$$\frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1}\sqrt{\frac{4}{2^n}}\Big[e^{-\pi ik/2^{n-2}} + e^{-\pi ik3/2^{n-2}} + e^{-\pi ik5/2^{n-2}} + e^{-\pi ik7/2^{n-2}} + \cdots e^{-\pi ik(2^{n-1}-1)/2^{n-2}}\Big] \tag{6.21}$$

$$\frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1}\sqrt{\frac{4}{2^n}}\Big[\sum_{j=0}^{2^{n-2}-1} e^{-\pi ik(2j+1)/2^{n-2}}\Big] \tag{6.22}$$

O exemplo se segue analisando para $n = 11$, ou seja $2^n = 2048$ e apenas para os valores de $k \in \{0, 512, 1024, 1536\}$. A seguir deverá ficar claro o porquê de analisar apenas para estes valores de k.

Para $k = 0$:

$$\begin{aligned}
&\frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} e^{\frac{-\pi i 0(2j+1)}{2^9}}\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} e^{0}\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} 1\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[2^9\right] \\
&= \frac{2}{2048}\left[2^9\right] \\
&= \frac{1}{2}
\end{aligned} \tag{6.23}$$

Ou seja, a probabilidade de se obter o estado $|0\rangle$, $k = 0$, é de $\frac{1}{2}^2 = \frac{1}{4}$.

Para $k = 512$:

$$\begin{aligned}
&\frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} e^{\frac{-\pi i 512(2j+1)}{2^9}}\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} e^{-\pi i(2j+1)}\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} -1\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[-2^9\right] \\
&= \frac{2}{2048}\left[-2^9\right] \\
&= -\frac{1}{2}
\end{aligned} \tag{6.24}$$

Ou seja, a probabilidade de se obter o estado $|512\rangle$, $k = 512$, é de $-\frac{1}{2}^2 = \frac{1}{4}$.

Para $k = 1024$:

$$\begin{aligned}
&\frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} e^{\frac{-\pi i 1024(2j+1)}{2^9}}\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} e^{-\pi i 2(2j+1)}\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} 1\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[2^9\right] \\
&= \frac{2}{2048}\left[2^9\right] \\
&= \frac{1}{2}
\end{aligned} \tag{6.25}$$

Ou seja, a probabilidade de se obter o estado $|1024\rangle$, $k = 1024$, é de $\frac{1}{2}^2 = \frac{1}{4}$.

Para $k = 1536$:

$$\begin{aligned}
&\frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} e^{\frac{-\pi i 1536(2j+1)}{2^9}}\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} e^{-\pi i 3(2j+1)}\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[\sum_{j=0}^{2^9-1} -1\right] \\
&= \frac{1}{\sqrt{2048}}\sqrt{\frac{4}{2048}}\left[-2^9\right] \\
&= \frac{2}{2048}\left[-2^9\right] \\
&= -\frac{1}{2}
\end{aligned} \tag{6.26}$$

Ou seja, a probabilidade de se obter o estado $|1536\rangle$, $k = 1536$, é de $-\frac{1}{2}^2 = \frac{1}{4}$.

A soma as probabilidades do estado resultar em $|0\rangle$, $|512\rangle$, $|1024\rangle$ e $|1536\rangle$ é 1. Ou seja, a probabilidade de $|k\rangle$ colapsar em qualquer outro estado é zero.

Suponha que o resultado da medição do primeiro registrador após a aplicação da $\mathcal{QFT}^\dagger$ seja $|512\rangle$. Com isso $512/2048 = 1/4 = 0.25$ será o resultado do algoritmo.

A partir do resultado 0.25 o algoritmo de frações contínuas retorna os valores $k = 1$ e $r = 4$, ou seja a ordem para $y = 7$ é 4. Como 4 é par e $7^{\frac{4}{2}} \not\equiv \pm 1 \pmod{15}$, o algoritmo funcionou. Computando $mdc(7^2 - 1, 15) = 3$ e $mdc(7^2 + 1, 15) = 5$, é possível concluir que $15 = 3 \cdot 5$.

6.2 LOGARITMO DISCRETO

Nesta seção será descrito o algoritmo quântico proposto por Shor, capaz de solucionar o problema do logaritmo discreto em tempo polinomial. Este algoritmo utiliza uma variação do algoritmo proposto para solucionar o problema da ordem.

Esta seção foi baseada na análise presente no livro "An Introduction to Quantum Computing" [KLM07], porém foram realizadas algumas modificações relevantes, adicionando informações que facilitam a compreensão do algoritmo.

A seguir é apresentada uma definição formal do problema do logaritmo discreto.

Problema do Logaritmo Discreto (PLD):
Entrada: Um número primo p. Dois inteiros $a, b \in \{1, \cdots p - 1\}$, tais que $a \equiv b^t \pmod{p}$, para algum $t \in \mathbb{Z}_{>0}$.
Problema: Encontre um inteiro t, tal que $a \equiv b^t \pmod{p}$ e $t \in \{0, 1, 2, \cdots, r - 1\}$, onde r é a ordem de b módulo p.

6.2.1 Redução do Problema

O objetivo desta subseção é reduzir PLD, a um subproblema de PLD, onde a entrada contém adicionalmente um r, primo, tal que r é a ordem de b módulo p.

Problema do Logaritmo Discreto onde r é primo e conhecido (PLDP):
Entrada: Um número primo p. Dois inteiros $a, b \in \{1, \cdots p - 1\}$, tais que $a \equiv b^t \pmod{p}$, para algum $t \in \mathbb{Z}_{>0}$. Um número primo r, onde r é a ordem de b módulo p.
Problema: Encontre um inteiro t, tal que $a \equiv b^t \pmod{p}$ e $t \in \{0, 1, 2, \cdots, r - 1\}$.

Problema do Logaritmo Discreto onde r é primo e conhecido (PLDP):
Entrada: Dois números primos p e r. Dois inteiros $a, b \in \{1, \cdots p - 1\}$, onde r é a ordem de b módulo p e

$$a \equiv b^t \pmod{p}, \quad t \in \mathbb{Z}_{\geq 0}.$$

Problema: Encontre um valor para t, onde $0 \leq t < r$.

Dada uma instância do PLD, note que a ordem de b módulo p pode ser encontrada utilizando o Algoritmo 7. Caso r, a ordem de b módulo p, seja primo a redução está completa. Caso contrario, $r = r_1 r_2$, para $1 < r_1, r_2 < r$. Na análise que segue, será estabelecido que ambos r_1 e r_2 são números primos. Se um destes dois números não for primo, o método pode ser aplicado recursivamente.

Note que é possível encontrar r_1 e r_2 utilizando o Algoritmo 9, para fatoração de números inteiros.

Como $t < r$, existem c_1, c_2, onde $0 \leq c_1 < r_1$ e $0 \leq c_2 < r_2$, tais que:

$$t = c_1 r_2 + c_2. \tag{6.27}$$

Faça:

$$a_1 = a^{r_1} \bmod p.$$

$$b_1 = b^{r_1} \bmod p.$$

Por definição, $a \equiv b^t \pmod{p}$, portanto o seguinte é verdade:

$$\begin{aligned} a &\equiv b^t \pmod{p} \\ (a)^{r_1} &\equiv (b^t)^{r_1} \pmod{p} \\ (a)^{r_1} &\equiv (b^{r_1})^t \pmod{p} \\ a_1 &\equiv b_1^t \pmod{p}. \end{aligned} \tag{6.28}$$

Como $r = r_1 r_2$ e r é a ordem de b módulo p, então $b^{r_1 r_2} \equiv 1 \pmod{p}$, portanto:

$$b_1^{r_2} \equiv 1 \pmod{p}. \tag{6.29}$$

A partir de (6.27), (6.28) e (6.29), tem-se que:

$$\begin{aligned} a_1 &\equiv b_1^t \pmod{p} \\ a_1 &\equiv b_1^{c_1 r_2 + c_2} \pmod{p} \\ a_1 &\equiv b_1^{c_1 r_2} b_1^{c_2} \pmod{p} \\ a_1 &\equiv (b_1^{r_2})^{c_1} b_1^{c_2} \pmod{p} \\ a_1 &\equiv 1 b_1^{c_2} \pmod{p} \\ a_1 &\equiv b_1^{c_2} \pmod{p}. \end{aligned} \tag{6.30}$$

Note encontrar c_2 tal que $a_1 \equiv b_1^{c_2} \pmod{p}$, onde o primo r_2 é a ordem de b_1 módulo p é uma instância do PLDP. A seguir será visto como c_2 pode ser usado na solução do PLD original.

Defina b_2 da seguinte maneira:

$$b_2 = b^{r_2} \bmod p. \tag{6.31}$$

Como, por definição $a \equiv b^t \pmod{p}$ então

$$\begin{aligned} ab^{-c_2} &\equiv b^t b^{-c_2} \pmod{p} \\ &\equiv b^{c_1 r_2 + c_2} b^{-c_2} \pmod{p} \\ &\equiv b^{c_1 r_2} \pmod{p} \\ &\equiv (b^{r_2})^{c_1} \pmod{p} \\ &\equiv (b_2)^{c_1} \pmod{p}. \end{aligned} \tag{6.32}$$

Fazendo $a_2 = ab^{-c_2} \bmod p$

$$a_2 \equiv b_2^{c_2} \pmod{p}. \tag{6.33}$$

Note encontrar c_1 tal que $a_2 \equiv b_2^{c_1} \pmod{p}$, onde o primo r_1 é a ordem de b_2 módulo p é uma instância do PLDP.

Finalmente, a partir de c_1, c_2 e r_2 pode se encontrar t, uma vez que $t = c_1 r_2 + c_2$. Portanto o problema do logaritmo discreto (PLD) pode ser reduzido ao $PLDP$.

6.2.2 Intuição do Algoritmo

Sejam os seguintes operadores de multiplicação modular, os quais são utilizados no algoritmo:

$$U_a : |s\rangle \rightarrow |sa \bmod p\rangle, \quad s < p. \tag{6.34}$$

$$U_b : |s\rangle \rightarrow |sb \bmod p\rangle, \quad s < p. \tag{6.35}$$

Considere o estado quântico (6.36) a seguir, o qual será fundamental para a compreensão do algoritmo:

$$|u_k\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |b^s \bmod p\rangle. \tag{6.36}$$

Inicialmente será demonstrado que $|u_k\rangle$ é autovetor de U_a e U_b, com autovalores correspondentes $e^{2\pi i w}$ e $e^{2\pi i w'}$, onde $w = \frac{kt}{r}$ e $w' = \frac{k}{r}$. Em seguida é utilizado o Lema 5.3.2 para evidenciar que $\sum_{k=0}^{r-1} |u_k\rangle = |1\rangle$. Finalmente, com um pouco de teoria dos números, é obtido t a partir das estimativas para w e w'.

A seguir será analisada a aplicação da porta U_a ao estado $|u_k\rangle$, demonstrando que $|u_k\rangle$ é um autovetor de U_a com autovalor de $e^{2\pi i \frac{kt}{r}}$:

$$\begin{aligned}
U_a|u_k\rangle &= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} U_a |b^s \bmod p\rangle \\
&= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |b^s a \bmod p\rangle \\
&= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |b^s b^t \bmod p\rangle \\
&= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |b^{s+t} \bmod p\rangle \\
&= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} (e^{2\pi i \frac{k}{r} t})(e^{-2\pi i \frac{k}{r} t})(e^{-2\pi i \frac{k}{r} s}) |b^{s+t} \bmod p\rangle \\
&= e^{2\pi i \frac{k}{r} t} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} (s+t)} |b^{s+t} \bmod p\rangle.
\end{aligned} \tag{6.37}$$

Como demonstrado no Lema 5.3.1:

$$\sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} (s+t)} |b^{s+t} \bmod p\rangle = \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} (s)} |b^s \bmod p\rangle. \tag{6.38}$$

Substituindo em (6.37):

$$\begin{aligned} U_a|u_k\rangle &= e^{2\pi i \frac{k}{r} t} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+t)} |b^{s+t} \bmod p\rangle \\ &= e^{2\pi i \frac{k}{r} t} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |b^{s} \bmod p\rangle \\ &= e^{2\pi i \frac{k}{r} t} |u_k\rangle. \end{aligned} \tag{6.39}$$

Portanto $|u_k\rangle$ é um autovetor de U_a com autovalor de $e^{2\pi i \frac{kt}{r}}$.

A seguir será analisada a aplicação da porta U_b ao estado $|u_k\rangle$, demonstrando que $|u_k\rangle$ é um autovetor de U_b com autovalor de $e^{2\pi i \frac{k}{r}}$:

$$\begin{aligned} U_b|u_k\rangle &= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} U_b |b^{s} \bmod p\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |b^{s} b \bmod p\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |b^{s+1} \bmod p\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} (e^{2\pi i \frac{k}{r}})(e^{-2\pi i \frac{k}{r}})(e^{-2\pi i \frac{k}{r} s}) |b^{s+1} \bmod p\rangle \\ &= e^{2\pi i \frac{k}{r}} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+1)} |b^{s+1} \bmod p\rangle. \end{aligned} \tag{6.40}$$

Como demonstrado no Lema 5.3.1:

$$\sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+1)} |b^{s+1} \bmod p\rangle = \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |b^{s} \bmod p\rangle. \tag{6.41}$$

Substituindo em (6.40):

$$\begin{aligned} U_b|u_k\rangle &= e^{2\pi i \frac{k}{r}} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s+1)} |b^{s+1} \bmod p\rangle \\ &= e^{2\pi i \frac{k}{r}} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r}(s)} |b^{s} \bmod p\rangle \\ &= e^{2\pi i \frac{k}{r}} |u_k\rangle. \end{aligned} \tag{6.42}$$

Portanto $|u_k\rangle$ é um autovetor de U_b com autovalor de $e^{2\pi i \frac{k}{r}}$.

A principal ideia do algoritmo para resolver PLDP é utilizar o algoritmo quântico para estimação de fase sobre o estado $|u_k\rangle$, utilizando as portas U_a e U_b para determinar os valores $\frac{kt \bmod r}{r}$ e $\frac{k}{r}$.

A partir do Lema 5.3.2 conclui-se que:

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

$$\begin{aligned} |0\rangle|1\rangle &= |0\rangle\left(\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |u_k\rangle\right) \\ &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |0\rangle|u_k\rangle. \end{aligned} \tag{6.43}$$

Seja m a quantidade de qubits suficiente para a aplicação dos operadores U_a e U_b, e n um valor arbitrário, considere o seguinte estado:

$$|0\rangle^n|1\rangle^m \tag{6.44}$$

Construindo o circuito para a estimação de fase utilizando o operador U_a e aplicando no estado $|0\rangle^n|1\rangle^m$, será obtida a superposição de estados (6.45). Isto ocorre pois $e^{2\pi i \frac{kt}{r}}$ é o autovalor de $|u_k\rangle$ em relação à U_a, ou seja, o valor w estimado pelo Algoritmo 5 é $\frac{kt \bmod r}{r}$.

$$\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\widetilde{\frac{kt \bmod r}{r}}\rangle|u_k\rangle \tag{6.45}$$

Ao efetuar a medição no primeiro qubit é obtido um valor y, tal que $\frac{y}{2^n} = \widetilde{\frac{kt \bmod r}{r}}$, onde $k, t \in \{0, 1, \cdots, r-1\}$.

