

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE  
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA  
COORDENAÇÃO DE PÓS-GRADUAÇÃO EM INFORMÁTICA

A Utilização do Algoritmo Quântico de Busca em  
Problemas da Teoria da Informação

Nigini Abilio Oliveira

Julho 2007

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE  
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA  
COORDENAÇÃO DE PÓS-GRADUAÇÃO EM INFORMÁTICA

# A Utilização do Algoritmo Quântico de Busca em Problemas da Teoria da Informação

Nigini Abilio Oliveira

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande – Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Ciência da Computação

Linha de Pesquisa: Modelos Computacionais e Cognitivos

Francisco Marcos de Assis

(Orientador)

Bernardo Lula Jr.

(Orientador)

Julho 2007

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

O49u

2007 Oliveira, Nigini Abilio.

A utilização do algoritmo quântico de busca em problemas da teoria da informação / Nigini Abilio. — Campina Grande: 2007.

50f. : il

Dissertação (Mestrado em Informática) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.

Referências.

Orientadores: Dr. Francisco Marcos de Assis e Dr. Bernardo Lula Júnior.

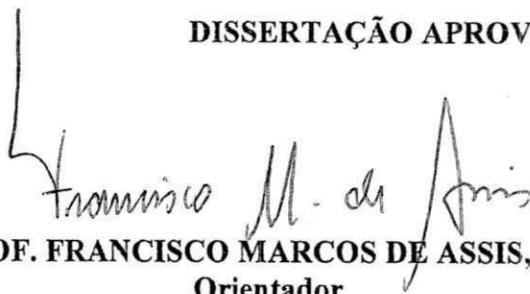
1. Teoria da Informação. 2. Computação Quântica. 3. Algoritmo de Grover. I. Título.

CDU 621.39 (043)

**“A UTILIZAÇÃO DO ALGORITMO QUÂNTICO DE BUSCA EM  
PROBLEMAS DA TEORIA DA INFORMAÇÃO”**

**NIGINI ABILIO OLIVEIRA**

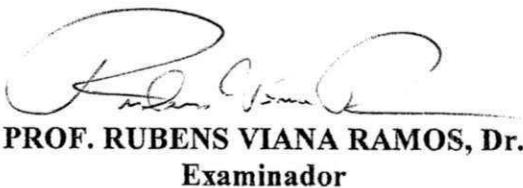
**DISSERTAÇÃO APROVADA EM 12.07.2007**



**PROF. FRANCISCO MARCOS DE ASSIS, Dr.  
Orientador**



**PROF. BERNARDO LULA JÚNIOR, Dr.  
Orientador**



**PROF. RUBENS VIANA RAMOS, Dr.  
Examinador**



**PROF. EDMAR CANDEIA GURJÃO, D.Sc  
Examinador**

**CAMPINA GRANDE - PB**

## Resumo

Esta dissertação apresenta dois resultados da aplicação do Algoritmo Quântico de Busca (Algoritmo de Grover) em problemas da área da Teoria da Informação. O primeiro deles visa testar a qualidade de um Código de Bloco Linear dado que uma importante característica dos mesmos é a distância mínima  $d$  entre as palavras que o compõe. A solução apresentada utiliza o Algoritmo da Contagem Quântica para melhorar o desempenho do cálculo de  $d$ , para o qual algoritmos clássicos são normalmente intratáveis. Assim tal distância pode ser utilizada como comparativo entre códigos diferentes. O segundo problema estudado é denominado Ataque Quântico ao Gerador Pseudo-aleatório de Blum-Micali. Tais geradores são utilizados em simulações de sistemas físicos e criptografia, substituindo geradores de números realmente aleatórios que são de difícil implementação. Neste trabalho utiliza-se o Algoritmo de Grover para quebrar a segurança do processo de geração dos números.

## **Abstract**

This work presents two results related to Quantum Search Algorithm (Grover's Algorithm) application in Information Theory problems. The first one aims to test the quality of a Linear Block Code, given that they have an important characteristic known as the minimum distance ( $d$ ) among its composing words. The presented solution uses the Quantum Counting Algorithm to improve  $d$ 's calculation since the classical algorithms are intractable. Therefore, such distance can be used as a comparative parameter between different codes. The second studied problem is called Quantum Attack to Blum-Micali's Pseudo-random Generator. Such generators are used at physical systems simulations and cryptography, replacing really random numbers generators that are difficult to implement. The Grover's Algorithm is used here to break the number generation process' security.

## **Agradecimentos**

- A Cheyenne Ribeiro pela sua dedicação e paciência.
- A meus pais por todo o amor e esforço ao ensinar-me o caminho da dedicação.
- A meus orientadores por todo ensinamento e flexibilidade no encaminhamento da pesquisa.
- Ao governo brasileiro, através do CNPQ, pelo apoio financeiro.

# Conteúdo

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	Computação Quântica . . . . .	1
1.2	Teoria da Informação . . . . .	2
1.3	Relevância . . . . .	2
1.4	Objetivos . . . . .	3
1.5	Estrutura do Trabalho . . . . .	3
<b>2</b>	<b>Computação Quântica</b>	<b>4</b>
2.1	Mecânica Quântica . . . . .	5
2.1.1	Postulados . . . . .	5
2.1.2	Efeitos Quânticos . . . . .	9
2.2	Computação . . . . .	12
2.2.1	Algoritmos Quânticos . . . . .	13
2.3	Ferramenta: Busca Quântica . . . . .	13
2.3.1	Primeira Etapa . . . . .	14
2.3.2	Segunda Etapa . . . . .	15
2.3.3	Terceira Etapa . . . . .	17
2.3.4	Leitura e Desempenho . . . . .	17
2.4	Ferramenta: Contagem Quântica . . . . .	18
2.4.1	Introdução . . . . .	19
2.4.2	Estimação de Fase . . . . .	20
2.4.3	Definição do Algoritmo da Contagem Quântica . . . . .	22
2.4.4	Conclusões . . . . .	23

---

<b>3</b>	<b>Códigos para Controle de Erros</b>	<b>25</b>
3.1	Canais . . . . .	25
3.2	Códigos de Bloco Lineares . . . . .	27
3.2.1	Decodificação . . . . .	29
3.3	Teste Quântico para Códigos de Bloco Lineares . . . . .	30
3.3.1	Cálculo da Distância Mínima . . . . .	30
3.3.2	Algoritmo para Encontrar $d$ . . . . .	35
3.3.3	Avaliação de Desempenho e Recursos . . . . .	36
<b>4</b>	<b>Ataque Quântico ao Gerador Pseudo-Aleatório de Blum-Micali</b>	<b>38</b>
4.1	Gerador de Blum-Micali . . . . .	38
4.1.1	Ataque a Geradores Pseudo-aleatórios . . . . .	39
4.2	Busca Quântica de $x_j$ . . . . .	40
4.2.1	Simulação do Ataque Clássico utilizando o Paralelismo Quântico . . . . .	41
4.2.2	Utilização do Algoritmo de Grover . . . . .	43
4.2.3	Conclusões . . . . .	44
<b>5</b>	<b>Conclusões e Trabalhos Futuros</b>	<b>46</b>
5.1	Conclusões . . . . .	46
5.2	Trabalhos Futuros . . . . .	47

# Lista de Símbolos

$|v\rangle$  - Vetor  $v$  na notação de Dirac. Lê-se ket- $v$ .

$\langle v|$  - Vetor  $v$  transposto na notação de Dirac. Lê-se bra- $v$ .

$\langle v|w\rangle$  - Produto interno entre vetores na notação de Dirac.

$\dagger$  - Operação linear sobre matrizes denominado adjunto ou conjugado Hermitiano.

$\otimes$  - Produto tensorial utilizado para a composição de espaços vetoriais.

$|+\rangle$  - Superposição equiprovável gerada pela aplicação do operador Hadamard em  $|0\rangle$ .

$|-\rangle$  - Superposição equiprovável gerada pela aplicação do operador Hadamard em  $|1\rangle$ .

$\oplus$  - Operação binário denominada XOR, ou ainda, ou-exclusivo.