Construindo o circuito para a estimação de fase utilizando o operador U_b e aplicando no estado $|0\rangle^n|1\rangle^m$, será obtida a superposição de estados (6.46). Isto ocorre pois $e^{2\pi i \frac{k}{r}}$ é o autovalor de $|u_k\rangle$ em relação à U_b, ou seja, o valor w estimado pelo Algoritmo 5 é $\frac{k}{r}$.

$$\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\widetilde{k/r}\rangle|u_k\rangle. \tag{6.46}$$

Ao efetuar a medição no primeiro qubit é obtido um valor x, tal que $\frac{x}{2^n} = \widetilde{\frac{k}{r}}$, onde $k \in \{0, 1, \cdots, r-1\}$.

É importante mencionar que, ao se fazer a medida, k pode ser qualquer valor pertencente ao intervalo $\{0, 1, \cdots, r-1\}$. No entanto é desejado que as saídas $\widetilde{\frac{k}{r}}$ e $\widetilde{\frac{kt \bmod r}{r}}$, compartilhem um mesmo k, o que o algoritmo de estimação de fase não garante. Contudo os ajustes necessários serão realizados na Subseção 6.2.3.

Pelo Lema A.2.2, a probabilidade dos valores x e y satisfazerem simultaneamente (6.47) e (6.48), é de $\left(\frac{8}{\pi^2}\right)^2$.

$$\left|\frac{x}{2^n} - \frac{k}{r}\right| \leq \frac{1}{2^n}. \tag{6.47}$$

$$\left|\frac{y}{2^n} - \frac{kt \bmod r}{r}\right| \leq \frac{1}{2^n}. \tag{6.48}$$

Reescrevendo (6.47) e (6.48):

$$\left|\frac{xr}{2^n} - k\right| \leq \frac{r}{2^n};$$

$$\left|\frac{yr}{2^n} - kt \bmod r\right| \leq \frac{r}{2^n};$$

Escolhendo n de forma que $2r < 2^n$, então:

$$\left|\frac{xr}{2^n} - k\right| \leq \frac{r}{2^n} < \frac{r}{2r} = \frac{1}{2}.$$

$$\left|\frac{yr}{2^n} - kt \bmod r\right| \leq \frac{r}{2^n} < \frac{r}{2r} = \frac{1}{2}.$$

Ou seja:

$$\left|\frac{xr}{2^n} - k\right| < \frac{1}{2}.$$

$$\left|\frac{yr}{2^n} - kt \bmod r\right| < \frac{1}{2}.$$

Portanto arredondando $\frac{xr}{2^n}$ para o inteiro mais próximo se obtém k. Similarmente, arredondando $\frac{yr}{2^n}$ para o inteiro mais próximo, se obtém $kt \bmod r$. Então, dado que r é conhecido e x e y são valores computados, tem-se que:

$$k = round\left(\frac{xr}{2^n}\right).$$

$$kt \bmod r = round\left(\frac{yr}{2^n}\right).$$

Para $k \in \{1, 2, 3, \cdots, r-1\}$ e dado que r é primo, $mdc(k, r) = 1$, então, pelo Lema de Bézout B.2.7, k possui um inverso módulo r. Ou seja, $\exists k'$ tal que $kk' \equiv 1 \pmod{r}$.

Portanto:

$$t \equiv k'kt \pmod{r}.$$

Dado que r é primo, o Pequeno Teorema de Fermat B.2.2 afirma que:

$$k^{r-1} \equiv 1 \pmod{r}.$$

Portanto:

$$k^{r-2}k \equiv 1 \pmod{r}.$$

Ou seja $k' = k^{r-2}$ e com isso é possível obter t.

6.2.3 Construindo o Circuito

Como demonstrado na subseção anterior, o algoritmo de estimação de fase pode ser usado para encontrar t, no entanto serão realizadas algumas mudanças.

Inicia-se o circuito com o estado $|0\rangle^n|0\rangle^n|1\rangle^m$.

Os n primeiros qubits serão chamados de qubits Q_a e os próximos n qubits são qubits como qubits Q_b, enquanto os qubits do estado $|1\rangle^m$ serão chamados de qubits alvo.

Aplica-se a $\mathcal{QFT}$ sobre os qubits Q_a e também sobre os qubits Q_b, como representado na Figura 6.1. Assim como no algoritmo de estimação de fase, o objetivo desta etapa é deixá-los em superposições de iguais probabilidades.

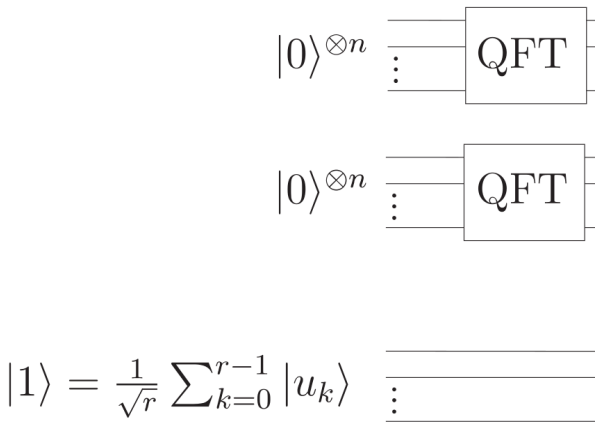


Figura 6.1: Primeira etapa da construção do circuito quântico que soluciona o problema do logaritmo discreto [KLM07].

Com isso, o estado obtido é a seguinte superposição:

$$\begin{aligned} &\frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} |j\rangle \frac{1}{\sqrt{2^n}} \sum_{l=0}^{2^n-1} |l\rangle|1\rangle \\ &= \frac{1}{\sqrt{2^{2n}}} \sum_{j=0}^{2^n-1} \sum_{l=0}^{2^n-1} |j\rangle|l\rangle|1\rangle \\ &= \frac{1}{\sqrt{r2^{2n}}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} \sum_{l=0}^{2^n-1} |j\rangle|l\rangle|u_k\rangle. \end{aligned} \tag{6.49}$$

Os qubits Q_a serão utilizados como os qubits controle dos operadores U_a e os qubits Q_b serão utilizados como qubits controle dos operadores U_b. Ambos os operadores U_a e U_b irão utilizar os qubits de $|1\rangle^m$ como qubits alvo.

Em seguida é aplicado o operador U_b^x que é a seguinte sequência de operadores:

$$U_b^x = U_b^{2^{n-1}} U_b^{2^{n-2}} \cdots U_b^2 U_b. \tag{6.50}$$

Onde cada operador $U_b^{2^{n-k}}, k \in \{1, \cdots, n\}$ utiliza o k-ésimo qubit dos qubits Q_b como qubit de controle.

Aplica-se também uma sequência U_a^x que é a seguinte sequência de operadores:

$$U_a^x = U_a^{2^{n-1}} U_a^{2^{n-2}} \cdots U_a^2 U_a. \tag{6.51}$$

Onde cada operador $U_a^{2^{n-k}}, k \in \{1, \cdots, n\}$ utiliza o k-ésimo qubit dos qubits Q_a como qubit de controle.

A Figura 6.2 representa a aplicações das sequências de operadores U_b^x e U_a^x.

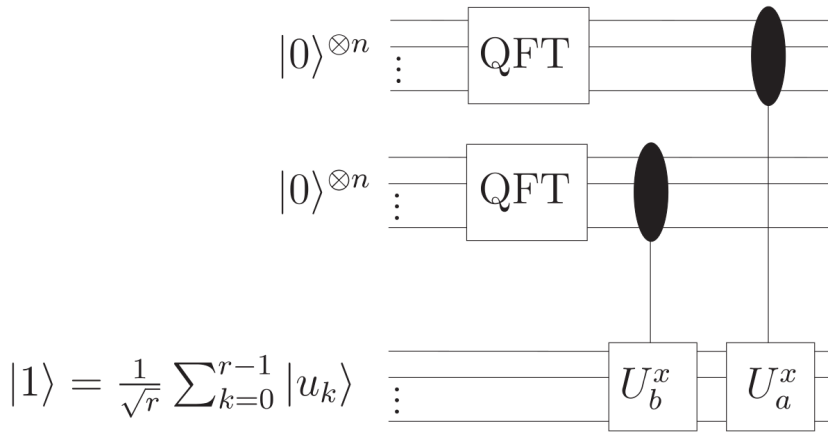


Figura 6.2: Segunda etapa da construção do circuito quântico que soluciona o problema do logaritmo discreto [KLM07].

Resultando no estado:

$$\frac{1}{\sqrt{r2^{2n}}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} \sum_{l=0}^{2^n-1} e^{2\pi i \frac{kt}{r} j} |j\rangle e^{2\pi i \frac{k}{r} l} |l\rangle |u_k\rangle. \tag{6.52}$$

Após a aplicação das sequências de operadores U_a e U_b aplica-se a $\mathcal{QFT}^\dagger$ sobre os qubits Q_a e qubits Q_b, como demonstrado na Figura 6.3.

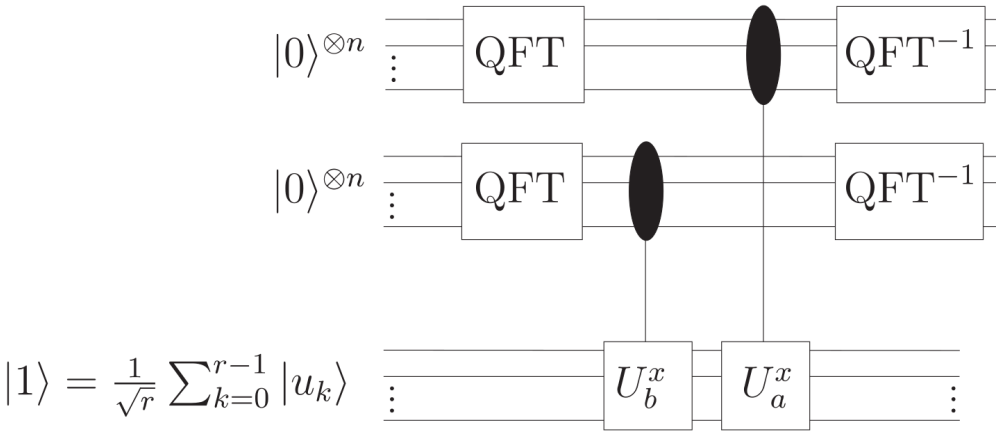


Figura 6.3: Terceira etapa da construção do circuito quântico que soluciona o problema do logaritmo discreto [KLM07].

Resultando no estado:

$$\sum_{k=0}^{r-1} |\widetilde{\frac{kt \bmod r}{r}}\rangle |\widetilde{\frac{k}{r}}\rangle |u_k\rangle. \tag{6.53}$$

Com isso, aplicando uma medição nos qubits Q_a e qubits Q_b, a superposição colapsará em $|\widetilde{\frac{kt \bmod r}{r}}\rangle |\widetilde{\frac{k}{r}}\rangle |u_k\rangle$, onde $k, t \in \{0, 1, \cdots, r-1\}$. Caso $k = 0$ o algoritmo retornará falha. Portanto, com probabilidade $\frac{r-1}{r}$, $k \in \{1, \cdots, r-1\}$.

A Figura 6.4 demonstra o circuito completo.

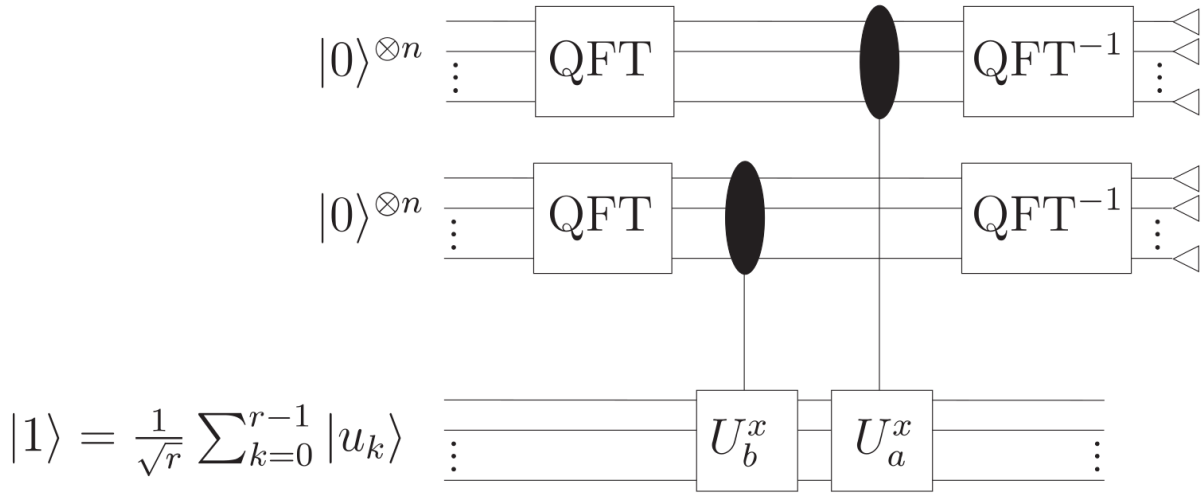


Figura 6.4: Circuito quântico que soluciona o problema do logaritmo discreto [KLM07].

6.2.4 Algoritmo e Complexidade Computacional

Nesta subseção será apresentado o algoritmo *Logaritmo Discreto*(a, b, p, r, U_a, U_b), capaz de solucionar o PLDP. Assim como uma análise de seu custo computacional.

Inicialmente note que $2r < 2^n$ e, pelo Teorema B.2.10, $r < p$. Portanto será feito $n = \lceil \log(2p) + 1 \rceil$.

Algoritmo 11 *Logaritmo Discreto*(a, b, p, r, U_a, U_b):

1: $n \leftarrow \lceil \log(2p) + 1 \rceil$.
2: Inicia um registrador no estado $|0\rangle^n|0\rangle^n|1\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |0\rangle^n|0\rangle^n|u_k\rangle$.
3: Aplica a $\mathcal{QFT}$ sobre os n primeiros qubits, $|0\rangle^n$, transformando o estado $|0\rangle^n$ no estado $\frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} |j\rangle$.
4: Aplica a $\mathcal{QFT}$ sobre os n qubits, $|0\rangle^n$, transformando o estado $|0\rangle^n$ no estado $\frac{1}{\sqrt{2^n}} \sum_{l=0}^{2^n-1} |l\rangle$.
5: Obtendo o estado: $\frac{1}{\sqrt{r2^{2n}}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} \sum_{l=0}^{2^n-1} |j\rangle|l\rangle|u_k\rangle$
6: Aplica a sequência de operadores de controle U_b^x, transformando o estado $\frac{1}{\sqrt{r2^n}} \sum_{k=0}^{r-1} \sum_{l=0}^{2^n-1} |l\rangle|u_k\rangle$ no estado $\frac{1}{\sqrt{r2^n}} \sum_{k=0}^{r-1} \sum_{l=0}^{2^n-1} e^{2\pi i \frac{k}{r} l}|l\rangle|u_k\rangle$.
7: Aplica a sequência de operadores de controle U_a^x, transformando o estado $\frac{1}{\sqrt{r2^n}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} |j\rangle|u_k\rangle$ no estado $\frac{1}{\sqrt{r2^n}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} e^{2\pi i \frac{kt}{r} j}|j\rangle|u_k\rangle$.
8: Obtendo o estado: $\frac{1}{\sqrt{r2^{2n}}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} \sum_{l=0}^{2^n-1} e^{2\pi i \frac{kt}{r} j}|j\rangle e^{2\pi i \frac{k}{r} l}|l\rangle|u_k\rangle$.
9: Aplica a $\mathcal{QFT}^\dagger$ sobre o estado $\frac{1}{\sqrt{r2^n}} \sum_{k=0}^{r-1} \sum_{l=0}^{2^n-1} e^{2\pi i \frac{k}{r} l}|l\rangle$, obtendo o estado $\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\widetilde{\frac{k}{r}}\rangle$.
10: Aplica a $\mathcal{QFT}^\dagger$ sobre o estado $\frac{1}{\sqrt{r2^n}} \sum_{k=0}^{r-1} \sum_{j=0}^{2^n-1} e^{2\pi i \frac{kt}{r} j}|j\rangle$, obtendo o estado $\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\widetilde{\frac{kt \bmod r}{r}}\rangle$.
11: Obtendo o estado $\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\widetilde{\frac{kt \bmod r}{r}}\rangle|\widetilde{\frac{k}{r}}\rangle|u_k\rangle$.
12: Realiza duas medições no estado $|\widetilde{\frac{kt \bmod r}{r}}\rangle|\widetilde{\frac{k}{r}}\rangle$, obtendo dois valores x e y, onde, para algum $k \in \{0, 1, \cdots, r-1\}$, $\frac{x}{2^n} = \widetilde{\frac{k}{r}}$ e $\frac{y}{2^n} = \widetilde{\frac{kt \bmod r}{r}}$.
13: $\hat{x} \leftarrow round\left(\frac{xr}{2^n}\right)$.
14: $\hat{y} \leftarrow round\left(\frac{yr}{2^n}\right)$.
15: **if** $\hat{x} = 0$ **or** $\hat{y} = 0$ **then**
16: **return** ERRO
17: **end if**
18: $\hat{x}^{-1} \leftarrow \hat{x}^{r-2} \bmod p$.
19: $\widetilde{t} \leftarrow \hat{x}^{-1}\hat{y} \bmod p$.
20: **if** $a \not\equiv b^{\widetilde{t}} \pmod p$ **then**
21: **return** ERRO
22: **end if**
23: **return** $\widetilde{t}$.

Dado que o custo computacional para realizar operações de multiplicação modular, $a^j \bmod p$ e $b^j \bmod p$, é $O(\log(p) \log\log(p) \log\log\log(p))$ [BCDP96]. O custo computacional de implementar o operador U_a^x, assim como o custo computacional de implementar o operador U_b^x, é equivalente ao custo de implementar n operações de multiplicação modular [KLM07], ou seja:

$$O(n \log(p) \log\log(p) \log\log\log(p)) \tag{6.54}$$

Dado que $N = 2^n$, e utilizando a linha 1 do Algoritmo 11:

$$n = \lceil \log(2p) + 1 \rceil \therefore O(\log(N))) = O(\log(p))) \tag{6.55}$$

Portanto o custo computacional de implementar os operadores U_a^x e U_b^x é:

$$O((\log(p))^2 \log\log(p) \log\log\log(p)) \tag{6.56}$$

A Tabela 6.3 a seguir, contém uma análise do custo computacional das linhas mais relevantes do Algoritmo 11.

Tabela 6.3: Análise do custo computacional das linhas mais importantes do Algoritmo 11

Linha	Custo computacional em Tempo	Referência
3	$O((\log(p))^2)$	Anánálise da QFT e (6.55)
4	$O((\log(p))^2)$	Anánálise da QFT e (6.55)
6	$O((\log(p))^2 \log\log(p) \log\log\log(p))$	(6.56)
7	$O((\log(p))^2 \log\log(p) \log\log\log(p))$	(6.56)
9	$O((\log(p))^2)$	Anánálise da QFT e (6.55)
10	$O((\log(p))^2)$	Anánálise da QFT e (6.55)
18	$O(\log(p) \log\log(p) \log\log\log(p))$	[BCDP96]
19	$O(\log(p) \log\log(p) \log\log\log(p))$	[BCDP96]
Total	$O((\log(p))^2 \log\log(p) \log\log\log(p))$	Linha 6

Analisando o Algoritmo 11, pode ser afirmado que a sua probabilidade de sucesso é:

$$\frac{8}{\pi^2}\frac{8}{\pi^2}\frac{r-1}{r} = \frac{r-1}{r}0.657\ldots.$$

6.2.5 Algoritmo de Shor e a troca de chaves de Diffie-Hellman-Merkle

Como apresentado na seção troca de chaves de Diffie-Hellman-Merkle, a dificuldade de decodificar este protocolo se resume na dificuldade de resolver o problema do logaritmo discreto.

Nesta seção foi possível observar que o algoritmo quântico proposto por Peter Shor soluciona o problema do logaritmo discreto, utilizando $O((\log(p))^2 \log\log(p) \log\log\log(p))$ operações. Portanto o algoritmo de Shor realiza um ataque crítico ao modelo de troca de chaves de Diffie-Hellman-Merkle, tornando-o inseguro.

Adicionalmente, conforme demonstrado em [PZ03], o algoritmo apresentado nesta seção também pode ser utilizado para fazer ataques a sistemas que usam protocolos criptográficos baseados em curvas elípticas, pois tais protocolos estão relacionados ao problema do logaritmo discreto.

7 ALGORITMO DE GROVER

Neste capítulo será discutido o algoritmo de Grover para busca não estruturada no modelo caixa preta, publicado em 1996 [Gro96]. Tal algoritmo resulta em uma melhoria polinomial em comparação com algoritmos projetados para computadores clássicos, possuindo custo computacional de $O(\sqrt{N})$, onde N é o número de possíveis entradas, em contrapartida à $O(N)$, custo computacional de algoritmo projetados para computadores clássicos.

Em 1997 foi provado por Bennett, Bernstein, Brassard e Vazirani que o algoritmo de Grover é o melhor algoritmo possível para busca baseada em oráculos [BBBV97].

A análise proposta neste capítulo foi inspirada nas análises dos livros "Quantum Computation and Quantum Information: 10th Anniversary Edition" [NC11] e "An Introduction to Quantum Computing" [KLM07] sobre o algoritmo de Grover. Onde no primeiro, o algoritmo de Grover é analisado para o problema de busca não estruturada de múltiplos elementos, resultando no custo custo computacional de $O(\sqrt{\frac{N}{M}})$, onde M é a quantidade de elementos que satisfazem uma determinada função e N é o número de possíveis entradas.

7.1 ALGORITMO DE GROVER PARA BUSCA NÃO ESTRUTURADA

Nesta seção será apresentado o conceito do algoritmo de Grover para busca não estruturada, assim como uma prova de que seu custo computacional está em $O(\sqrt{N})$.

A seguir é apresentada a definição formal do problema de busca não estruturada de único elemento.

Busca Não Estruturada de Único Elemento:
Entrada: Um operador U_f que implementa uma função $f : \{0,1\}^n \Rightarrow \{0,1\}$.
Problema: Encontrar o único valor $w \in \{0,1\}^n$, tal que $f(w) = 1$.

7.1.1 Definição dos operadores utilizados

7.1.1.1 Operador U_f

Utilizando a função $f(x)$ é construído o operador de controle U_f, o qual atua sobre um estado de controle de n qubits, e um estado alvo de um único qubit:

$$U_f|x\rangle|b\rangle = |x\rangle|b \otimes f(x)\rangle.$$

$$U_f|x\rangle|0\rangle = |x\rangle|f(x)\rangle.$$

Definindo o qubit alvo como $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$, observe que ocorre o fenômeno de *Phase-Kickback*:

$$\begin{aligned}
&\text{Para } f(x) = 0: \quad U_f|x\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = |x\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = (1)|x\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right);\\
&\text{Para } f(x) = 1: \quad U_f|x\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = |x\rangle\left(\frac{|1\rangle - |0\rangle}{\sqrt{2}}\right) = (-1)|x\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).
\end{aligned} \tag{7.1}$$

Portanto:

$$U_f|x\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = (-1)^{f(x)}|x\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).$$

Analisando apenas o primeiro estado da equação anterior:

$$U_f|x\rangle = (-1)^{f(x)}|x\rangle.$$

7.1.1.2 Operador de Hadamard

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

$$H^{\otimes n} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1\cdot[H^{\otimes n-1}] & 1\cdot[H^{\otimes n-1}] \\ 1\cdot[H^{\otimes n-1}] & -1\cdot[H^{\otimes n-1}] \end{bmatrix}.$$

Observando o operador o operador $H^{\otimes n}$, é possível concluir que:

$$\begin{aligned} H_{i,0}^{\otimes n} &= \frac{1}{\sqrt{2^n}}, \quad \forall i \in \{1, \cdots, n\}; \\ H_{0,i}^{\otimes n} &= \frac{1}{\sqrt{2^n}}, \quad \forall i \in \{1, \cdots, n\}. \end{aligned} \tag{7.2}$$

Isto será utilizado na construção de (7.10) e (7.11).

7.1.1.3 Operador $U_{0^\perp}$

$$\begin{aligned} U_{0^\perp}|x\rangle &= -|x\rangle, \quad x \in \mathbb{Z}_{>0} \\ U_{0^\perp}|0\rangle &= |0\rangle. \end{aligned} \tag{7.3}$$

$$U_{0^\perp} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & -1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1 \end{bmatrix}.$$

7.1.1.4 Operador $U_{\psi^\perp}$

Dado um estado qualquer $|\phi\rangle = \sum_{x=0}^{2^n} \alpha_x|x\rangle$, este operador tem como propósito inverter os coeficientes α_i deste estado, onde $i \in \{0, 1, \cdots 2^n\}$, sobre a média destes coeficientes.

Ou seja, dada a média dos coeficientes do estado $|\phi\rangle$:

$$\mu = \frac{1}{2^n}\sum_{x=0}^{2^n} \alpha_x.$$

Este operador realiza a seguinte transformação:

$$U_{\psi^\perp}|\phi\rangle = \sum_{x=0}^{2^n} (2\mu - \alpha_x)|x\rangle.$$

Este operador pode ser construído utilizando o operador de Hadamard e o operador $U_{0^\perp}$. Como descrito a seguir:

$$U_{\psi^\perp} = H^{\otimes n} U_{0^\perp} H^{\otimes n}. \tag{7.4}$$

$$U_{\psi^\perp} = H^{\otimes n} \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 0 & \cdots & 0 \\ 0 & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -1 \end{bmatrix} H^{\otimes n}. \tag{7.5}$$

$$U_{\psi^\perp} = H^{\otimes n} \cdot \left(\begin{bmatrix} 2 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \right) \cdot H^{\otimes n}. \tag{7.6}$$

$$U_{\psi^\perp} = H^{\otimes n} \cdot \left(\begin{bmatrix} 2 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} - I \right) \cdot H^{\otimes n}. \tag{7.7}$$

$$U_{\psi^\perp} = H^{\otimes n} \begin{bmatrix} 2 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} H^{\otimes n} - H^{\otimes n} I H^{\otimes n}. \tag{7.8}$$

$$U_{\psi^\perp} = H^{\otimes n} \begin{bmatrix} 2 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} H^{\otimes n} - I. \tag{7.9}$$

$$U_{\psi^\perp} = \begin{bmatrix} \frac{2}{\sqrt{2^n}} & 0 & 0 & \cdots & 0 \\ \frac{2}{\sqrt{2^n}} & 0 & 0 & \cdots & 0 \\ \frac{2}{\sqrt{2^n}} & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{2}{\sqrt{2^n}} & 0 & 0 & \cdots & 0 \end{bmatrix} H^{\otimes n} - I. \tag{7.10}$$

$$U_{\psi^\perp} = \begin{bmatrix} \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \end{bmatrix} - I. \tag{7.11}$$

$$U_{\psi^\perp} = \begin{bmatrix} \frac{2}{2^n} - 1 & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} - 1 & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} - 1 & \cdots & \frac{2}{2^n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} - 1 \end{bmatrix}. \tag{7.12}$$

Analisando a aplicação do operador $U_{\psi^\perp}$ sobre o estado arbitrário $|\phi\rangle$:

$$U_{\psi^\perp}|\phi\rangle = \begin{bmatrix} \frac{2}{2^n} - 1 & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} - 1 & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} - 1 & \cdots & \frac{2}{2^n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} - 1 \end{bmatrix} |\phi\rangle. \tag{7.13}$$

$$U_{\psi^\perp}|\phi\rangle = \begin{bmatrix} \frac{2}{2^n} - 1 & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} - 1 & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} - 1 & \cdots & \frac{2}{2^n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{2}{2^n} & \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} - 1 \end{bmatrix} \cdot \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_x \\ \vdots \\ \alpha_{2^n} \end{bmatrix}. \tag{7.14}$$

$$U_{\psi^\perp}|\phi\rangle = \begin{bmatrix} \sum_{i=0}^{2^n} \frac{2}{2^n}\alpha_i - \alpha_1 \\ \vdots \\ \sum_{i=0}^{2^n} \frac{2}{2^n}\alpha_i - \alpha_x \\ \vdots \\ \sum_{i=0}^{2^n} \frac{2}{2^n}\alpha_i - \alpha_{2^n} \end{bmatrix}. \tag{7.15}$$

$$U_{\psi^\perp}|\phi\rangle = \begin{bmatrix} 2\mu - \alpha_1 \\ \vdots \\ 2\mu - \alpha_x \\ \vdots \\ 2\mu - \alpha_{2^n} \end{bmatrix}. \tag{7.16}$$

$$U_{\psi^\perp}|\phi\rangle = \sum_{x=0}^{2^n} (2\mu - \alpha_x)|x\rangle. \tag{7.17}$$

7.1.2 Intuição do algoritmo

Este algoritmo pode ser resumido em uma sequência aplicações dos operadores U_f e $U_{\psi^\perp}$. Esta sequência de operadores tem como objetivo fazer com que a probabilidade de se

obter o estado $|w\rangle$, após uma medição, seja a mais alta possível, como será visto ao longo deste capítulo.