# Lista de Figuras

2.1	Um qubit definido como um vetor num espaço bi-dimensional de base computacional. . . . .	6
2.2	Circuito que cria emaranhamento entre 2 qubits, onde a leitura do segundo define o valor do primeiro. . . . .	12
2.3	Estrutura de um circuito que executa uma iteração do algoritmo de Grover. . . . .	14
2.4	A evolução do sistema para o estado $ \Psi_1^*\rangle = Orac \Psi^*\rangle$ . . . . .	16
2.5	A evolução do sistema para o estado $ \Psi_2^*\rangle = Ampl \Psi_1^*\rangle$ . . . . .	17
2.6	Esquema geral do circuito que implementa o algoritmo quântico de contagem. . . . .	21
2.7	Implementação detalhada da primeira fase do algoritmo de contagem. . . . .	22
2.8	Esquema geral do circuito da contagem quântica de soluções. . . . .	23
3.1	Um canal de alfabeto binário e erro de apagamento. . . . .	26
3.2	Estrutura geral do uso de um canal de informação. . . . .	27
3.3	Duas palavras do código separadas por uma distância $d = 2t + 1$ . . . . .	30
3.4	Circuito quântico para cálculo de $d$ em códigos de blocos lineares. . . . .	32
3.5	Circuito para codificação da porta $G'$ . . . . .	33
3.6	Circuito para cálculo da porta $C$ . . . . .	35
3.7	Conjunto de testes para a etapa de contagem. . . . .	35
4.1	Circuito de ataque ao gerador pseudo-aleatório de Blum-Micali. . . . .	41
4.2	Porta similar à porta $F$ utilizando o bit mais significativo de $x_i$ . . . . .	43
4.3	Estrutura do oráculo utilizado para as iterações de Grover da porta $G^*$ . . . . .	44

# Lista de Tabelas

4.1	Funcionamento do circuito para $p = 19$ , $g = 2$ e $b = 10001000$ . . . . .	42
4.2	Permutação dos elementos utilizando $p = 19$ e $g = 2$ . . . . .	43

# Capítulo 1

## Introdução

Este trabalho apresenta o uso de Algoritmos Quânticos como ferramenta para a obtenção de soluções quânticas mais eficientes que as soluções clássicas existentes. Esta é uma abordagem bastante utilizada nos dias atuais e que tem ajudado a popularizar a Computação Quântica no meio científico. Neste capítulo serão apresentadas as áreas abrangidas, relevância e objetivos desta dissertação.

### 1.1 Computação Quântica

No último século, a base científica vigente adquiriu novas ferramentas. A denominada Física Clássica foi revista, resultando em uma nova área denominada Física Moderna. Dentro desta “nova física” encontra-se a Mecânica Quântica que reúne os postulados e resultados referentes ao funcionamento das partículas subatômicas.

A Computação Quântica (CQ) é um novo paradigma que utiliza a Mecânica Quântica como base para o processamento e transmissão de informação. Esta mudança de paradigma parece estar no caminho natural da evolução tecnológica, já que a redução no tamanho dos componentes dos atuais computadores, se aproxima rapidamente do nível atômico. A curva que descreve o uso de átomos para registrar um bit de informação prevê que na década de 2020 a proporção será de um para um [Wil98], sendo assim, as leis clássicas da Física atualmente utilizadas na computação terão que ser substituídas pela abordagem quântica.

O estudo da CQ iniciou-se na década de 1980 quando Feynman [Fey82] afirmou que nenhum sistema quântico pode ser simulado eficientemente por um sistema clássico. Tal

trabalho incentivou a pesquisa tanto na busca da construção de uma máquina quântica, como na obtenção de resultados que provassem os ganhos do novo sistema em relação ao clássico.

## 1.2 Teoria da Informação

A Teoria da Informação (TI) é uma disciplina da Matemática Aplicada relacionada ao estudo da utilização de sistemas de transmissão e armazenamento de informação criando, principalmente, limitantes de desempenhos para os mesmos. A TI é uma área de pesquisa da qual depende todo sistema que envolva o tratamento de informação, estando portanto presente em todo sistema tecnológico utilizado. Tal área abrange a transmissão de dados por canais físicos sujeitos a interferências, a compactação sem perda de informação, a segurança dos dados através de criptografia, entre outras linhas estudadas.

Esta área foi fundada pelo matemático Claude Shannon em 1948 com o trabalho denominado *Uma Teoria Matemática para a Comunicação* [Sha48]. Shannon foca-se principalmente na utilização de canais imperfeitos que durante a transmissão de informação possibilita a introdução de erros na mesma.

Neste trabalho dois problemas desta área serão analisados sob a ótica de sistemas quânticos. O primeiro deles está inserido na sub-área denominada *Codificação para Controle de Erros* que define técnicas e limitantes para a utilização de meios de armazenamento e transmissão de informação. O segundo problema está inserido na sub-área da *Segurança e Criptografia*, mais especificamente a algoritmos geradores de números pseudo-aleatórios.

## 1.3 Relevância

A Teoria da Informação é de fundamental importância para o mundo atual pois contribui para o funcionamento de toda tecnologia utilizada no dia-a-dia. Melhorias em sub-áreas como transmissão e segurança da informação são suficientemente relevantes para o contexto deste trabalho.

Além disso, a obtenção de resultados que atestem os ganhos computacionais conseguidos através da utilização da teoria quântica, fomenta o desenvolvimento da ciência em busca, principalmente, da construção de hardwares baseados na mecânica quântica.

## 1.4 Objetivos

O objetivo geral deste trabalho é demonstrar a utilização da Computação Quântica como importante meio para solucionar problemas computacionais de modo mais eficiente. Deseja-se atingir tal objetivo utilizando ferramentas/algoritmos quânticos na solução de problemas da Teoria da Informação.

Na realização do mesmo pretende-se alcançar outras metas tais como:

- gerar resultados ao menos inspiradores para futuras pesquisas nos temas relacionados;
- criar um texto acessível à pessoas da área da Computação e afins que desejem entender melhor o estudo e pesquisa da Computação Quântica.

## 1.5 Estrutura do Trabalho

Além desta introdução, este trabalho conta com quatro outros capítulos. No capítulo 2 serão apresentados os principais conceitos teóricos relacionados à Computação Quântica e os algoritmos utilizados nas contribuições científicas aqui realizadas. O capítulo 3 contém a primeira solução desenvolvida relacionada a Códigos Lineares. O capítulo 4 expõe uma segunda contribuição desenvolvida na sub-área de geradores pseudo-aleatórios. Por fim, a última parte mostra algumas conclusões e trabalhos futuros, buscando expor a satisfação dos objetivos anteriormente definidos.

# Capítulo 2

## Computação Quântica

O século XX apresenta os grandes passos do desenvolvimento científico necessários para o surgimento da chamada Computação Quântica (CQ), originada de uma vertente da Física hoje conhecida como Mecânica Quântica (MQ). Esta, por sua vez, surgiu da busca por respostas não dadas pela Física Clássica, um conjunto de leis formuladas nos séculos anteriores para explicar o funcionamento do universo.

As primeiras décadas do século XX foram marcadas pela quebra de paradigmas na Física, baseada principalmente nas pesquisas relacionadas aos átomos e suas partículas. Esta “física do muito pequeno” respondeu a questões importantes e em meados do século, a MQ já embasava os trabalhos científicos. Nas décadas de 1980 e 1990 concentram-se os principais trabalhos de matemáticos, engenheiros e cientistas da Computação na construção da CQ.

Na década de 80, percebeu-se que se a linha evolutiva dos computadores se mantivesse (o que tem sido um fato), em 40 anos as unidades de processamento e memória estariam sendo implementadas em tamanhos tão pequenos que o paradigma físico a ser aplicado deveria mudar. Desde então, a preocupação com o estudo da computabilidade destas futuras máquinas tem sido o grande impulsionador da CQ. Estudo das classes de complexidade, autômatos e máquinas de turing, circuitos e algoritmos quânticos são as principais linhas de trabalho desta área nos últimos 30 anos.

## 2.1 Mecânica Quântica

A Mecânica Quântica é uma teoria física que estuda e descreve o comportamento do mundo das partículas microscópicas. Quando se estuda o nível atômico e subatômico, esta teoria corrige e complementa a Física Clássica, usualmente abordada em dois ramos: Mecânica Newtoniana e Eletromagnetismo. Existem algumas formulações matemáticas para esta mecânica, dentre as mais utilizadas estão a mecânica ondulatória e a matricial, posteriormente unificadas pela descrição de Paul Dirac [Dir58]. Como os resultados apresentados por estas formulações não são intuitivos, várias interpretações são dadas à MQ. Este trabalho não discute tais interpretações e considera apenas o fato destacado por Nick Herbert [Her85] quando afirma que: *A mecânica funciona, independente do que se creia*. Nesta seção será feita uma introdução à formulação matricial proposta por Werner Heisenberg [Hei25] e à notação de Dirac, juntamente com os princípios da MQ, que serão utilizados por todo o texto.

### 2.1.1 Postulados

Uma mecânica pode ser vista como um conjunto de propriedades aplicáveis no estudo e manipulação de sistemas físicos. No caso de sistemas descritos pela MQ é possível a criação de métodos para a manipulação e, por assim dizer, a computação de informação. Nesta subseção será feita uma introdução a esta mecânica baseada em conceitos fundamentais como os seus postulados e efeitos.

#### Postulado 1 - Espaço de Hilbert

*Define o modelo matemático que descreve um sistema quântico.*

Este primeiro postulado associa um sistema físico qualquer a um espaço vetorial complexo denominado *Espaço de Hilbert*. Por complexo deve-se entender que as componentes do mesmo pertencem ao conjunto dos números complexos. O espaço de Hilbert é um espaço vetorial provido de produto interno (notação  $\langle v, w \rangle$ ) cuja métrica <sup>1</sup>, definida pela norma  $|v| = \sqrt{\langle v, v \rangle}$ , o torna um espaço completo <sup>2</sup>.