O estado de controle é iniciado no estado $|0\rangle^n$. Em seguida é aplicada a porta de Hadamard $H^{\otimes n}$, deixando-o no estado $|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$.

A Figura 7.1 representa os coeficientes do estado $|\psi\rangle$.

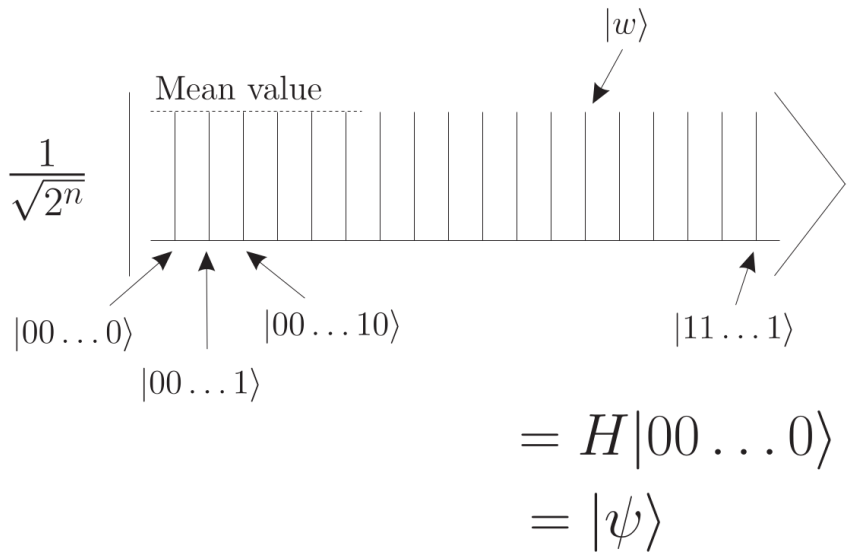


Figura 7.1: Superposição inicial do estado de controle no algoritmo de Grover [KLM07].

O qubit alvo é iniciado no estado $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$. Portanto a entrada do algoritmo de Grover é:

$$|\psi\rangle \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

Seja $|w\rangle$ o estado que satisfaz a seguinte equação:

$$U_f|w\rangle|0\rangle = |w\rangle|1\rangle.$$

A principal ideia do algoritmo é aumentar o coeficiente do estado $|w\rangle$, para que, ao efetuar uma medição, a saída do circuito seja o próprio estado $|w\rangle$.

Para isso, o operador U_f é aplicado ao estado $|\psi\rangle \frac{|0\rangle-|1\rangle}{\sqrt{2}}$, invertendo o sinal do coeficiente do estado $|w\rangle$ e reduzindo o valor da média dos coeficientes do estado de controle. O qubit alvo será omitido das equações, em vista que ele não será alterado.

$$U_f|\psi\rangle = -\frac{1}{\sqrt{2^n}}|w\rangle + \sum_{x=0, x\neq w}^{2^n-1} \frac{1}{\sqrt{2^n}}|x\rangle.$$

$$\begin{aligned}\mu &= \frac{1}{2^n}\Big(-\frac{1}{\sqrt{2^n}} + \sum_{x=0,x\neq w}^{2^n-1}\frac{1}{\sqrt{2^n}}\Big)\\ &= \frac{1}{2^n}\Big(-\frac{1}{\sqrt{2^n}} + \frac{2^n-1}{\sqrt{2^n}}\Big)\\ &= \frac{1}{2^n}\Big(\frac{2^n-2}{\sqrt{2^n}}\Big)\\ &= \frac{2^n-2}{\sqrt{2^n}2^n} < \frac{1}{\sqrt{2^n}}.\end{aligned} \tag{7.18}$$

A Figura 7.2 representa os coeficientes do estado de controle, após a aplicação do operador U_f.

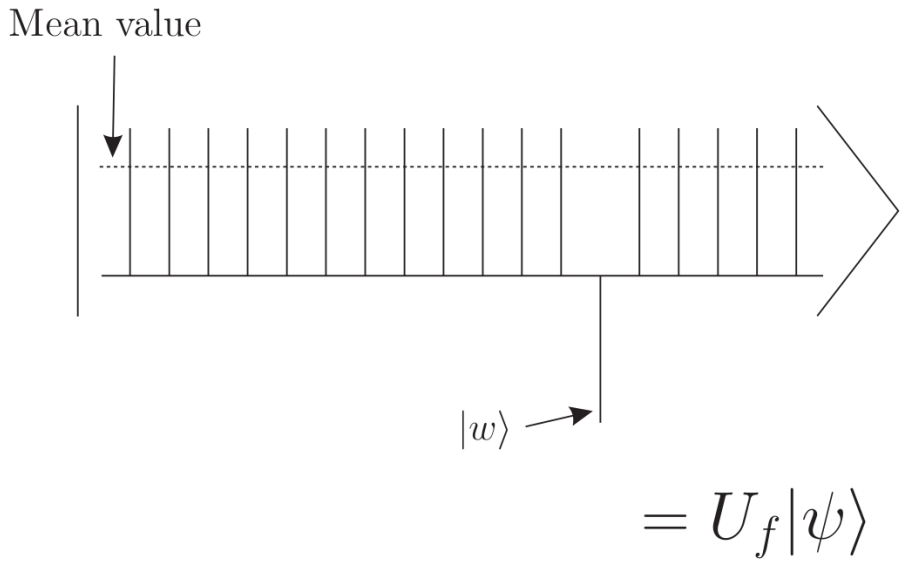


Figura 7.2: Superposição do estado de controle no algoritmo de Grover, após a primeira aplicação a porta U_f [KLM07].

Em seguida é aplicado o operador $U_{\psi^\perp}$, com o intuito de inverter o valor de cada coeficiente sobre a média dos coeficientes, deixando o coeficiente do estado $|w\rangle$ maior que o

coeficiente de outros estados. Novamente será omitido o qubit alvo das equações, em vista que ele não será alterado.

$$
\begin{aligned}
U_{\psi^\perp}U_f|\psi\rangle &= \big(2\mu - \big(-\frac{1}{\sqrt{2^n}}\big)\big)|w\rangle + \sum_{x=0,x\neq w}^{2^n-1}\big(2\mu - \frac{1}{\sqrt{2^n}}\big)|x\rangle \\
&= \big(2\mu + \frac{1}{\sqrt{2^n}}\big)|w\rangle + \sum_{x=0,x\neq w}^{2^n-1}\big(2\mu - \frac{1}{\sqrt{2^n}}\big)|x\rangle \\
&= \big(2\frac{2^n-2}{\sqrt{2^n}2^n} + \frac{1}{\sqrt{2^n}}\big)|w\rangle + \sum_{x=0,x\neq w}^{2^n-1}\big(2\frac{2^n-2}{\sqrt{2^n}2^n} - \frac{1}{\sqrt{2^n}}\big)|x\rangle \\
&= \big(\frac{2\cdot 2^n - 4 + 2^n}{\sqrt{2^n}2^n}\big)|w\rangle + \sum_{x=0,x\neq w}^{2^n-1}\big(\frac{2\cdot 2^n - 4 - 2^n}{\sqrt{2^n}2^n}\big)|x\rangle \\
&= \big(\frac{3\cdot 2^n - 4}{\sqrt{2^n}2^n}\big)|w\rangle + \sum_{x=0,x\neq w}^{2^n-1}\big(\frac{2^n-4}{\sqrt{2^n}2^n}\big)|x\rangle.
\end{aligned}
\tag{7.19}
$$

A Figura 7.3 representa os coeficientes do estado de controle, após a aplicação do operador $U_{\psi^\perp}$

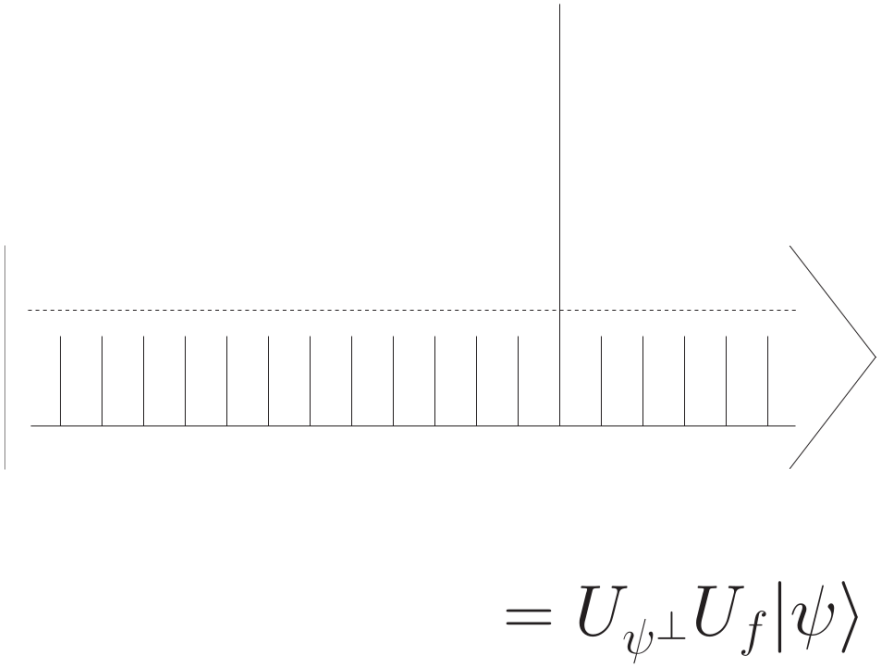


Figura 7.3: Superposição do estado de controle no algoritmo de Grover, após a primeira aplicação a porta $U_{\psi^\perp}$ [KLM07].

É denominado como o operador de Grover G, a aplicação dos operadores U_f e $U_{\psi^\perp}$, ou seja $G = U_f U_{\psi^\perp}$.

No algoritmo Grover realiza-se uma sequência de operadores G, o objetivo desta sequência de operações é obter um estado onde a probabilidade de obter $|w\rangle$, após uma medição, é a mais próxima de 1 possível. Esta sequência utiliza $O(\sqrt{2^n})$ operadores para atingir seu ótimo, algo que será demonstrado na Seção 7.1.3.

7.1.3 Convergência

Inicialmente serão realizadas algumas alterações para facilitar a análise:

$$N = 2^n.$$

Utilizando (7.11) e o estado $|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$, é possível notar que:

$$U_{\psi^\perp} = 2|\psi\rangle\langle\psi| - I.$$

Seja $|w_\perp\rangle$ uma superposição uniforme de todos os estados que são ortogonais à $|w\rangle$, na base $\{|0\rangle, |1\rangle\}$. Ou seja:

$$|w_\perp\rangle = \frac{1}{\sqrt{N-1}} \sum_{x=0, x\neq w}^{2^n-1} |x\rangle.$$

É possível reescrever $|\psi\rangle$, utilizando $\{|w\rangle, |w_\perp\rangle\}$, da seguinte forma:

$$|\psi\rangle = \sqrt{\frac{N-1}{N}}|w_\perp\rangle + \frac{1}{\sqrt{N}}|w\rangle.$$

Fazendo:

$$\cos(\theta) = \sqrt{\frac{N-1}{N}}; \quad \sin(\theta) = \frac{1}{\sqrt{N}}.$$

$$|\psi\rangle = \cos(\theta)|w_\perp\rangle + \sin(\theta)|w\rangle.$$

Analisando a atuação dos operadores U_f e $U_{\psi^\perp}$ sobre $\{|w\rangle, |w_\perp\rangle\}$:

$$U_f = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

$$U_{\psi^\perp} = 2|\psi\rangle\langle\psi| - I.$$

$$\begin{aligned} |\psi\rangle\langle\psi| &= \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \cdot \begin{bmatrix} \cos(\theta) & \sin(\theta) \end{bmatrix} \\ &= \begin{bmatrix} \cos(\theta)^2 & \cos(\theta)\sin(\theta) \\ \sin(\theta)\cos(\theta) & \sin(\theta)^2 \end{bmatrix}. \end{aligned} \tag{7.20}$$

$$\begin{aligned} U_{\psi^\perp} &= 2\begin{bmatrix} \cos(\theta)^2 & \cos(\theta)\sin(\theta) \\ \sin(\theta)\cos(\theta) & \sin(\theta)^2 \end{bmatrix} - I \\ &= \begin{bmatrix} 2\cos(\theta)^2 - 1 & 2\cos(\theta)\sin(\theta) \\ 2\sin(\theta)\cos(\theta) & 2\sin(\theta)^2 - 1 \end{bmatrix}. \end{aligned} \tag{7.21}$$

Utilizando as propriedades trigonométricas (7.22), é possível simplificar o operador $U_{\psi^\perp}$, resultando em (7.23).

$$\begin{aligned} &\sin(x+y) = \sin(x)\cos(y) + \cos(x)\sin(y) \\ &\cos(x+y) = \cos(x)\cos(y) - \sin(x)\sin(y) \\ &\cos(x)^2 + \sin(x)^2 = 1. \end{aligned} \tag{7.22}$$

$$\begin{aligned} U_{\psi^\perp} &= \begin{bmatrix} 2\cos(\theta))^2 - 1 & 2\cos(\theta))\sin(\theta)) \\ 2\sin(\theta))\cos(\theta)) & 2\sin(\theta))^2 - 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos(2\theta) & \sin(2\theta) \\ \sin(2\theta) & -\cos(2\theta) \end{bmatrix}. \end{aligned} \tag{7.23}$$

Utilizando os operadores $U_{\psi^\perp}$ e U_f é possível definir o operador de Grover:

$$\begin{aligned} G &= U_{\psi^\perp} U_f \\ &= \begin{bmatrix} \cos(2\theta) & \sin(2\theta) \\ \sin(2\theta) & -\cos(2\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \\ &= \begin{bmatrix} \cos(2\theta) & -\sin(2\theta) \\ \sin(2\theta) & \cos(2\theta) \end{bmatrix}. \end{aligned} \tag{7.24}$$

Aplicando G ao estado $|\psi\rangle$ e utilizando as propriedades trigonométricas (7.22):

$$\begin{aligned} G|\psi\rangle &= \begin{bmatrix} \cos(2\theta) & -\sin(2\theta) \\ \sin(2\theta) & \cos(2\theta) \end{bmatrix} \begin{bmatrix} \cos(\theta)) \\ \sin(\theta)) \end{bmatrix} \\ &= \begin{bmatrix} \cos(2\theta)\cos(\theta)) - \sin(2\theta)\sin(\theta)) \\ \sin(2\theta)\cos(\theta)) + \cos(\theta))\sin(2\theta) \end{bmatrix} \\ &= \begin{bmatrix} \cos(2\theta+\theta) \\ \sin(2\theta+\theta) \end{bmatrix} \\ &= \begin{bmatrix} \cos(3\theta) \\ \sin(3\theta) \end{bmatrix}. \end{aligned} \tag{7.25}$$

Note que a aplicação de k vezes o operador G sobre o estado $|\psi\rangle$ resulta no seguinte estado:

$$\begin{aligned} G^k|\psi\rangle &= \begin{bmatrix} \cos((2k+1)\theta) \\ \sin((2k+1)\theta) \end{bmatrix} \\ &= \cos((2k+1)\theta)|\alpha\rangle + \sin((2k+1)\theta)|\beta\rangle. \end{aligned} \tag{7.26}$$

É desejado que o coeficiente de $|w\rangle$ seja o mais próximo a 1 possível, portanto:

$$\sin((2k+1)\theta) = 1.$$

$$\sin((2k+1)\theta) = \sin(\frac{\pi}{2}).$$

$$(2k+1)\theta = \frac{\pi}{2}.$$

$$2k + 1 = \frac{\pi}{2\theta}.$$

$$2k = \frac{\pi}{2\theta} - 1.$$

$$k = \frac{\pi}{4\theta} - \frac{1}{2}.$$

Na maioria dos casos em que o algoritmo de Grover será utilizado N será extremamente grande ($N = 2^{128}$). Nestes casos é possível observar que a seguinte equação é verdadeira:

$$\sin(\theta) = \frac{1}{\sqrt{N}} \approx \theta.$$

$$\theta \approx \frac{1}{\sqrt{N}}.$$

Portanto:

$$k \approx \frac{\pi}{4\frac{1}{\sqrt{N}}} - \frac{1}{2}.$$

$$k \approx \frac{\pi\sqrt{N}}{4} - \frac{1}{2}.$$

Finalmente o número aplicações do operador G será o inteiro mais próximo de $\frac{\pi\sqrt{N}}{4} - \frac{1}{2}$, ou seja:

$$k = round(\frac{\pi\sqrt{N}}{4} - \frac{1}{2}).$$

Com isso, o algoritmo pode ser resumido à uma medição no estado $G^k|\psi\rangle$.

7.1.4 Construindo o circuito

Inicialmente é iniciado um registrador no estado $|0\rangle^n|1\rangle$ e aplicado os operadores de Hadamard $H^{\otimes n}$ sobre $|0\rangle^n$ e H sobre $|1\rangle$. Deixando o registrador no estado:

$$\begin{aligned} H^{\otimes n}|0\rangle H|0\rangle &= |\psi\rangle|-\rangle \\ &= \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \frac{|0\rangle - |1\rangle}{\sqrt{2}}. \end{aligned} \tag{7.27}$$

Em seguida é aplicada uma sequência de $round(\frac{\pi\sqrt{N}}{4} - \frac{1}{2})$, $O(\sqrt{N})$, operadores de Grover G, onde:

$$G = U_{\psi^\perp} U_f = H^{\otimes n} U_{0^\perp} H^{\otimes n} U_f.$$

Finalmente é realizada uma medição nos n qubits de controle, obtendo o valor desejado. O circuito completo para a solução do problema pode ser visualizado na Figura 7.4.

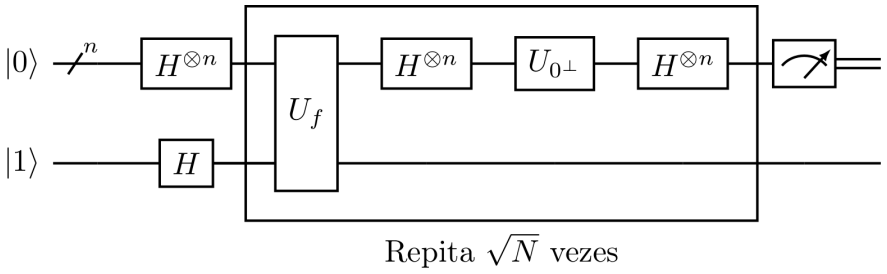


Figura 7.4: Circuito de Grover para o problema de busca não estruturada de único elemento [Hep21].

7.1.5 Algoritmo e Complexidade Computacional

Nesta subseção será apresentado o algoritmo $Grover(U_f)$, capaz de solucionar o problema de busca não estruturada para um único elemento.

Algoritmo 12 $Grover(U_f)$:

1: Inicia um estado $|0\rangle^n \frac{|0\rangle-|1\rangle}{\sqrt{2}}$.
2: Aplica o operador de Hadamard $H^{\otimes n}$ sobre o estado $|0\rangle^n$, gerando o estado $|\psi\rangle \frac{|0\rangle-|1\rangle}{\sqrt{2}}$, onde $|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$.
3: $k \leftarrow round(\frac{\pi\sqrt{2^n}}{4} - \frac{1}{2})$.
4: Aplica k vezes o operador de Grover ($G = U_f U_{\psi^\perp}$), gerando o estado $|\psi'\rangle \frac{|0\rangle-|1\rangle}{\sqrt{2}} = G^k |\psi\rangle \frac{|0\rangle-|1\rangle}{\sqrt{2}}$
5: Aplica uma medição no estado $|\psi'\rangle$, obtendo um valor $\widetilde{w}$
6: **if** $f(\widetilde{w}) = 1$ **then**
7: **return** $\widetilde{w}$
8: **else**
9: **return** ERRO
10: **end if**

Tendo que que o custo computacional dos operadores U_f e $U_\perp$ seja linear, o custo computacional do operador de Grover G será linear.

Analisando o algoritmo, é possível observar que seu maior custo está durante a aplicação dos operadores de Grover G. Como foi estipulado que o custo de G é linear o custo do algoritmo está no núermo de aplicações do operador G.

Na Linha 3 do Algoritmo 12 é definido o número de aplicações do operador de Grover como $round(\frac{\pi\sqrt{N}}{4} - \frac{1}{2})$, onde $N = 2^n$.

Portanto o custo computacional do algoritmo é:

$$O(\sqrt{N}).$$

Dado um valor grande para N, a probabilidade de sucesso deste algoritmo é aproximadamente 1. Quanto maior o valor para N, maior será a probabilidade de sucesso.

7.1.6 Algoritmo de Grover e o algoritmo AES

Como apresentado na Seção 3.4, o protocolo criptográfico AES possui uma função criptográfica f e sua inversa f^{-1}, ambas computáveis em tempo polinomial.

Estas funções utilizam-se de uma chave secreta k de 128, 192 ou 256 bits. Atualmente chaves de 128 bits são consideradas seguras, pois um ataque de força bruta teria que encontrar uma chave específica num espaço de chaves com 2^{128} chaves diferentes.

Caso o algoritmo de Grover se torne viável, implementando um circuito quântico, uma caixa preta U_f, que atue como a função f^{-1}, o custo computacional para decifrar uma mensagem criptografada utilizando o protocolo AES seria reduzido.

Na prática, chaves de 128 bits seriam tão resistentes quanto as chaves de 64 bits nos dias atuais e chaves de 256 bits seriam equivalente à chaves atuais de 128 bits.

8 CONSIDERAÇÕES FINAIS

Este estudo apresentou de forma detalhada os algoritmos quânticos propostos por Peter Shor para para resolver os problemas da fatoração e do logaritmo discreto, assim como o Algoritmo de Grover para o problema da busca não estruturada. A partir do estudo destes algoritmos foi possível estabelecer relações com os protocolos RSA, troca de chaves de Diffie-Hellman-Merkle e AES.

Ao longo dos Capítulos 4, 5 e 6 foi possível compreender os algoritmos de Shor para fatoração e logaritmo discreto, assim como construir ambos os circuitos quânticos destes algoritmos utilizando um número polinomial de portas quânticas. Através destas analises, foi possível concluir que estes algoritmos inviabilizam os protocolos de criptografia RSA e a troca de chaves de Diffie-Hellman-Merkle, descritos ao longo do Capítulo 3. No decorrer do Capítulo 7 foi explicado o algoritmo de Grover, capaz de solucionar o problema de busca não estruturada de único elemento, assim como a construção de seu circuito. Com isso foi possível relacioná-lo com o protocolo AES, descrito no Capítulo 3, constatando que este realiza uma melhoria polinomial em relação a outros ataques desenvolvidos para decifrar o protocolo, porém não chegando a inviabilizá-lo, dado que chaves grandes, como a chave de 256 bits, ainda necessitam de um tempo impraticável para serem decifradas.

A Tabela 8.1 a seguir contém uma comparação do custo computacional de algoritmos clássicos e quânticos, onde é possível visualizar uma melhoria significativa.

Tabela 8.1: Comparação entre o ataques de Shor e Grover e os melhores ataque de algoritmos clássicos conhecidos, sobre os procolos criptográficos RSA, troca de chaves de Diffie-Hellman-Merkle (D. H. M.) e AES.

Protocolo	Melhor algoritmo Clássico	Algoritmo Quântico
RSA	Tempo $e^{O((\log N)^{\frac{1}{3}})(\log\log N)^{\frac{2}{3}})}$	Tempo $O((\log(N))^2 \log\log(N) \log\log\log(N))$
D. H. M.	Tempo $e^{O((\log P)^{\frac{1}{3}})(\log\log P)^{\frac{2}{3}})}$	Tempo $O((\log(P))^2 \log\log(P) \log\log\log(P))$
AES-128	Tempo $2^{126.13}$ e espaço 2^{56}	Tempo 2^{64}
AES-196	Tempo $2^{189.91}$ e espaço 2^{48}	Tempo 2^{98}
AES-256	Tempo $2^{189.91}$ e espaço 2^{40}	Tempo 2^{128}

Durante a realização das pesquisas, relativas ao tema deste estudo, deparou-se com uma porção majoritária de materiais que não estão escritos na língua portuguesa e que exigem certo conhecimento prévio acerca de teoria dos números e matemática. Em muitos casos, isto acaba sendo um empecilho para que alunos de graduação se aproximem do tema e consigam compreendê-lo. Nesse sentido o presente estudo colabora com a disseminação do conteúdo sobre computação quântica na língua portuguesa e por apresentar um elevado nível de detalhe facilita a compreensão de tais conteúdos.

O estudo apresentado cobre apenas uma pequena parcela do conteúdo acerca de computação quântica e da criptografia, tendo como objetivo apenas realizar uma análise matemática precisa sobre uma fração dos algoritmos quântico famosos e relacioná-los com alguns protocolos criptográficos importantes. O livro “Criptografia e Segurança de Redes: Princípios e Praticas” [SVBF14], possui um bom referencial acerca de criptografia e pode ser utilizado para estudar o tema mais afundo, enquanto os livros “Quantum Computation and Quantum Information: 10th Anniversary Edition” [NC11] e “An Introduction to Quantum Computing”

[KLM07] contemplam o tema de computação quântica de forma mais abrangente, e aconselha-se a leitura destes materiais para adquirir uma visão ampla sobre o tema.

Uma vez que este trabalho relaciona os algoritmos de Shor e Grover com os protocolos RSA, troca de chaves de Diffie-Hellman-Merkle e AES, ele pode ser ampliado com um estudo de outros protocolos criptográficos que são impactados por tais algoritmos, como é o caso protocolos baseados em curvas elípticas [PZ03]. Outras possíveis ampliações deste trabalho são algoritmos quânticos que realizam ataques a protocolos criptográficos e criptografia pós-quântica, esta ultima fundamentada em protocolos resistentes a ataques realizados por algoritmos quânticos. Adicionalmente, este trabalho pode ser estendido para abordar criptografia quântica, que é um modelo de criptografia que utiliza canais quânticos para realizar trocas de informação, um tema de extrema importância, pelo fato deste modelo possuir recursos que não estão disponíveis em modelos clássicos de criptografia, como pode ser observado nos protocolos BB84 [BB20] e E91 [Eke91].

O presente estudo proporcionou o contato com uma variedade de temas sobre computação quântica e seus relacionamentos com a criptografia, temas esses que atraem fortes interesses para novas pesquisas. Por fim este estudo também proporcionou um aprofundamento nos conhecimentos de computação quântica e algoritmos quânticos, criptografia, teoria dos números e complexidade computacional, abrindo portas para futuros estudos.

REFERÊNCIAS

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. Primes is in p. *Annals of mathematics*, pages 781–793, 2004.

[BB20] Charles H Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *arXiv preprint arXiv:2003.06557*, 2020.

[BBBV97] Charles H Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523, 1997.

[BCDP96] David Beckman, Amalavoyal N. Chari, Srikrishna Devabhaktuni, and John Preskill. Efficient networks for quantum factoring. *Physical Review A*, 54(2):1034–1063, Aug 1996.

[Ben80] Paul Benioff. The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *Journal of statistical physics*, 22(5):563–591, 1980.

[Ber98] Daniel Bernstein. Detecting perfect powers in essentially linear time. *Mathematics of computation*, 67(223):1253–1283, 1998.

[CEMM98] Richard Cleve, Artur Ekert, Chiara Macchiavello, and Michele Mosca. Quantum algorithms revisited. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1969):339–354, 1998.

[Che03] Donny Cheung. *Using generalized quantum Fourier transforms in quantum phase estimation algorithms*. PhD thesis, Citeseer, 2003.

[Cop94] Don Coppersmith. An approximate fourier transform useful in quantum factoring. *IBM Research Report*, pages RC–19642, 1994.

[Dir39] Paul Adrien Maurice Dirac. A new notation for quantum mechanics. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 35, pages 416–418. Cambridge University Press, 1939.

[DJ92] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 439(1907):553–558, 1992.

[Eke91] Artur K Ekert. Quantum cryptography based on bell's theorem. *Physical review letters*, 67(6):661, 1991.

[Gro96] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[Hep21] Henrique Hepp. Retirado do manuscrito a ser submetido para o exame de qualificação de doutorado, 2021.

[HH00] Lisa Hales and Sean Hallgren. An improved quantum fourier transform algorithm and applications. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 515–525. IEEE, 2000.

[HW80] Godfrey H. Hardy and Edward M. Wright. *An introduction to the theory of numbers.* Oxford University Press, 5 edition, 1980.

[Kit95] A. Yu. Kitaev. Quantum measurements and the abelian stabilizer problem. *arXiv preprint quant-ph/9511026*, 1995.

[KLM07] Phillip Kaye, Raymond Laflamme, and Michele Mosca. *An Introduction to Quantum Computing*. Oxford University Press, Inc., USA, 2007.

[LLMP93] Arjen K Lenstra, Hendrik W Lenstra, Mark S Manasse, and John M Pollard. The number field sieve. In *The development of the number field sieve*, pages 11–42. Springer, 1993.

[Mer78] Ralph C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4):294–299, April 1978.