---

<sup>1</sup>Um espaço métrico  $(X, d)$  é um conjunto  $X$  munido de uma métrica de distância  $d(x, y)$  ( $x, y \in X$ ) que obedece a quatro propriedades:  $d(x, y)$  é um valor real, não negativo e finito; é nulo apenas para  $x = y$ ; é simétrico; e obedece a desigualdade triangular.

<sup>2</sup>Um espaço métrico é completo quando todas as sequências de Cauchy convergem.

O mais simples dos sistemas quânticos (e mais importante para este trabalho) é o *qubit* (leia-se kiubit). Este sistema pode ser descrito como um vetor num espaço de Hilbert bi-dimensional. O estado de um qubit é descrito pela soma das componentes do vetor relativo às bases do espaço. Na figura 2.1 pode ser visto um exemplo de espaço vetorial composto pela base computacional  $\{|0\rangle, |1\rangle\}$ .

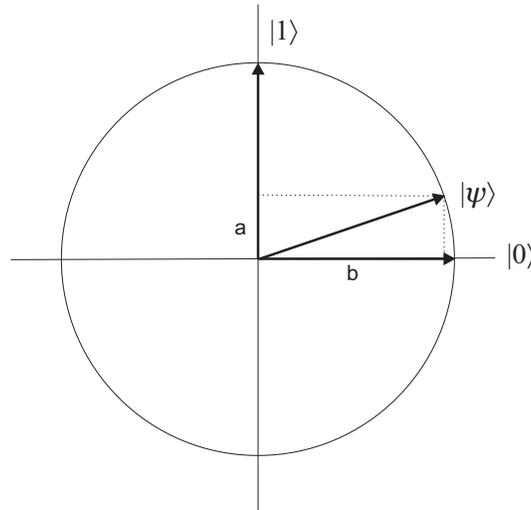


Figura 2.1: Um qubit definido como um vetor num espaço bi-dimensional de base computacional.

Na notação de Dirac, um vetor  $v$  é descrito pelo símbolo  $|v\rangle = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$  denominado *ket*.

O símbolo para o vetor transposto é denominado *bra* e escreve-se  $\langle v| = \begin{bmatrix} v_1 & v_2 \end{bmatrix}$ . Assim, pode-se escrever o estado de um qubit genérico da seguinte forma:  $|\Psi\rangle = a|0\rangle + b|1\rangle$ ; onde  $a, b \in \mathbb{C}$ , e  $\{|0\rangle, |1\rangle\}$  compõem a base do espaço.

## Postulado 2 - Operadores Unitários

*Sobre a evolução do sistema.*

A evolução de um sistema quântico de um estado  $|\Psi_1\rangle$  para um estado  $|\Psi_2\rangle$  se dá através da aplicação de um operador unitário  $U$ . Um operador é definido como uma matriz quadrada com as dimensões do espaço em questão. A aplicação do mesmo a um estado é definida como a multiplicação da matriz pelo vetor, resultando em um novo estado. Utilizando a notação de Dirac, a aplicação de uma operação é escrita da seguinte forma:  $|\Psi_2\rangle = U|\Psi_1\rangle$ .

Este postulado define que, no caso dos sistemas quânticos, os operadores devem ser unitários, o que significa que o operador  $U$  deve satisfazer:  $UU^\dagger = U^\dagger U = I$ . Esta exigência está relacionada com a manutenção da *norma*, uma característica intrínseca aos vetores. O cálculo da norma é feito através da fórmula  $\|v\| = \sqrt{\langle v|v \rangle}$ , onde  $\langle v|v \rangle$  é o produto interno também escrito como  $(|v\rangle, |v\rangle)$ . A equação 2.1 mostra que o produto interno não é alterado por operadores unitários e por conseguinte sua norma permanece constante.

$$(|v\rangle, |w\rangle) = U(|v\rangle, |w\rangle) = (U|v\rangle, U|w\rangle) = \langle v|U^\dagger U|w\rangle = \langle v|w\rangle \quad (2.1)$$

A interpretação geométrica do resultado deste produto é a do ângulo entre os vetores:  $\text{angulo}(v, w) = \arccos \frac{\langle v|w \rangle}{\|v\| \cdot \|w\|}$ . Para a MQ os vetores devem ser unitários (ou normalizados) o que significa que sua norma é 1, reduzindo a fórmula acima, ao cálculo do produto interno ( $\text{angulo}(v, w) = \arccos \frac{\langle v|w \rangle}{1 \cdot 1}$ ).

Os vetores da base do espaço devem ser ortogonais, logo o produto interno entre eles é 0. A equação 2.2 mostra tal relação.

$$\langle v|w \rangle = \begin{cases} 0, & \text{se } v \perp w \\ 1, & \text{quando } v = w \end{cases} \quad (2.2)$$

### Postulado 3 - Medições

*Mostra como devem ser feitas leituras de informações no sistema quântico.*

Um sistema computacional clássico é dito aberto pois uma entidade externa ao sistema é capaz de conhecer seu estado sem interferir na computação. Já os sistemas quânticos são fechados por definição, pois as leituras externas interferem no próprio estado lido. Este postulado vem definir a medição quântica como sendo um conjunto de operadores  $M_m$  onde o índice  $m$  refere-se ao valor de leitura possível, denominado observável. O conjunto de observáveis é definido pela base do espaço utilizada para codificar o sistema. Por exemplo, a base computacional define um estado  $|\Psi\rangle = a|0\rangle + b|1\rangle$ , determinando os observáveis  $m = \{0, 1\}$ .

A medição é um experimento probabilístico que retorna  $m$  com uma probabilidade  $p(m) = \langle \Psi | M_m^\dagger M_m | \Psi \rangle$ . Os operadores de medição na base computacional são:  $M_0 = |0\rangle\langle 0|$  e  $M_1 = |1\rangle\langle 1|$ . A equação 2.3 mostra a probabilidade de leitura do observável 0 na

base computacional. Como resultado, as probabilidades  $p(0) = |a|^2$  e  $p(1) = |b|^2$  mostram a correlação das componentes complexas do vetor e o experimento de leitura.

$$\begin{aligned}
 p(0) &= \langle \Psi | M_0^\dagger M_0 | \Psi \rangle \\
 &= (a^* \langle 0 | + b^* \langle 1 |) (|0\rangle \langle 0|0\rangle \langle 0|) (a |0\rangle + b |1\rangle) \\
 &= (a^* \langle 0 | + b^* \langle 1 |) (|0\rangle \langle 0|) (a |0\rangle + b |1\rangle) \\
 &= (a^* \langle 0 | + b^* \langle 1 |) (a \langle 0|0\rangle |0\rangle + b \langle 1|0\rangle |0\rangle) \\
 &= (a^* \langle 0 | + b^* \langle 1 |) (a \cdot 1 \cdot |0\rangle + b \cdot 0 \cdot |0\rangle) \\
 &= (a^* \langle 0 | + b^* \langle 1 |) (a |0\rangle) \\
 &= |a|^2 \langle 0|0\rangle + |b|^2 \langle 1|0\rangle \\
 &= |a|^2.
 \end{aligned} \tag{2.3}$$

Ainda como parte do postulado, o estado após a medição é definido como sendo  $|\Psi'\rangle = M_m |\Psi\rangle / \sqrt{p(m)}$ . A equação 2.4 descreve o estado após a aplicação do operador  $M_0$ . O resultado  $|\Psi'\rangle = |0\rangle$  demonstra que o estado colapsa para o vetor da base escolhido para a leitura. Este tipo de medida é conhecida como projetiva, pois leva o sistema a uma base do espaço.

$$\begin{aligned}
 |\Psi'\rangle &= \frac{M_0 |\Psi\rangle}{\sqrt{p(0)}} \\
 &= \frac{(|0\rangle \langle 0|) (a |0\rangle + b |1\rangle)}{\sqrt{|a|^2}} \\
 &= \frac{a \langle 0|0\rangle |0\rangle + b \langle 1|0\rangle |0\rangle}{|a|} \\
 &= \frac{a}{|a|} |0\rangle.
 \end{aligned} \tag{2.4}$$

Várias outras importantes consequências são definidas na literatura da área sobre medições. Porém, pelo motivo deste trabalho não utilizar mais que as idéias expostas acima, não serão definidos outros cenários de medição.

#### **Postulado 4 - Composição de Espaços**

*Tece sobre a composição de qubits em sistemas quânticos capazes de computar tarefas reais.*

Nenhum sistema com apenas uma posição de informação (i.e. um qubit no caso computacional) parece ser suficiente para computar alguma tarefa real. No caso dos computadores clássicos atuais, os processadores usam 32 ou 64 bits por instrução computacional, ou ainda

milhões de bits para guardar informações relevantes aos processos. A questão abordada por este postulado é a montagem de sistemas quânticos eficazes utilizando vários qubits.