[Mol08] Richard A. Mollin. *Fundamental Number Theory with Applications, Second Edition*, pages 21–22. Chapman & Hall/CRC, 2 edition, 2008.

[NC11] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, USA, 10th edition, 2011.

[PZ03] John Proos and Christof Zalka. Shor's discrete logarithm quantum algorithm for elliptic curves. *arXiv preprint quant-ph/0301141*, 2003.

[RD01] Vincent Rijmen and Joan Daemen. Advanced encryption standard. *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology*, pages 19–22, 2001.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, February 1978.

[Sho94] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee, 1994.

[Sho99] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[SVBF14] W. Stallings, D. Vieira, A. N. Barbosa, and M. S. J. Ferreira. *Criptografia e Segurança de Redes: Princípios e Praticas*. Pearson Universidades, 6 edition, 2014.

[TW15] Biaoshuai Tao and Hongjun Wu. Improving the biclique cryptanalysis of aes. In Ernest Foo and Douglas Stebila, editors, *Information Security and Privacy*, pages 39–56, Cham, 2015. Springer International Publishing.

[Vin03] I. M. Vinogradov. *Elements of number theory*, chapter § VI Primitive roots and indices, pages 105–121. Dover phoenix editions. Dover Publications, Mineola, N.Y, 2003.

APÊNDICE A – COMPLEMENTOS DA COMPUTAÇÃO QUÂNTICA

Este apêndice está destinado à complementar análises relacionadas à algoritmos quânticos apresentadas neste trabalho.

A.1 DEMONSTRAÇÃO DE QUE É POSSÍVEL CONSTRUIR UM OPERADOR DE CONTROLE *CT-U*, A PARTIR DE UM OPERADOR *U* UNITÁRIO, QUE ATUA SOBRE UM ÚNICO QUBIT

Esta seção foi retirada do livro "Quantum Computation and Quantum Information: 10th Anniversary Edition" [NC11], recebendo mínimas alterações.

Dado as matrizes e Pauli e a matriz identidade:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}; \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}; \quad I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

É possível escrever os seguintes operadores de rotação, os quais rotacionam o estado do qubit nos eixos $\hat{x}$, $\hat{y}$ e $\hat{z}$:

$$\begin{aligned} R_x(\theta) &\equiv e^{-i\theta X/2} = \cos(\frac{\theta}{2})I - i\sin(\frac{\theta}{2})X = \begin{bmatrix} \cos(\frac{\theta}{2}) & -i\sin(\frac{\theta}{2}) \\ -i\sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}; \\ R_y(\theta) &\equiv e^{-i\theta Y/2} = \cos(\frac{\theta}{2})I - i\sin(\frac{\theta}{2})Y = \begin{bmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}; \\ R_z(\theta) &\equiv e^{-i\theta Z/2} = \cos(\frac{\theta}{2})I - i\sin(\frac{\theta}{2})Z = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}. \end{aligned} \tag{A.1}$$

Teorema A.1.1 (decomposição Z-Y para um único qubit)**.** *Seja U um operação unitária sobre um único qubit. Então existem os números reais α, β, γ e δ, tais que:*

$$U = e^{i\alpha} R_z(\beta) R_y(\gamma) R_z(\delta). \tag{A.2}$$

Demonstração. Como U é unitário, as linhas e as colunas de U são ortonormais, do qual segue que existem números reais α, β, γ e δ, tais que:

$$U = \begin{bmatrix} e^{-i(\alpha-\beta/2-\delta/2)}\cos(\frac{\gamma}{2}) & -e^{-i(\alpha-\beta/2+\delta/2)}\sin(\frac{\gamma}{2}) \\ e^{-i(\alpha+\beta/2-\delta/2)}\sin(\frac{\gamma}{2}) & e^{-i(\alpha+\beta/2+\delta/2)}\cos(\frac{\gamma}{2}) \end{bmatrix}.$$

Observe que (A.2) segue imediatamente das definições das matrizes de rotação e da multiplicação matricial. □

Lema A.1.2. *Seja U uma operação unitária sobre um único qubit. Então existem os operador A, B e C, que atuam sobre um único qubit, tais que $ABC = I$ e $U = e^{i\alpha}AXBXC$, onde α é uma fase global.*

Demonstração. Utilizando a notação do Teorema A.1.1 faça:

$$\begin{aligned} A &\equiv R_z(\beta)R_y(\frac{\gamma}{2}); \\ B &\equiv R_y(-\frac{\gamma}{2})R_z(-\frac{\gamma-\beta}{2}); \\ C &\equiv R_z(\frac{\gamma+\beta}{2}). \end{aligned} \tag{A.3}$$

Utilizando as seguintes propriedades trigonométricas:

$$\begin{aligned} &\sin(\theta_1+\theta_2) = \sin(\theta_1)\cos(\theta_2) + \cos(\theta_1)\sin(\theta_2), \quad \theta_1,\theta_2 \in \mathbb{R}; \\ &\cos(\theta_1+\theta_2) = \cos(\theta_1)\cos(\theta_2) - \sin(\theta_1)\sin(\theta_2), \quad \theta_1,\theta_2 \in \mathbb{R}. \end{aligned} \tag{A.4}$$

É possível inferir que:

$$\begin{aligned} R_x(\theta_1)R_x(\theta_2) &= R_x(\theta_1+\theta_2), \quad \theta_1,\theta_2 \in \mathbb{R}; \\ R_y(\theta_1)R_y(\theta_2) &= R_y(\theta_1+\theta_2), \quad \theta_1,\theta_2 \in \mathbb{R}; \\ R_z(\theta_1)R_z(\theta_2) &= R_z(\theta_1+\theta_2), \quad \theta_1,\theta_2 \in \mathbb{R}. \end{aligned} \tag{A.5}$$

E com isso:

$$ABC = R_z(\beta)R_y(\frac{\gamma}{2})R_y(-\frac{\gamma}{2})R_z(-\frac{\delta-\beta}{2})R_z(\frac{\delta+\beta}{2}) = I. \tag{A.6}$$

Como:

$$\begin{aligned} &X^2 = I; \\ &XR_y(\theta)X = R_y(-\theta), \quad \theta \in \mathbb{R}; \\ &XR_z(\theta)X = R_z(-\theta), \quad \theta \in \mathbb{R}. \end{aligned} \tag{A.7}$$

Pode ser visto que:

$$\begin{aligned} XBX &\equiv XR_y(-\frac{\gamma}{2})XXR_z(-\frac{\delta-\beta}{2})X \\ &\equiv R_y(\frac{\gamma}{2})R_z(\frac{\delta-\beta}{2}). \end{aligned} \tag{A.8}$$

Portanto:

$$\begin{aligned} AXBXC &= R_z(\beta)R_y(\frac{\gamma}{2})R_y(\frac{\gamma}{2})R_z(\frac{\delta-\beta}{2})R_z(\frac{\delta+\beta}{2}) \\ &= R_z(\beta)R_y(\gamma)R_z(\delta). \end{aligned} \tag{A.9}$$

Portanto $U = e^{i\alpha}AXBXC$ e $ABC = I$, conforme foi enunciado. □

Teorema A.1.3. *Dado um operador unitário U, que atua sobre um único qubit, é possível construir um operador de controle ct-U*

Demonstração. Utilizando o Lema A.1.2 é possível escrever U na forma:

$$U = e^{i\alpha}AXBXC. \tag{A.10}$$

Onde A, B e C são matrizes que atuam sobre um único qubit, tais que $ABC = I$.

Note que o operador de controle *ct-X* pode ser construído utilizando a matriz de Pauli X.

Portanto é possível implementar a porta lógica *ct-U*, substituindo X por *ct-X* e adicionando o seguinte operador ao qubit de controle:

$$\begin{bmatrix} 1 & 0 \\ 1 & e^{i\alpha} \end{bmatrix}. \tag{A.11}$$

O circuito presente na Figura A.1 corresponde à construção de um operador de controle arbitrário *ct-U*.

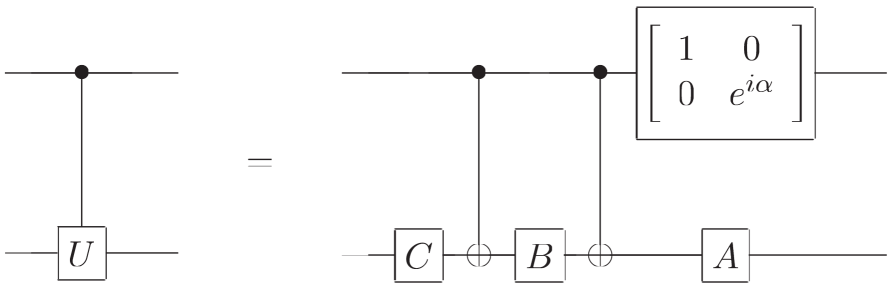


Figura A.1: Construção do operador de controle *ct-U*, onde U é um operador de que atua sobre um único qubit. α, A, B e C satisfazem $U = e^{i\alpha}AXBXC$, $ABC = I$ [NC11].

□

A.2 ANÁLISE DE ERRO NO ALGORITMO DE ESTIMAÇÃO DE FASE

Teorema A.2.1. *Dado um operador U e um autovetor $|\psi\rangle$ de U, com autovalor $e^{2\pi iw}$ correspondente, onde $0 \leq w < 1$ e não pode ser representado com n bits.*
Então a probabilidade do algoritmo quântico para estimação de fase, ao aplicar a transformação quântica inversa de Fourier ($\mathcal{QFT}^\dagger$), gerar a melhor aproximação para w utilizando n bits é de, pelo menos, $4/\pi^2 = 0.405\ldots$.

Demonstração. Para provar isso, utilizando a notação definida em (4.6), faça $\frac{a}{2^n} = [0.a_1 \ldots a_n]$, onde $a \in \{1, 2, \cdots, 2^n - 1\}$ a melhor n-bit estimação para w. Então $w = \frac{a}{2^n} + \delta$, onde $0 < \delta \leq \frac{1}{2^{n+1}}$

Aplicando a transformada quântica inversa de Fourier ao estado (5.6), descrito a seguir:

$$|\phi\rangle = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} e^{2\pi iwj}|j\rangle$$

$$|\phi\rangle = \left(\frac{|0\rangle + e^{2\pi i(2^{n-1}w)}|1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle + e^{2\pi i(2^{n-2}w)}|1\rangle}{\sqrt{2}}\right) \otimes \cdots \otimes \left(\frac{|0\rangle + e^{2\pi i(w)}|1\rangle}{\sqrt{2}}\right)$$

o seguinte estado é obtido:

$$\begin{aligned}\frac{1}{2^n}\sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1} e^{-2\pi ikj/2^n} e^{2\pi iwj}|k\rangle &= \frac{1}{2^n}\sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1} e^{-2\pi ikj/2^n} e^{2\pi i(a/2^n+\delta)j}|k\rangle \\ &= \frac{1}{2^n}\sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1} e^{2\pi i(a-k)j/2^n} e^{2\pi i\delta j}|k\rangle\end{aligned} \tag{A.12}$$

Então o coeficiente para $k = a$ é:

$$\begin{aligned}\frac{1}{2^n}\sum_{j=0}^{2^n-1} e^{2\pi i(a-a)j/2^n} e^{2\pi i\delta j} &= \frac{1}{2^n}\sum_{j=0}^{2^n-1} e^{(2\pi i\delta)^j} \\ &= \frac{1}{2^n}\left(\frac{1-(e^{2\pi i\delta})^{2^n}}{1-e^{2\pi i\delta}}\right)\end{aligned} \tag{A.13}$$

Como $|1 - e^{2i\theta}| = 2|\sin(\theta)|$, então:

$$|1 - e^{2\pi i\delta 2^n}| = 2|\sin(\pi\delta 2^n)| \tag{A.14}$$

$$|1 - e^{2\pi i\delta}| = 2|\sin(\pi\delta)| \tag{A.15}$$

Como $\sin(\pi x) \geq 2x, \forall x \in [0, \frac{1}{2}]$ e $0 < \delta 2^n \leq \frac{1}{2}$, pois $0 < \delta \leq \frac{1}{2^{n+1}}$, então:

$$|\sin(\pi\delta 2^n)| \geq 2\delta 2^n \tag{A.16}$$

Portanto, substituindo em (A.14) se obtém o seguinte resultado:

$$|1 - e^{2\pi i\delta 2^n}| \geq 2|2\delta 2^n| \tag{A.17}$$

Como $\sin(\pi x) \leq \pi x, \forall x > 0$ e como $0 < \delta$, então:

$$|\sin(\pi\delta 2^n)| \leq \pi\delta \tag{A.18}$$

Portanto, substituindo em (A.15) se obtém o seguinte resultado:

$$|1 - e^{2\pi i\delta 2^n}| \leq 2|\pi\delta| \tag{A.19}$$

Consequentemente, substituindo (A.17), (A.19) em (A.13), a probabilidade de se obter o estado $a_1 \ldots a_n$ é:

$$\left|\frac{1}{2^n}\left(\frac{1-(e^{2\pi i\delta})^{2^n}}{1-e^{2\pi i\delta}}\right)\right|^2 \geq \left(\frac{1}{2^n}\left(\frac{4\delta 2^n}{2\pi\delta}\right)\right)^2 = \frac{4}{\pi^2} \tag{A.20}$$

□

Note que, se δ, utilizado na prova do Teorema A.2.1, for igual a $\frac{1}{2}$, então, com probabilidade de $8/\pi^2 = 0.811\ldots$, será obtido uma das duas representação de n bits mais próximas de w como resultado do algoritmo de estimação de fase.

Utilizando os resultados obtidos por Donny Cheung [Che03], é possível inferir o seguinte lema:

Lema A.2.2. *Dado um operador U e um autovetor $|\psi\rangle$ de U, com autovalor $e^{2\pi i w}$ correspondendo, onde $0 \leq w < 1$ e não pode ser representado com n bits.*
Então, com probabilidade $8/\pi^2$, o algoritmo de estimação de fase irá produzir um valor $\hat{x} \in \{0, \cdots, 2^n - 1\}$, de modo a satisfazer:

$$\left| \frac{\hat{x}}{2^n} - w \right| \leq \frac{1}{2^n} \tag{A.21}$$

APÊNDICE B – TEORIA DOS NÚMEROS

Este apêndice tem como proposito trazer ferramentas matemáticas, as quais são utilizadas neste trabalho. Este apêndice foi retirado e adaptado do livro "Quantum Computation and Quantum Information: 10th Anniversary Edition" [NC11] em conjunto com o livro "An introduction to the theory of numbers" [HW80].

B.1 FUNDAMENTOS

Esta seção está destinada à enunciar fundamentos matemáticos, os quais serão utilizados neste apêndice.