A composição dos sistemas físicos aqui estudados é dada pelo produto tensorial dos espaços vetoriais. Matricialmente um produto tensorial ( $A \otimes B$ ) é definido como a multiplicação de cada elemento da matriz A pela matriz B, como mostra a equação 2.5. Na notação de Dirac, o produto tensorial entre vetores pode ser escrito como  $|v\rangle \otimes |w\rangle$ ,  $|v\rangle |w\rangle$  ou ainda  $|vw\rangle$ . De forma geral, sejam dois espaços vetoriais V e W de dimensões m e n respectivamente. O novo espaço  $V \otimes W$  tem dimensão  $m \times n$  e seus vetores são descritos como combinações lineares da sua base. A base do espaço  $V \otimes W$  é definida pelo produto tensorial das bases dos espaços menores.

Como exemplo, descreve-se a seguir um estado genérico para um sistema com dois qubits na base computacional. Como a base para cada um dos qubits é formada pelos vetores  $|0\rangle$  e  $|1\rangle$  tem-se que a nova base computacional é definida pelos vetores:  $|00\rangle$ ,  $|01\rangle$ ,  $|10\rangle$ ,  $|11\rangle$ , que são os produtos tensoriais das combinações entre vetores das bases menores. Assim, um estado genérico para este sistema é escrito como:  $|\Psi\rangle = a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle$ .

$$I \otimes X = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 \cdot X & 1 \cdot X \\ 1 \cdot X & 0 \cdot X \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (2.5)$$

A equação 2.6 mostra duas propriedades do produto tensorial: (1) não distributividade com relação a multiplicação por escalares e (2) distributividade com relação à soma vetorial, sendo  $z$  um escalar,  $v$  e  $w$  vetores (pertencentes a espaços V e W diferentes).

$$\left\{ \begin{array}{l} (1) \quad z(|v\rangle \otimes |w\rangle) = z|v\rangle \otimes |w\rangle = |v\rangle \otimes z|w\rangle \\ (2) \quad (|v_1\rangle + |v_2\rangle) \otimes |w\rangle = |v_1\rangle \otimes |w\rangle + |v_2\rangle \otimes |w\rangle, \text{ ou} \\ \quad |v\rangle \otimes (|w_1\rangle + |w_2\rangle) = |v\rangle \otimes |w_1\rangle + |v\rangle \otimes |w_2\rangle. \end{array} \right. \quad (2.6)$$

### 2.1.2 Efeitos Quânticos

Além dos postulados, alguns outros conceitos são de fundamental importância para o entendimento da Computação Quântica. Estes conceitos podem ser vistos como ferramentas que

dão à CQ melhorias computacionais não antes disponíveis nos sistemas físicos clássicos.

## Superposição

A superposição ou sobreposição é talvez o mais conhecido efeito do modelo quântico. É a propriedade que os qubits possuem de registrar ao mesmo tempo diferentes valores. Este conceito é equivalente à definição do estado  $|\Psi\rangle$  definido no Postulado 1 onde o mesmo é descrito pela combinação linear dos vetores da base.

Um operador particularmente importante na descrição deste conceito é o operador de Hadamard. A matriz que define este operador é mostrada na equação 2.7 (1), assim como sua representação na notação de Dirac (2).

$$(1) \quad H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (2.7)$$

$$(2) \quad H = \frac{1}{\sqrt{2}}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|)$$

A aplicação do Haddamard nos vetores da base computacional cria um estado em superposição com leitura dos observáveis igualmente provável ( $p = 1/2$ ), como mostrado na equação 2.8.

$$\begin{aligned} (1) \quad H |0\rangle &= \frac{1}{\sqrt{2}}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|) |0\rangle \\ &= \frac{1}{\sqrt{2}}(|0\rangle\langle 0|0\rangle + |0\rangle\langle 1|0\rangle + |1\rangle\langle 0|0\rangle - |1\rangle\langle 1|0\rangle) \\ &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle \\ (2) \quad H |1\rangle &= \frac{1}{\sqrt{2}}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|) |1\rangle \\ &= \frac{1}{\sqrt{2}}(|0\rangle\langle 0|1\rangle + |0\rangle\langle 1|1\rangle + |1\rangle\langle 0|1\rangle - |1\rangle\langle 1|1\rangle) \\ &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle \end{aligned} \quad (2.8)$$

De forma geral, se um sistema quântico pode estar em um dentre  $n$  estados  $|\Psi_1\rangle, |\Psi_2\rangle, \dots, |\Psi_n\rangle$ , este mesmo sistema está em um estado superposto descrito por:

$$|\Psi\rangle = \sum_{i=0}^{n-1} a_i |\Psi_i\rangle \quad (2.9)$$

## Paralelismo

O paralelismo é uma consequência direta da superposição quântica. Como o sistema evolui através de operações unitárias (vide Postulado 2), todas as componentes superpostas do es-

tado em questão são modificadas ao mesmo tempo, como se fossem argumentos individuais para a função computada pela operação. Um exemplo pode ser dado com a aplicação da transformação  $X$ , equivalente ao NOT clássico, em um sistema de um qubit na base computacional.

$$\begin{aligned}
 |\Psi_1\rangle &= |1\rangle && \text{(um qubit no estado 1)} \\
 |\Psi_2\rangle &= H|1\rangle && \text{(aplicação da porta Hadamard)} \\
 &= |-\rangle && \text{(estado sobreposto)} \\
 |\Psi_3\rangle &= X\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) && \text{(aplicação do NOT)} \\
 &= \frac{1}{\sqrt{2}}(|1\rangle - |0\rangle) && \text{(bits invertidos pelo NOT)}.
 \end{aligned} \tag{2.10}$$

A questão do paralelismo possui uma contra-partida importante. Quanto mais componentes superpostos num estado, maior a capacidade de computação paralela e menor a probabilidade de leitura de um determinado valor superposto.

### Emaranhamento

O emaranhamento é uma característica relacionada ao Postulado 4. Os sistemas quânticos quando combinados possuem a característica de não poderem mais ser descritos como dois espaços, mas sim, como um único espaço maior. Por estarem emaranhados, a leitura em um dos sistemas (e.g. qubit) leva os outros a um *colapso compartilhado*.

Um exemplo pode ser dado com a utilização de dois qubits. O circuito da figura 2.2 recebe como entrada dois qubits  $|0\rangle$  e seu estado inicial é descrito por  $|\Psi_0\rangle = |0\rangle \otimes |0\rangle = |00\rangle$ . Quando o primeiro qubit é posto em superposição através da porta Hadamard, o estado passa a ser  $|\Psi_1\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$ , pois o segundo qubit não foi alterado. No terceiro passo aplica-se uma porta conhecida como CNOT, ou NOT controlado, significando que o segundo qubit será invertido apenas se o primeiro for igual a 1. Esta operação leva o sistema ao estado  $|\Psi_2\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ . A leitura do segundo qubit no estado  $|\Psi_2\rangle$  retorna 0 ou 1, com uma probabilidade de  $p(0) = p(1) = (1/\sqrt{2})^2 = 1/2$ , levando o primeiro qubit a colapsar precisamente para o mesmo valor obtido na leitura, demonstrando o emaranhamento.

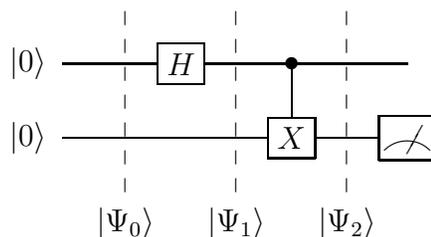


Figura 2.2: Circuito que cria emaranhamento entre 2 qubits, onde a leitura do segundo define o valor do primeiro.

## 2.2 Computação

A Teoria da Computação (TC) tem como principal linha de pesquisa os modelos que descrevem a tarefa de computar. Deste estudo surgem caracterizações de tais tarefas como por exemplo a sua análise de complexidade. Esta análise busca definir em termos gerais, quanto recurso consome uma determinada solução para um problema. Por exemplo: Quanta memória é utilizada para realizar uma soma de dois números inteiros? Quanto tempo é necessário para ordenar um vetor de  $n$  elementos?

A caracterização dos problemas de acordo com suas soluções mais eficientes deu origem às classes de complexidade. Para definir uma classe de complexidade é necessário fixar o modelo computacional utilizado e o recurso analisado. Uma classe fundamental para a computação é a classe **P**. Esta inclui todos os problemas que possuem solução em tempo polinomial numa máquina determinística. Por tempo polinomial deve-se entender que o consumo de tempo pelo algoritmo é expresso por uma função polinomial, como:  $n^2$ ,  $n^3$ ,  $2n^2$ ,  $5n^3 + 7$  (onde  $n$  é o tamanho da entrada do problema). Neste caso, o importante da função é a magnitude de seu crescimento, e por isso a função  $n^2$  pode ser dita da mesma ordem de  $2n^2$ . Formalmente escreve-se:  $5n^3 + 7 \equiv O(n^3)$ .

Assim como o crescimento polinomial, existem problemas em classes que definem um crescimento linear, exponencial, fatorial, etc. Para fins deste trabalho, a principal informação neste contexto é a comparação entre soluções polinomiais e exponenciais já que, na TC clássica, estas definem os problemas como tratáveis e intratáveis, respectivamente. O estudo da Computação Quântica prova que problemas até então intratáveis em modelos computacionais clássicos, podem ter uma solução tratável no modelo quântico.

Nas próximas seções deste capítulo serão introduzidos os algoritmos quânticos, com ênfase no algoritmo de busca, utilizado como principal ferramenta deste trabalho.

### 2.2.1 Algoritmos Quânticos

Algoritmos quânticos são aqueles que fazem uso de algumas das características citadas previamente neste capítulo para computar uma tarefa. O estudo destes algoritmos teve início apenas na década de 1980 com resultados comparativos entre máquinas de Turing (MT) determinísticas, probabilísticas e não-determinísticas. Deutsch e Jozsa [Deu85], [Joz91], [DJ92] mostraram resultados com funções que eram computadas mais rapidamente em uma MT quântica (não-determinística), porém os ganhos não foram tão representativos pois algoritmos probabilísticos também realizavam a tarefa facilmente. Outros trabalhos também analisaram os modelos computacionais quânticos como [BV97], [BB92], [Sim94] fundamentando os estudos das classes de complexidade quânticas.

Apenas no ano de 1994, Shor [Sho94] criou o primeiro algoritmo significativo na área: um algoritmo polinomial para fatorar números grandes. Como a segurança computacional dos dias atuais é baseada na dificuldade de fatoração, o algoritmo de Shor aumentou o interesse pela CQ. No ano de 1996, um outro algoritmo foi descrito por Grover [Gro96] para realizar a busca de elementos dentro de um conjunto desordenado, apresentando um ganho quadrático sobre o algoritmo clássico mais eficiente.

Até os dias atuais, a *Transformada Quântica de Fourier* utilizada por Shor em seu famoso algoritmo de fatoração, e o *Algoritmo Quântico de Busca* de Grover têm sido as bases para o desenvolvimento dos algoritmos quânticos evidenciando as potencialidades de uma máquina quântica. Neste trabalho, o algoritmo de Grover será utilizado em duas situações distintas: a primeira aplicada à teoria dos códigos de bloco lineares, descrita no capítulo 3, e a segunda relacionada com geradores pseudo-aleatórios, descrita no capítulo 4. Uma introdução ao mesmo será feita na próxima seção.

## 2.3 Ferramenta: Busca Quântica

O Algoritmo Quântico de Busca, ou simplesmente Algoritmo de Grover [Gro96], foi escrito por L.K. Grover no ano de 1996. O melhor algoritmo clássico resolve a busca de um elemento em uma estrutura desordenada em tempo  $O(N)$  (onde  $N$  é a quantidade de elementos do conjunto de entrada), enquanto a solução apresentada por Grover encontra o elemento em tempo  $O(\sqrt{N})$ . Este algoritmo quadrático tem sido utilizado por alguns traba-

lhos desde sua publicação incluindo [LLM05] e parte dos resultados já publicados [OA06; ONA07], descritos no capítulo 3 e 4 respectivamente.

O Algoritmo de Grover está dividido em três etapas. A primeira delas, bastante comum na Computação Quântica, é a preparação dos qubits utilizados colocando-os em superposição. A segunda etapa deve marcar o elemento buscado através de uma operação unitária denominada *oráculo*. Por fim, na terceira etapa, é aplicada uma operação conhecida como *amplificação de amplitude*, que deve aumentar a probabilidade de leitura do elemento anteriormente marcado.

O circuito da figura 2.3 é dividido nas três partes do algoritmo citadas acima. Considera-se a existência de dois registradores: o primeiro possui  $n = \log N$  qubits, todos iniciados com o valor  $|0\rangle$ ; e o segundo possui um único qubit auxiliar iniciado com o valor  $|1\rangle$ . Dados os valores dos registradores é possível escrever o estado inicial do sistema como:  $|\Psi\rangle = |0\dots 01\rangle$ .

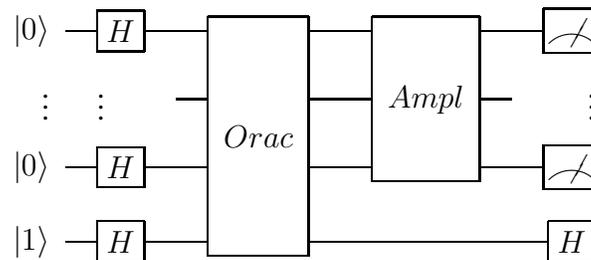


Figura 2.3: Estrutura de um circuito que executa uma iteração do algoritmo de Grover.

### 2.3.1 Primeira Etapa

O primeiro estágio do algoritmo é a preparação dos qubits utilizando portas Hadamard. Esta porta coloca os qubit em superposição, com uma amplitude de probabilidade igualmente distribuída para cada estado da base do espaço (vide seção 2.1.2). A ação desta operação pode ser vista na equação 2.11, onde as três últimas igualdades são formas diferentes de notação para o mesmo resultado.

$$\begin{aligned}
|\Psi_1\rangle &= H^{\otimes n+1} |\Psi\rangle \\
&= H^{\otimes n} |0\rangle^{\otimes n} H |1\rangle \\
&= \left[ \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \right] \left[ \frac{|0\rangle - |1\rangle}{\sqrt{2}} \right] \\
&= \left[ \left( \frac{|0\rangle + |1\rangle}{\sqrt{2}} \right)^{\otimes n} \right] \left[ \frac{|0\rangle - |1\rangle}{\sqrt{2}} \right] \\
&= |+\rangle^{\otimes n} |-\rangle
\end{aligned} \tag{2.11}$$

### 2.3.2 Segunda Etapa

O segundo estágio do algoritmo refere-se à aplicação do oráculo (porta *Orac*). Este é um operador unitário que utiliza qubits auxiliares para marcar o elemento procurado. A equação 2.12 mostra como uma operação unitária computa uma função  $f$  qualquer, colocando o resultado no segundo qubit (auxiliar).

$$U_f(|i\rangle |j\rangle) = |i\rangle |j \oplus f(i)\rangle. \tag{2.12}$$

No caso estudado,  $U_f$  representa o oráculo, o  $|i\rangle$  representa a entrada da função (superposição dos elementos da estrutura desordenada) e o  $|j\rangle$  é um qubit utilizado para marcar o elemento. A equação 2.13 define a função a ser implementada pelo oráculo.

$$f(i) = \begin{cases} 1, & \text{se } i \text{ é o elemento procurado} \\ 0, & \text{caso contrário} \end{cases}. \tag{2.13}$$

A aplicação dos resultados exibidos na equação 2.12 sobre o estado  $|\Psi_1\rangle$  do circuito pode ser vista na equação 2.14.

$$\begin{aligned}
|\Psi_2\rangle &= U_f(|\Psi_1\rangle) \\
&= U_f(|+\rangle^{\otimes n} |-\rangle) \\
&= \text{(chamando o primeiro registrador de } |i\rangle \text{ escreve-se)} \\
&= U_f(|i\rangle |-\rangle) \\
&= U_f \left( \frac{|i\rangle|0\rangle - |i\rangle|1\rangle}{\sqrt{2}} \right) \\
&= \frac{U_f(|i\rangle|0\rangle) - U_f(|i\rangle|1\rangle)}{\sqrt{2}} \\
&= \frac{1}{\sqrt{2}} (|i\rangle |f(i)\rangle - |i\rangle |1 \oplus f(i)\rangle) \\
&= |i\rangle \left( \frac{|f(i)\rangle - |1 \oplus f(i)\rangle}{\sqrt{2}} \right)
\end{aligned} \tag{2.14}$$

Para finalizar a aplicação do oráculo resta considerar as possibilidades de valor para a função  $f(i) = \{0, 1\}$  (vide equação 2.15). O resultado final do estado  $|\Psi_2\rangle$  mostra que, apenas os elementos que satisfazem a função computada por  $U_f$  recebem uma fase, ou seja, o elemento buscado será marcado com um sinal negativo.

$$\begin{aligned} |\Psi_2\rangle &= \begin{cases} |i\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = |i\rangle |-\rangle, & \text{Se } f(i) = 0 \\ |i\rangle \left(\frac{|1\rangle - |0\rangle}{\sqrt{2}}\right) = |i\rangle (-1) |-\rangle, & \text{Se } f(i) = 1 \end{cases} \\ &= (-1)^{f(i)} |i\rangle |-\rangle \end{aligned} \quad (2.15)$$

É possível fazer uma interpretação geométrica deste algoritmo. Como mostra a primeira parte da figura 2.4 o estado do sistema pode ser reescrito utilizando uma nova base:  $|\beta\rangle$  representa o vetor que define o elemento procurado pelo oráculo e  $|\alpha\rangle$  o seu vetor ortogonal. Normalmente, nesta nova base, o vetor  $|\Psi^*\rangle = a|\alpha\rangle + b|\beta\rangle$  estará mais próximo do vetor  $|\alpha\rangle$  pois existem muito mais elementos que não satisfazem o oráculo que o contrário. Na mesma figura é mostrado o estado após a ação do oráculo, que nos moldes acima definidos, representa sempre uma reflexão do estado inicial em relação ao vetor  $|\alpha\rangle$ , pois  $|\Psi_1^*\rangle = a|\alpha\rangle - b|\beta\rangle$ .