B.1.1 Inverso modular

Sejam $a, n \in \mathbb{Z}$. Define-se $x \in \mathbb{Z}$, como o inverso modular de a módulo n, todos os valores de x que satisfazem a seguinte equação:

$$ax \equiv 1 \pmod{n}. \tag{B.1}$$

O inverso modular de a módulo n, também pode ser representado na forma de a^{-1}:

$$aa^{-1} \equiv 1 \pmod{n}. \tag{B.2}$$

B.1.2 Ordem módulo n

Seja $a, n \in \mathbb{Z}$. Define-se $r \in \mathbb{Z}_{>0}$, como a ordem de a módulo n, o menor inteiro positivo, que satisfaz a seguinte equação:

$$a^r \equiv 1 \pmod{n}. \tag{B.3}$$

Ao decorrer deste apêndice, a ordem de a módulo n poderá ser representada na forma $ord(a, n)$:

$$r = ord(a, n). \tag{B.4}$$

B.1.3 Função Totiente de Euler

Seja φ totiente de Euler. Então $\varphi : \mathbb{Z}_{>0} \mapsto \mathbb{Z}_{>0}$ onde $\varphi(n)$ é definido da seguinte forma:

$$\varphi(n) = \sum_{k=0}^{n-1} \begin{cases} 1 & \text{se } mdc(k, n) = 1; \\ 0 & \text{se } mdc(k, n) \neq 1. \end{cases}$$

É possível observar que a função φ possui as seguintes propriedades, onde $a, b \in \mathbb{Z}_{>0}$ e p um número primo:

$$\varphi(p) = (p - 1). \tag{B.5}$$

$$\varphi(p^a) = (p^{a-1})(p - 1). \tag{B.6}$$

$$\varphi(ab) = \varphi(a)\varphi(b). \tag{B.7}$$

B.1.4 Teoria de grupos

Seja $\mathbb{Z}_n^\times$ o conjunto de números inteiros positivos menores que n coprimos de n, onde $n \in \mathbb{Z}_{>0}$.

Diz-se que o grupo $\mathbb{Z}_n^\times$ é cíclico multiplicativo módulo n, se existe um número gerador $g \in \mathbb{Z}_n^\times$, tal que $\forall x \in \mathbb{Z}_n^\times, \exists k \in \mathbb{Z}_{>0}$ que satisfaz a seguinte equação:

$$x \equiv g^k \pmod{n}. \tag{B.8}$$

Para favorecer a simplicidade, um grupo $\mathbb{Z}_n^\times$ será denominado cíclico quando este for cíclico multiplicativo módulo n.

Teorema B.1.1. *Seja p um número primo ímpar, $a \in \mathbb{Z}_{>0}$. Então $\mathbb{Z}_{p^a}^\times$ é cíclico.*

Este teorema foi adaptado do livro "Elements of number theory"[Vin03] e não receberá demonstração.

B.2 COROLÁRIOS, LEMAS E TEOREMAS UTEIS

Esta seção tem como objetivo enunciar e provar corolários, lemas e teoremas que serão utilizados neste trabalho.

Lema B.2.1. *Seja p um número primo e $k \in \{1, 2, \cdots, p-1\}$. Então p divide $\binom{p}{k}$.*

Demonstração. Dado que $\binom{p}{k} = \frac{p!}{k!(p-k)!}$. Considere a identidade:

$$p(p-1)\cdots(p-k+1) = \binom{p}{k} k!. \tag{B.9}$$

Como $k \geq 1$ o lado esquerdo da equação, assim como o direito, são divisíveis por p. Como $k < p$, o termo $k!$ não é divisível por p. Portanto $\binom{p}{k}$ é divisível por p. □

Teorema B.2.2 (Pequeno teorema de Fermat)**.** *Seja p um número primo e $y \in \mathbb{Z}$. Então $y^p \equiv y \pmod{p}$. Se y não é divisível por p, então $y^{p-1} \equiv 1 \pmod{p}$*

Demonstração. Dado que se $y^p \equiv y \pmod{p}$ e $p \nmid y$, é possível inferir a segunda parte do teorema:

$$y^{p-1} \equiv 1 \pmod{p}. \tag{B.10}$$

Portanto será provado que:

$$y^p \equiv y \pmod{p}. \tag{B.11}$$

A prova será conduzida por indução em y. Após completar a prova por indução, a construção da prova para inteiros não positivos y se tornará trivial.

Base $y = 1$**:** $1^p \equiv 1 \pmod{p}$.
Hipótese de indução: Suponha que $y^p \equiv y \pmod{p}$ seja verdade.
Passo: Será provado que $(y+1)^p \equiv (y+1) \pmod{p}$.
Utilizando expansão binomial a seguinte equação é obtida:

$$(1+y)^p = \sum_{k=0}^{p} \binom{p}{k} y^k. \tag{B.12}$$

A qual pode ser reescrita na seguinte forma:

$$\begin{aligned}(1+y)^p &= \sum_{k=0}^{p} \binom{p}{k} y^k \\ &= \binom{p}{0} y^0 + \sum_{k=1}^{p-1} \binom{p}{k} y^k + \binom{p}{p} y^p \\ &= y^0 + \sum_{k=1}^{p-1} \binom{p}{k} y^k + y^p.\end{aligned} \tag{B.13}$$

Pelo Lema B.2.1, $p \mid \sum_{k=1}^{p-1} \binom{p}{k} y^k$, então:

$$\begin{aligned}(1+y)^p &= 1 + y^p + \sum_{k=1}^{p-1} \binom{p}{k} y^k \\ &\equiv 1 + y^p + \sum_{k=1}^{p-1} \binom{p}{k} y^k \pmod{p} \\ &\equiv 1 + y^p \pmod{p}.\end{aligned} \tag{B.14}$$

Como, pela hipótese de indução, $y^p \equiv y \pmod{p}$, portanto:

$$\begin{aligned}(1+y)^p &\equiv 1 + y^p \pmod{p} \\ &\equiv 1 + y \pmod{p}.\end{aligned} \tag{B.15}$$

□

Lema B.2.3 (Lema do cancelamento)**.** *Sejam,* $a, b, x, n \in \mathbb{Z}$*, tais que*

$$xa \equiv xb \pmod{n}. \tag{B.16}$$

Se x é coprimo de n, então

$$a \equiv b \pmod{n}. \tag{B.17}$$

Demonstração. Dado que:

$$xa \equiv xb \pmod{n}. \tag{B.18}$$

Por definição:

$$\begin{aligned}n &\mid xa - xb \\ n &\mid x(a-b).\end{aligned} \tag{B.19}$$

Como x é coprimo de n, então:

$$n \mid a - b. \tag{B.20}$$

Portanto, por definição:

$$a \equiv b \pmod{n}. \tag{B.21}$$

□

Lema B.2.4. *Sejam $a, b, n \in \mathbb{Z}_{>0}$, onde $a, b < n$, $mdc(a, n) = 1$ e $mdc(b, n) = 1$. Então $ab \pmod{n}$ é coprimo de n.*

Demonstração. Inicialmente note que se a e b são coprimos de n, então, por definição, $mdc(ab, n) = 1$. Seja:

$$ab \equiv r \pmod{n}. \tag{B.22}$$

Será provado que:

$$mdc(r, n) = 1. \tag{B.23}$$

A prova segue por contradição. Suponha que $mdc(r, n) = d > 1$. Então:

$$\begin{aligned} r &= dk_r, \quad k_r \in \mathbb{Z}; \\ n &= dk_n, \quad k_n \in \mathbb{Z}. \end{aligned} \tag{B.24}$$

Como $ab \equiv r \pmod{n}$ é possível deduzir que:

$$\begin{aligned} ab &\equiv dk_r \pmod{n}, \quad k_r \in \mathbb{Z} \\ ab &= dk_r + dk_n c, \quad k_r, k_n, c \in \mathbb{Z} \\ ab &= d(k_r + k_n c), \quad k_r, k_n, c \in \mathbb{Z} \\ d &\mid ab. \end{aligned} \tag{B.25}$$

Portanto $mdc(ab, n) = d > 1$. Contradição. □

Teorema B.2.5 (Teorema de Euler)**.** *Seja y um inteiro coprimo de N. Então $y^{\varphi(N)} \equiv 1 \pmod{N}$.*

Demonstração. Seja $\{a_1, a_2, \cdots, a_{\varphi(N)}\}$ o conjunto de número coprimos de N.

Utilizando o Lema B.2.4, é possível observar que $\forall a_i \in \{a_1, a_2, \cdots, a_{\varphi(N)}\}, \exists a_j \in \{a_1, a_2, \cdots, a_{\varphi(N)}\}$, tal que $ya_i \equiv a_j$.

Como y é coprimo de N, o Lema do Cancelamento B.2.3 implica que:

$$ya_i \equiv ya_j \pmod{N} \Rightarrow a_i = a_j. \tag{B.26}$$

Portanto o conjunto $\{ya_1, ya_2, \cdots, ya_{\varphi(N)}\}$ é uma permutação, módulo N, do conjunto $\{a_1, a_2, \cdots, a_{\varphi(N)}\}$. Consequentemente:

$$\prod_{k=1}^{\varphi(n)} ya_k \equiv \prod_{k=1}^{\varphi(n)} a_k \pmod{N}. \tag{B.27}$$

$$y^{\varphi(n)} \prod_{k=1}^{\varphi(n)} a_k \equiv \prod_{k=1}^{\varphi(n)} a_k \pmod{N}. \tag{B.28}$$

Note que, pelo Lema B.2.4, $\prod_{k=1}^{\varphi(n)} a_k$ é coprimo de N. Utilizando o Lema do Cancelamento B.2.3 é possível remover $\prod_{k=1}^{\varphi(n)} a_k$ de ambos os termos da equação:

$$y^{\varphi(n)} \equiv 1 \pmod{N}. \tag{B.29}$$

□

Lema B.2.6 (Lema de Bézout)**.** *Sejam, $a, b \in \mathbb{Z}$, tais que $mdc(a, b) = d$*
Então, existem valores para $x, y, \in \mathbb{Z}$, tais que:

$$ax + by = d. \tag{B.30}$$

Demonstração. Seja $S = \{ax + by : x, y \in \mathbb{Z} \text{ e } ax + by > 0\}$.

É possível verificar que S não é vazio, pois fazendo $(x = \pm 1, y = 0)$ o conjunto S contém a ou $-a$. Portanto o conjunto contém um elemento mínimo $d = ax_d + by_d$.

Para provar que d é o $mdc(a, b)$, será provado que $d \mid a$ e para todo divisor comum c, de a e b, $c \leq d$.

Dada a divisão euclidiana de a por d:

$$a = a_1 d + r, \quad r\{0, 1, \cdots, d-1\}. \tag{B.31}$$

É possível verificar que $r \in S \cup \{0\}$:

$$\begin{aligned} r &= a - a_1 d \\ &= a - a_1(ax_d + by_d) \\ &= a(1 - a_1 x_d) + b(a_1 y_d). \end{aligned} \tag{B.32}$$

Portanto r pode ser escrito na forma $ax + by$. Como $r < d$ e, por definição d é o menor elemento do conjunto S, $r \notin S$. Consequentemente:

$$\begin{aligned} &r = 0; \\ &d \mid a. \end{aligned} \tag{B.33}$$

Similarmente:

$$d \mid b. \tag{B.34}$$

Seja c um divisor comum de a e b, então:

$$\begin{aligned} &d = ax_d + by_d \\ &d = ca_c x_d + cb_c y_d, \quad a_c, b_c \in \mathbb{Z} \\ &d = c(a_c x_d + b_c y_d), \quad a_c, b_c \in \mathbb{Z} \\ &c \mid d. \end{aligned} \tag{B.35}$$

Portanto $c \leq d$. □

Lema B.2.7. *Sejam $a, b \in \mathbb{Z}_{>0}$, onde a e b são coprimos e $a < b$.*
Então existe um inverso modular de a módulo b.

Demonstração. Utilizando o Lema de Bézout B.2.6, é possível verificar que existem valores inteiros x e y, tais que:

$$ax + by = 1. \tag{B.36}$$

$$ax = 1 + b(-y). \tag{B.37}$$

Por definição:

$$ax \equiv 1 \pmod{b}. \tag{B.38}$$

Portanto existe um inverso modular de a módulo b. □

Teorema B.2.8 (Teorema chinês do resto)**.** *Suponha que $m_1, \cdots, m_n$ sejam inteiros positivos tais que quaisquer pares m_i e m_j $(i \neq j)$ são coprimos.*
Então o sistema de equações:

$$\begin{aligned} x &\equiv a_1 \pmod{m_1}; \\ x &\equiv a_2 \pmod{m_2}; \\ &\cdots \\ x &\equiv a_n \pmod{m_n}. \end{aligned} \tag{B.39}$$

possui uma solução. Além disso, quaisquer duas soluções, x_1 e x_2, para este sistema de equações, são iguais módulo $M = m_1 m_2 \cdots m_n$, ou seja $x_1 \equiv x_2 \pmod{M}$.

Demonstração. Seja $M_i = \frac{M}{m_i}$. Observe que M_i e m_i são coprimos. Portanto, a partir do Lema B.2.7, M_i possui um inverso modular m_i, o qual será denominado de N_i.

Note que para todo i e para todo j, tais que $(i \neq j)$:

$$M_i N_i \equiv 1 \pmod{m_i}. \tag{B.40}$$

$$M_i N_i \equiv 0 \pmod{m_j}. \tag{B.41}$$

Fazendo $x = \sum_i a_i M_i N_i$:

$$\begin{aligned} x &= \sum_i a_i M_i N_i \\ &= \sum_{i,(i\neq j)} a_i M_i N_i + a_j M_j N_j \\ &\equiv \left(\sum_{i,(i\neq j)} a_i M_i N_i + a_j M_j N_j \right) \pmod{m_j} \\ &\equiv \left(\sum_{i,(i\neq j)} a_i M_i N_i \pmod{m_j} + a_j M_j N_j \pmod{m_j} \right) \pmod{m_j} \\ &\equiv \left(\sum_{i,(i\neq j)} a_i 0 \pmod{m_j} + a_j 1 \pmod{m_j} \right) \pmod{m_j} \\ &\equiv a_j \pmod{m_j}. \end{aligned} \tag{B.42}$$

Portanto $\sum_i a_i M_i N_i$ é uma solução para as equações.

Seja x_1 e x_2 quaisquer soluções para o sistema de equações, então, $x_1 - x_2 \equiv 0 \pmod{m_i}$, para todo i, e portanto m_i divide $(x_1 - x_2)$, para todo i. Como os números m_i são coprimos, então o produto $M = m_1 m_2 \cdots m_n$ também divide $(x_1 - x_2)$, portanto $x_1 \equiv x_2 \pmod{M}$. □

Lema B.2.9. *Seja n um número inteiro e seja $\mathbb{Z}_n^\times$ o subconjunto dos elementos de $\mathbb{Z}_n$ que são coprimos de n. Então se $\mathbb{Z}_n^\times$ é cíclico com um gerador g, a ordem de g módulo n é $\varphi(n)$*

Demonstração. Como $\varphi(n)$ corresponde a quantidade de números coprimo de n, podemos dizer que $\varphi(n) = |\mathbb{Z}_n^\times|$.