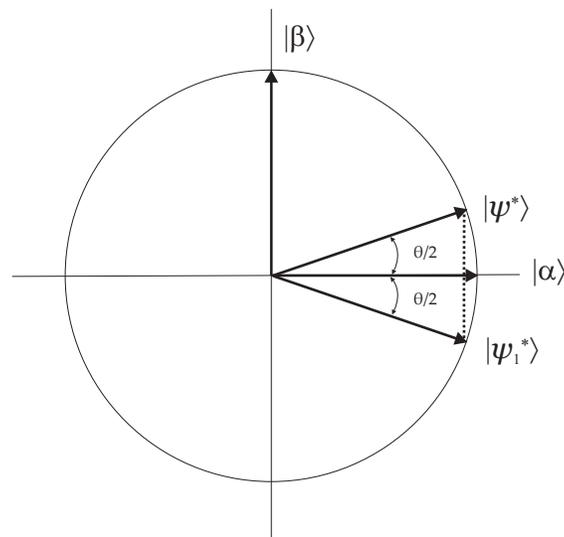


Figura 2.4: A evolução do sistema para o estado  $|\Psi_1^*\rangle = \text{Orac}|\Psi^*\rangle$ .

### 2.3.3 Terceira Etapa

A operação unitária  $Ampl = (2|\Psi\rangle\langle\Psi| - I)$  foi introduzida por Grover em [Gro96], e sua aplicação em um estado com elementos previamente marcados, acarreta na amplificação da amplitude [BHMT00] de tais elementos. A figura 2.5 mostra a reflexão, em relação a  $|\Psi^*\rangle$ , executada por este operador. A composição das reflexões impostas pelos operadores  $Orac$  e  $Ampl$  é conhecida como **iteração de Grover**. O resultado de uma aplicação desta iteração pode ser vista como a rotação do estado  $|\Psi^*\rangle$  num ângulo de  $\theta$  para mais próximo do vetor  $|\beta\rangle$ .

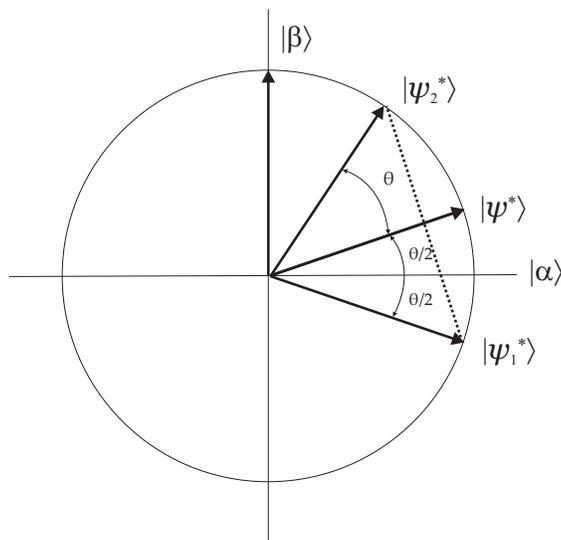


Figura 2.5: A evolução do sistema para o estado  $|\Psi_2^*\rangle = Ampl|\Psi_1^*\rangle$ .

### 2.3.4 Leitura e Desempenho

Como visto no Postulado 3, a medição dos qubits retornará um observável (0 ou 1) com uma determinada probabilidade. A iteração de Grover, por haver levado o sistema para mais próximo de  $|\beta\rangle$ , aumenta a probabilidade de leitura do elemento buscado, porém em muitos casos a aplicação de sucessivas iterações de Grover pode levar a uma probabilidade ainda maior. Mas existe um limite máximo de aproximação entre os vetores  $|\Psi^*\rangle$  e  $|\beta\rangle$ .

Para demonstrar esse limite de aplicações da iteração (rotação) de Grover, reescreve-se o estado do sistema considerando a possibilidade do oráculo marcar  $\mathbf{M}$  elementos. Sendo  $\mathbf{N}$  o número total de elementos do conjuntos, define-se:

- $\sum'_x$  indica a soma dos  $\mathbf{M}$  elementos que *são* solução para a busca; e
- $\sum''_x$  no caso contrário ( $\mathbf{N-M}$  elementos).

Usando estas definições os vetores que formam a base do espaço são escritos:

- $|\alpha\rangle = \frac{1}{\sqrt{N-M}} \sum''_x |x\rangle$ ; e
- $|\beta\rangle = \frac{1}{\sqrt{M}} \sum'_x |x\rangle$ .

Considerando esta definição dos vetores da base, o estado inicial é definido por:  $|\Psi^*\rangle = \sqrt{\frac{N-M}{N}} |\alpha\rangle + \sqrt{\frac{M}{N}} |\beta\rangle$ . Como mostrado na figura 2.5, cada iteração de Grover, composta de duas reflexões, causa uma rotação de  $\theta$  graus em direção ao vetor  $|\beta\rangle$ . Desta forma, a rotação do vetor  $|\Psi^*\rangle$  de  $\arccos(\sqrt{M/N})$  radianos leva o sistema a uma aproximação máxima de  $|\beta\rangle$  definindo um valor para  $R$  como mostrado na equação 2.16.

$$R = \text{IMP} \left( \frac{\arccos \sqrt{M/N}}{\theta} \right) \text{ onde, IMP significa Inteiro Mais Próximo.} \quad (2.16)$$

Como  $\arccos \sqrt{M/N} \leq \pi/2$  a equação 2.16 pode ser reescrita como:

$$R \leq \left\lceil \frac{\pi}{2\theta} \right\rceil \quad (2.17)$$

Partindo da equação 2.17 pode-se criar um limite superior para  $R$  dado um limite inferior para  $\theta$ , já que os mesmos são inversamente proporcionais. Este limite pode ser estipulado dada a relação trigonométrica:  $\frac{\theta}{2} \geq \sin \frac{\theta}{2}$ . A equação 2.18 reescreve a equação 2.17 utilizando a relação e mostra que o algoritmo executa em  $O(\sqrt{N/M})$  passos, provando um ganho quadrático com relação à complexidade  $O(N/M)$  do caso clássico.

$$\sin \frac{\theta}{2} = \sqrt{\frac{M}{N}} \implies R \leq \left\lceil \frac{\pi}{4} \sqrt{\frac{N}{M}} \right\rceil \quad (2.18)$$

## 2.4 Ferramenta: Contagem Quântica

O Algoritmo de Grover apresentado na seção 2.3 pode ser visto como um caso especial de um processo mais geral denominado *Amplificação de Amplitude*. Da mesma forma, a ferramenta descrita nesta seção pode ser considerada um caso especial de um processo conhecido como

*Estimação de Amplitude.* A Contagem Quântica [BHT98] resulta da combinação de dois dos principais resultados da CQ: a busca quântica e a Transformada Quântica de Fourier (QFT) [Sho94].

### 2.4.1 Introdução

Uma transformação é uma função que age sobre um conjunto de dados de entrada e gera um conjunto transformado destes dados. Poucas são as transformadas que possuem uma usabilidade tão alta como a transformada de Fourier, largamente utilizada nas áreas de teoria dos números, análise combinatória, processamento de imagens e sinais, criptografia, entre outras. A QFT é uma implementação quântica análoga à sua versão clássica, cuja ação num estado arbitrário é descrita por:

$$\sum_{j=0}^{N-1} x_j |j\rangle \longrightarrow \sum_{k=0}^{N-1} y_k |k\rangle \quad (2.19)$$

onde a amplitude  $y_k$  é a transformada discreta de Fourier da amplitude  $x_j$ .

A descrição detalhada da QFT será omitida por uma questão de simplicidade, mas outras informações podem ser encontradas em [NC00] capítulo 5. Para fins de entendimento da contagem quântica é suficiente saber que ao aplicar esta transformação em um estado  $|j\rangle$ , o mesmo será decomposto em estados da base de forma bem peculiar. A equação 2.20 mostra tal decomposição utilizando duas notações: o valor decimal do estado  $|j\rangle$  é reescrito na forma binária  $|j_1, \dots, j_n\rangle$ , e a fração binária  $j_l/2 + j_{l+1}/4 + \dots + j_m/2^{m-l+1}$  é representada por  $0.j_l j_{l+1} \dots j_m$ .

$$|j_1, \dots, j_n\rangle \rightarrow \frac{(|0\rangle + e^{(2\pi i)(0.j_n)} |1\rangle)(|0\rangle + e^{(2\pi i)(0.j_{n-1}j_n)} |1\rangle) \dots (|0\rangle + e^{(2\pi i)(0.j_1 j_2 \dots j_n)} |1\rangle)}{2^{n/2}} \quad (2.20)$$

É importante perceber que um valor transformado de  $j$  foi escrito na fase do estado  $|1\rangle$  da base computacional. Como a entrada foi representada em binário, a saída é composta por vários qubits onde a fase de cada um dos estados  $|1\rangle$  carrega parte do resultado. Apesar de parecerem demasiado complexos, os resultados do cálculo desta equação são facilmente compreendidos. Na equação 2.21 é mostrado o cálculo da transformada de Fourier sobre os

estados da base computacional para  $n = 1$ ,  $\{|0\rangle, |1\rangle\}$ , cujo resultado equivale à aplicação da porta Hadamard.

$$\begin{aligned}
|0\rangle &\rightarrow \frac{(|0\rangle + e^{2\pi i(0.0)} |1\rangle)}{2^{1/2}} & (2.21) \\
&\rightarrow \frac{(|0\rangle + e^{2\pi i(0/2)} |1\rangle)}{\sqrt{2}} \\
&\rightarrow \frac{(|0\rangle + |1\rangle)}{\sqrt{2}} \\
|1\rangle &\rightarrow \frac{(|0\rangle + e^{2\pi i(0.1)} |1\rangle)}{\sqrt{2}} \\
&\rightarrow \frac{(|0\rangle + e^{2\pi i(1/2)} |1\rangle)}{\sqrt{2}} \\
&\rightarrow \frac{(|0\rangle - |1\rangle)}{\sqrt{2}}
\end{aligned}$$

Já na equação 2.22, o cálculo da transformada é realizado sobre um dos estados da base para  $n = 2$ ,  $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ , para que a utilização das frações binárias fique mais clara.

$$\begin{aligned}
|01\rangle &\rightarrow \frac{(|0\rangle + e^{2\pi i(0.1)} |1\rangle)(|0\rangle + e^{2\pi i(0.01)} |1\rangle)}{2^{2/2}} & (2.22) \\
&\rightarrow \frac{(|0\rangle + e^{2\pi i(1/2)} |1\rangle)(|0\rangle + e^{2\pi i(0/2+1/4)} |1\rangle)}{2} \\
&\rightarrow \frac{(|0\rangle + e^{\pi i} |1\rangle)(|0\rangle + e^{\pi i(1/2)} |1\rangle)}{2} \\
&\rightarrow \frac{(|0\rangle - |1\rangle)(|0\rangle + i |1\rangle)}{2} \\
&\rightarrow \frac{(|00\rangle + i |01\rangle + |10\rangle - i |11\rangle)}{2}
\end{aligned}$$

### 2.4.2 Estimação de Fase

Seja  $U$  uma operação unitária com autovetor  $|u\rangle$  e autovalor  $e^{2\pi i\phi}$ . O objetivo do algoritmo de estimação de fase é estimar o valor de  $\phi$ . A solução apresentada para este problema depende da existência de um oráculo capaz de preparar o autovetor  $|u\rangle$  para que seja utilizado como entrada do algoritmo. A figura 2.6 mostra o circuito geral que soluciona o problema proposto.

Este circuito utiliza dois registradores compostos por vários qubits e é executado em duas etapas.

O primeiro registrador, utilizado para guardar o valor binário de  $\phi$ , é iniciado com  $|0\rangle$  em todos os seus  $t$  qubits. A quantidade  $t$  depende da precisão desejada para o valor da fase buscada. É provado em [NC00] (seção 5.2.1) que para uma aproximação do valor de  $\phi$  com precisão  $2^{-m}$  são necessários

$$t = m + \left\lceil \log \left( 2 + \frac{1}{2\epsilon} \right) \right\rceil \quad (2.23)$$

com probabilidade de sucesso de ao menos  $1 - \epsilon$ . O segundo registrador possui tantos qubits quantos forem necessários para registrar o autovetor  $|u\rangle$  previamente preparado.

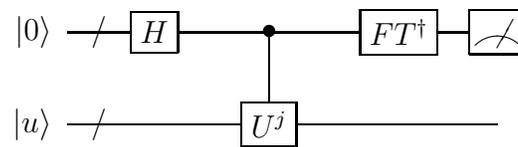


Figura 2.6: Esquema geral do circuito que implementa o algoritmo quântico de contagem.

### A primeira fase

A primeira fase do algoritmo (aplicação da porta  $U^j$ ) é responsável pela construção do valor  $\phi$  no primeiro registrador. É necessário ressaltar que a aplicação de uma operação unitária  $U$  qualquer ao seu autovetor  $|u\rangle$  resulta em uma fase do tipo  $e^{i\theta}$ . A equação 2.24 mostra a evolução de um sistema similar ao da figura 2.7 até a aplicação de  $U$ , mas utilizando apenas um qubit no primeiro registrador.

$$\begin{aligned} |\Psi_1\rangle &= |0\rangle |u\rangle \\ |\Psi_2\rangle &= H |0\rangle I |u\rangle = (|0\rangle |u\rangle + |1\rangle |u\rangle) / \sqrt{2} \\ |\Psi_3\rangle &= U(|\Psi_2\rangle) = (|0\rangle |u\rangle + e^{i\theta} |1\rangle |u\rangle) / \sqrt{2} \\ &= (|0\rangle + e^{i\theta} |1\rangle) |u\rangle / \sqrt{2} \end{aligned} \quad (2.24)$$

O circuito da figura 2.7 implementa detalhadamente a primeira fase do algoritmo. O resultado de cada qubit do primeiro registrador neste circuito baseia-se nos resultados da

equação 2.24 fazendo-se  $\theta = 2\pi i(2^z \phi)$  (onde  $0 \leq z \leq t-1$ ). O estado do primeiro registrador após a aplicação deste circuito está descrito na equação 2.25.

$$\frac{1}{2^{t/2}} (|0\rangle + e^{2\pi i(2^{t-1}\phi)} |1\rangle)(|0\rangle + e^{2\pi i(2^{t-2}\phi)} |1\rangle) \dots (|0\rangle + e^{2\pi i(2^0\phi)} |1\rangle) \quad (2.25)$$

$$= \frac{1}{2^{t/2}} \sum_{k=0}^{2^t-1} e^{2\pi i\phi k} |k\rangle.$$

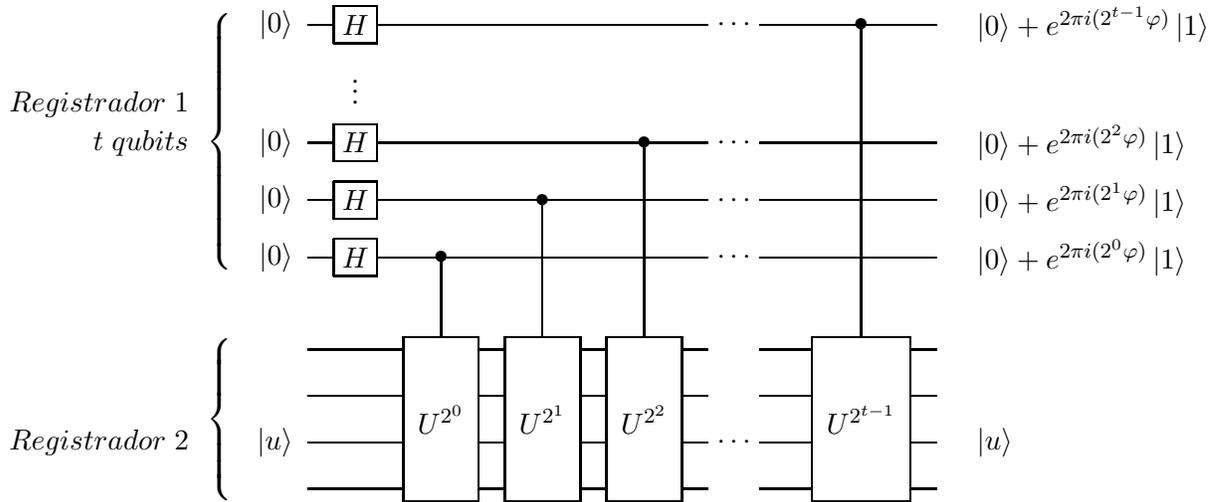


Figura 2.7: Implementação detalhada da primeira fase do algoritmo de contagem.