Seja g um gerador para $\mathbb{Z}_n^\times$ e r a ordem de g módulo n. Pelo Teorema de Euler B.2.5 $g^{\varphi(n)} \equiv 1 \pmod{n}$ e portanto, por definição, $r \leq \varphi(n)$.

A prova segue por contradição. Suponha que $r < \varphi(n)$, então:

$$\begin{aligned} g^1 &= a_1 \pmod{n}; \\ g^2 &= a_2 \pmod{n}; \\ &\cdots \\ g^r &= a_r \pmod{n}; \\ g^{r+1} &= a_1 \pmod{n}; \\ g^{r+2} &= a_2 \pmod{n}; \\ &\cdots . \end{aligned} \tag{B.43}$$

Portanto $|\{a_1, a_2, \cdots, a_r\}| = r$ é a quantidade de elementos que g pode gerar. Como $r < \varphi(n) = |\mathbb{Z}_n^\times|$, $\exists \alpha \in \mathbb{Z}_n^\times$ tal que $\alpha \notin \{a_1, a_2, \cdots, a_r\}$ e portanto g não é um gerador de $\mathbb{Z}_n^\times$. Contradição. □

Lema B.2.10. *Sejam $a, N \in \mathbb{Z}_{>0}$, tais que $mdc(a, N) = 1$ e $a < N$. Seja r a ordem de a módulo N. Então $r < N$.*

Demonstração. A prova segue diretamente do Teorema de Euler B.2.5.

Dado que $a^{\varphi(N)} \equiv 1 \pmod{N}$, então $r \leq \varphi(N)$.

Como $\varphi(N) < N$, portanto:

$$r \leq \varphi(N) < N$$

□

Lema B.2.11. *Seja $x, a, N \in \mathbb{Z}_{>0}$, tais que $x < N$ e $x^a \equiv 1 \pmod{N}$. Faça $r = ord(x, N)$. Então:*

$$r \mid a. \tag{B.44}$$

Demonstração. A prova segue por contradição. Suponha que $r \nmid a$, então:

$$a = rc + b, \quad b \in \{1, 2, \cdots, r-1\}, c \in \mathbb{Z}_{\geq 0}. \tag{B.45}$$

Portanto:

$$\begin{aligned} x^a &\equiv x^{rc+b} \pmod{N} \\ &\equiv x^{rc} x^b \pmod{N} \\ &\equiv (x^r)^c x^b \pmod{N} \\ &\equiv 1^c x^b \pmod{N} \\ &\equiv x^b \pmod{N} \end{aligned} \tag{B.46}$$

Como r é o menor inteiro positivo que satisfaz a equação $x^r \not\equiv 1 \pmod{N}$ e $b \in \{1, 2, \cdots, r-1\}$:

$$x^b \not\equiv 1 \pmod{N}. \tag{B.47}$$

Portanto:

$$x^a \not\equiv 1 \pmod{N}. \tag{B.48}$$

Contradição. □

B.3 COMPLEMENTO DA REDUÇÃO DO PROBLEMA DE FATORAÇÃO PARA O PROBLEMA DA ORDEM

Esta seção está destinada a complementar a redução do problema de fatoração para o problema da ordem, a qual foi introduzida na Seção 6.1.1. O resultado final desta seção será o Teorema B.3.4, enquanto os lemas apresentados servirão como ferramentas para prová-lo.

Lema B.3.1. *Seja p um número primo ímpar e $a \in \mathbb{Z}_{>0}$. Seja 2^d a maior potência de 2, tal que*

$$2^d \mid \varphi(p^a). \tag{B.49}$$

Seja $y \in \mathbb{Z}_{p^a}^{\times}$ escolhido uniformemente ao acaso, então

$$\mathbb{P}(2^d \mid r) = \frac{1}{2}. \tag{B.50}$$

Demonstração. Inicialmente note que p é impar, logo $\varphi(p^a) = p^{a-1}(p-1)$ é par e $d > 1$

Pelo Teorema B.1.1, o conjunto $\mathbb{Z}_{p^a}^{\times}$ é cíclico, portanto existe um gerador g para este conjunto. Isto é, qualquer elemento $y \in \mathbb{Z}_{p^a}^{\times}$ pode ser escrito na forma $g^k \pmod{p^a}$, onde $k \in \{1, 2, \cdots, \varphi(p^a)\}$.

Faça $r = ord(g^k, p^a)$. A prova segue dividida em dois casos.

Caso k seja impar:

Como $\varphi(p^a) = ord(g, p^a)$ e $g^{kr} \equiv 1 \pmod{p^a}$, o Lema B.2.11 implica que:

$$\varphi(p^a) \mid kr. \tag{B.51}$$

Posto que $2^d \mid \varphi(p^a)$, então:

$$\varphi(p^a) \mid kr \therefore 2^d \mid kr. \tag{B.52}$$

Como k é impar:

$$2^d \mid r. \tag{B.53}$$

Caso k seja par:

Dado que $g^{\varphi(p^a)} \equiv 1 \pmod{p^a}$ e k é par, é possível afirmar que:

$$\begin{aligned} g^{\varphi(p^a)\frac{k}{2}} &\equiv 1 \pmod{p^a} \\ g^{\frac{\varphi(p^a)}{2}k} &\equiv 1 \pmod{p^a} \\ (g^k)^{\frac{\varphi(p^a)}{2}} &\equiv 1 \pmod{p^a}. \end{aligned} \tag{B.54}$$

Utilizando o Lema B.2.11 em conjunto com o resultado da equação anterior e visto que $r = ord(g^k, p^a)$, pode se afirmar que:

$$r \mid \frac{\varphi(p^a)}{2}. \tag{B.55}$$

Consequentemente $r \mid \frac{2^d}{2}$, e portanto:

$$2^d \nmid r. \tag{B.56}$$

Note que $\forall y \in \mathbb{Z}_{p^a}^{\times}, \exists k \in \{1, 2, \cdots, \varphi(p^a)\}$, tal que $y \equiv g^k \pmod{p^a}$. Escolhendo uniformemente ao acaso um elemento $y \in \mathbb{Z}_{p^a}^{\times}$, k será par, com probabilidade exatamente de $\frac{1}{2}$. Portanto:

$$\mathbb{P}(2^d | r) = \frac{1}{2}. \tag{B.57}$$

□

Lema B.3.2. *Seja p um número primo ímpar, $a \in \mathbb{Z}_{>0}$, $y \in \mathbb{Z}_{p^a}^{\times}$ um número escolhido uniformemente ao acaso.*
Então $\forall e \in \mathbb{Z}_{\geq 0}$. A probabilidade de 2^e ser a maior potência de 2 que divide a ordem de y módulo p^a é de, no máximo, $\frac{1}{2}$.

Demonstração. Seja r a ordem de y módulo p^a, 2^h a maior potência de 2 que divide r e 2^d a maior potência de 2 que divide $\varphi(p^a)$.

A partir do Teorema B.3.1 é possível deduzir que:

$$\mathbb{P}(2^d \mid r) = \frac{1}{2} \therefore \mathbb{P}(2^d \mid 2^h) = \frac{1}{2}. \tag{B.58}$$

Utilizando o Teorema de Euler B.2.5, é possível inferir que:

$$y^r \equiv 1 \pmod{p^a}. \tag{B.59}$$

$$y^{\varphi(p^a)} \equiv 1 \pmod{p^a}. \tag{B.60}$$

Pelo Teorema B.2.11, $r \mid \varphi(p^a)$. Como $2^h \mid r$ e $2^d \mid \varphi(p^a)$, então:

$$r \mid \varphi(p^a) \therefore 2^h \mid 2^d. \tag{B.61}$$

Dado que $2^h \mid 2^d$ e $\mathbb{P}(2^d \mid 2^h) = \frac{1}{2}$, logo:

$$\mathbb{P}(2^d \mid 2^h) = \frac{1}{2} \therefore \mathbb{P}(2^d = 2^h) = \frac{1}{2}. \tag{B.62}$$

Será provado que $\forall e \in \mathbb{Z}_{\geq 0}$, a probabilidade de e ser a maior potência de 2 que divide a ordem de y módulo p^a é, no máximo $\frac{1}{2}$. Isto é, $\mathbb{P}(2^e = 2^h) \leq \frac{1}{2}$. A prova segue dividindo os possíveis valores de e em dois casos.

Caso em que $e = d$:

$$\mathbb{P}(2^e = 2^h) = \frac{1}{2}. \tag{B.63}$$

Caso em que $e \neq d$:

$$\begin{aligned} \mathbb{P}(2^e = 2^h) &\leq 1 - \mathbb{P}(2^d = 2^h)) \\ &\leq \frac{1}{2}. \end{aligned} \tag{B.64}$$

Portanto:

$$\mathbb{P}(2^e = 2^h) \leq \frac{1}{2}. \tag{B.65}$$

□

Lema B.3.3. *Seja $N = p_1^{a_1} p_2^{a_2} \cdots p_l^{a_l}$, onde $l > 1$, p_j são números primos ímpares distintos e $a_j \in \mathbb{Z}_{>0}$, $\forall j \in \{1, 2, \cdots, l\}$.*
Seja $y \in \mathbb{Z}_N^\times$.
Seja r a ordem de y módulo N e 2^h a maior potência de 2 que divide r.
Seja r_j a ordem de y módulo $p_j^{a_j}$ e 2^{h_j} a maior potência de 2 que divide r_j, $\forall j \in \{1, 2, \cdots, l\}$.
Então r será par ou $y^{\frac{r}{2}} \equiv -1 \pmod{N}$ somente se:

$$h = h_j, \quad \forall j \in \{1, 2, \cdots, l\}. \tag{B.66}$$

Demonstração. Inicialmente note que:

$$\begin{aligned} y^r &\equiv 1 \pmod{N} \\ &\equiv 1 \pmod{\prod_{j=1}^{l} p_j^{a_j}} \\ &\equiv 1 \pmod{p_j^{a_j}}, \quad \forall j \in \{1, 2, \cdots, l\}. \end{aligned} \tag{B.67}$$

A partir do Teorema B.2.11 e da equação anterior é possível concluir que:

$$r_j \mid r, \quad \forall j \in \{1, 2, \cdots, l\}. \tag{B.68}$$

Portanto caso r seja ímpar, então r_j será impar, para todo $j \in \{1, 2, \cdots, l\}$. Ou seja:

$$\begin{aligned} 2^h = 2^{h_j} = 1, \quad \forall j \in \{1, 2, \cdots, l\} \\ h = h_j = 0, \quad \forall j \in \{1, 2, \cdots, l\}. \end{aligned} \tag{B.69}$$

Caso r seja par e $y^{\frac{r}{2}} \equiv -1 \pmod{N}$, é possível utilizar o Teorema Chines do Resto B.2.8 para afirmar que:

$$y^{\frac{r}{2}} \equiv -1 \pmod{p_j^{a_j}}, \quad \forall j \in \{1, 2, \cdots, l\}. \tag{B.70}$$

Como:

$$\begin{aligned} y^{\frac{r}{2}} &\equiv -1 \pmod{p_j^{a_j}}, \quad \forall j \in \{1, 2, \cdots, l\}; \\ y^{r_j} &\equiv 1 \pmod{p_j^{a_j}}, \quad \forall j \in \{1, 2, \cdots, l\}. \end{aligned} \tag{B.71}$$

Logo:

$$r_j \nmid \frac{r}{2}, \quad \forall j \in \{1, 2, \cdots, l\}. \tag{B.72}$$

Por definição 2^{h_j} é a maior potência de dois que divide r_j e 2^h é a maior potência de dois que divide r. Posto que $r_j \mid r$, então:

$$\begin{aligned} &r_j \mid r \therefore 2^{h_j} \mid 2^h \therefore h_j \leq h, \quad \forall j \in \{1, 2, \cdots, l\}; \\ &r_j \nmid \frac{r}{2} \therefore 2^{h_j} \nmid 2^{h-1} \therefore h_j > h - 1, \quad \forall j \in \{1, 2, \cdots, l\}. \end{aligned} \tag{B.73}$$

Consequentemente:

$$h_j = h, \quad \forall j \in \{1, 2, \cdots, l\}. \tag{B.74}$$

□

Teorema B.3.4. *Seja $N = p_1^{a_1} p_2^{a_2} \cdots p_l^{a_l}$, onde $l > 1$, p_j são números primos ímpares distintos e $a_j \in \mathbb{Z}_{>0}$, $\forall j \in \{1, 2, \cdots, l\}$.*
Seja $y \in \mathbb{Z}_N^\times$ escolhido uniformemente ao acaso e r a ordem de y módulo N. Então:

$$\mathbb{P}(r \text{ ser par e } y^{\frac{r}{2}} \not\equiv -1 \pmod{N}) \geq 1 - \frac{1}{2^{l-1}}. \tag{B.75}$$

Demonstração. Será provado que:

$$\mathbb{P}(r \text{ ser impar ou } y^{\frac{r}{2}} \equiv -1 \pmod{N}) \leq \frac{1}{2^{l-1}}. \tag{B.76}$$

Dado que r é a ordem de y módulo N, faça 2^h a maior potência de 2 que divide r.

Dado que r_j é a ordem de y módulo $p_j^{a_j}$, faça 2^{h_j} a maior potência de 2 que divide r_j, $\forall j \in \{1, 2, \cdots, l\}$. Pelo Lema B.3.3, r é impar ou $y^{\frac{r}{2}} \equiv -1 \pmod{N}$, somente se:

$$h_j = h, \quad \forall j \in \{1, 2, \cdots, l\}. \tag{B.77}$$

Pelo Lema B.3.2:

$$\mathbb{P}(h_j = e) \leq \frac{1}{2}, \quad \forall j \in \{1, 2, \cdots, l\}, \quad e \in \mathbb{Z}_{\geq 0}. \tag{B.78}$$

Portanto, fazendo $e = h_1$:

$$\prod_{j=2}^{l} \mathbb{P}(h_j = e) \leq \frac{1}{2^{l-1}}. \tag{B.79}$$

E com isso:

$$\mathbb{P}(h_j = h, \quad \forall j \in \{1, 2, \cdots, l\}) \leq \frac{1}{2^{l-1}}. \tag{B.80}$$

$$\mathbb{P}(r \text{ ser impar ou } y^{\frac{r}{2}} \equiv -1 \pmod{N}) \leq \frac{1}{2^{l-1}}. \tag{B.81}$$

$$\mathbb{P}(r \text{ ser par e } y^{\frac{r}{2}} \not\equiv -1 \pmod{N}) \geq 1 - \frac{1}{2^{l-1}}. \tag{B.82}$$

□