### A segunda fase

A segunda fase do algoritmo é a aplicação da QFT inversa ( $QFT^\dagger$ ) no primeiro registrador seguido da medição do mesmo. Uma comparação do estado descrito na equação 2.25 e o resultado da aplicação de QFT na equação 2.20 mostra o porquê do uso da inversa de QFT: a equação 2.25 é exatamente a transformação de  $\phi$  decomposta em valores binários. Deste modo, a aplicação da  $QFT^\dagger$  resultará em uma estimativa de  $\phi$  com  $t$  bits de precisão no primeiro registrador.

### 2.4.3 Definição do Algoritmo da Contagem Quântica

O algoritmo de contagem quântica é um caso particular do procedimento de estimativa de fase descrito na subseção anterior. O circuito da figura 2.8 implementa a contagem quântica,

evidenciando a utilização do circuito referente à estimativa de fase (vide figura 2.6), onde as portas  $U^j$  são substituídas pela iteração de Grover ( $G$ ), já definida na seção 2.3.

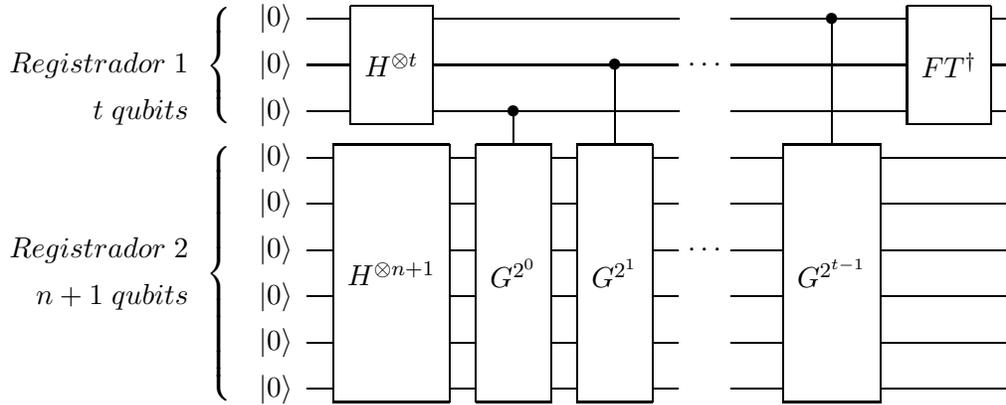


Figura 2.8: Esquema geral do circuito da contagem quântica de soluções.

Para melhor entender a solução apresentada, faz-se necessário contextualizar o uso do operador  $G$  como substituto de  $U^j$  no algoritmo de estimativa de fase. O estado de um sistema que receberá a aplicação de  $G$  pode ser escrito como  $|\Psi_0\rangle = a|\alpha\rangle + b|\beta\rangle$ , onde a primeira componente ( $|\alpha\rangle$ ) reúne todos os  $N - M$  elementos que não satisfazem o oráculo e a segunda ( $|\beta\rangle$ ), os  $M$  elementos que serão marcados pelo mesmo. O estado  $|\Psi_0\rangle$  pode ser reescrito em termos do ângulo  $\theta$ , pelo qual o estado será rotacionado a cada aplicação da amplificação de fase:  $|\Psi_0\rangle = e^{i(2\pi-\theta)}|\alpha\rangle + 2^{\theta i}|\beta\rangle$ .

Desta forma, ao submeter o estado  $|\Psi_0\rangle$  ao circuito da contagem, o mesmo irá estimar o valor do ângulo  $\theta$ , sendo possível então utilizá-lo na equação  $\sin^2(\theta/2) = M/N$  para calcular  $M$ .

## 2.4.4 Conclusões

Duas conclusões sobre o algoritmo da contagem quântica são destacadas nesta subseção e posteriormente serão utilizadas no capítulo 3. A primeira delas está relacionada à quantidade de qubits e de execuções de  $G$  requeridos pela contagem quântica. A quantidade de qubits depende do valor de  $t$  estimado, que será o número de qubits necessários para codificar os  $N$  elementos do espaço de busca. Para um  $m = \lceil n/2 \rceil + 1$  e  $\epsilon = 1/6$ , mostra-se que o algoritmo da contagem utilizaria  $t = \lceil n/2 \rceil + 3$  qubits no primeiro registrador, e faria chamadas ao oráculo na ordem de  $\Theta(\sqrt{N})$  vezes (vide [NC00] - seção 6.3). No capítulo 3, a análise de

desempenho da solução proposta fará uso destes valores.

O segundo ponto em destaque é que a contagem quântica pode ser diretamente utilizada tanto para descobrir a existência de um determinado valor no espaço de busca, quanto para auxiliar na execução correta do algoritmo de Grover. O algoritmo de busca pode ser utilizado apenas quando há ao menos uma solução a ser procurada pois caso contrário, nenhum elemento será marcado e assim amplificado, fazendo o algoritmo retornar um resultado incorreto. Com a contagem quântica, basta verificar que o valor de  $M$  calculado é maior que zero para atestar a existência de solução. O cálculo de quantas iterações de Grover são necessárias para encontrar o valor correto buscado também depende de  $M$ , assim, para um dado problema onde a quantidade de soluções é desconhecida, a contagem quântica pode ser utilizada primeiramente para estimar o valor de  $M$  e só então aplicar a busca quântica para encontrar tais elementos.

# Capítulo 3

## Códigos para Controle de Erros

A *Teoria da Informação* (TI) é uma área da Matemática Aplicada que estuda a quantificação de dados visando maximização e confiabilidade na transmissão e armazenamento de informação. Ao utilizar qualquer meio de comunicação ou armazenamento computacional faz-se necessária a consideração dos possíveis erros inerentes aos meios físicos. Estes meios são denominados canais de informação e para lidar com a ocorrência de erros são utilizados códigos para a detecção e correção dos mesmos.

Inicialmente serão apresentados conceitos introdutórios da área de codificação de informação. Na segunda parte deste capítulo serão apresentados alguns resultados obtidos com a utilização do algoritmo quântico de busca (vide seção 2.3) na montagem de um ambiente para teste de códigos de bloco lineares.

### 3.1 Canais

Um canal é qualquer meio físico que possua entrada e saída de informação. Alguns exemplos cotidianos de canais são os cabos telefônicos, mídias digitais como CDs e DVDs e o próprio meio ambiente. Por tramitar por meios físicos e imperfeitos, a informação pode ser corrompida mediante ocorrências indesejadas, como por exemplo: eventuais interferências eletromagnéticas sobre os cabos telefônicos e ondas de rádio; e erros de refração e reflexão ótica da superfície metálica dos discos.

Para a definição de um canal na TI, é indiferente a natureza física ou o tipo de uso (i.e. transmitir ou armazenar), sendo fundamental a formalização da inserção e leitura dos

dados e o tipo de erro possível. Matematicamente o modelo de um canal é descrito por um conjunto de informação de entrada, um conjunto de informação saída e as imperfeições são incorporadas como erros probabilísticos que alteram as informações que passam pelo canal.

No exemplo da figura 3.1, o conjunto de entrada é definido como sendo símbolos de um alfabeto binário  $\mathcal{A} = \{0, 1\}$ , e ao conjunto de saída adiciona-se apenas um símbolo sinalizador de erro ( $e$ ). O formato do erro é definido pela probabilidade  $p$  de qualquer um dos símbolos ser substituído pelo símbolo de erro  $e$ . Este tipo de erro possui a propriedade de que o local do erro é definido diretamente por um símbolo do alfabeto de saída. Em outros casos, o local do erro não é definido diretamente como acontece quando o erro é equivalente à troca entre símbolos do alfabeto de entrada.

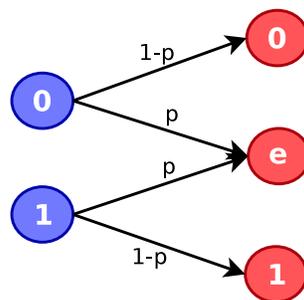


Figura 3.1: Um canal de alfabeto binário e erro de apagamento.

Com a eventualidade dos erros, o estudo dos canais de informação depara-se com a necessidade de detectar ou mesmo corrigi-los. Para isso a Teoria da Informação faz uso de *Códigos para Controle de Erros* (CCE). Estes códigos variam desde a mera repetição ou retransmissão da informação, até sofisticados algoritmos matemáticos. Porém, a base para qualquer CCE é a adição de informação redundante aos dados inseridos no canal.

A figura 3.2 mostra a sequência de passos necessários para o uso eficaz de um canal qualquer. Após a fonte gerar a informação a ser enviada, um processo denominado *codificação* adiciona, de forma organizada, informação redundante para então utilizar o canal. A informação na saída do canal pode estar danificada e um processo denominado *decodificação* é utilizado para tentar identificar erros e recuperar a informação original, para assim disponibilizá-la ao receptor. Os processos de de/codificação a serem utilizados estão intrinsecamente ligados ao tipo de erro inserido pelo canal.

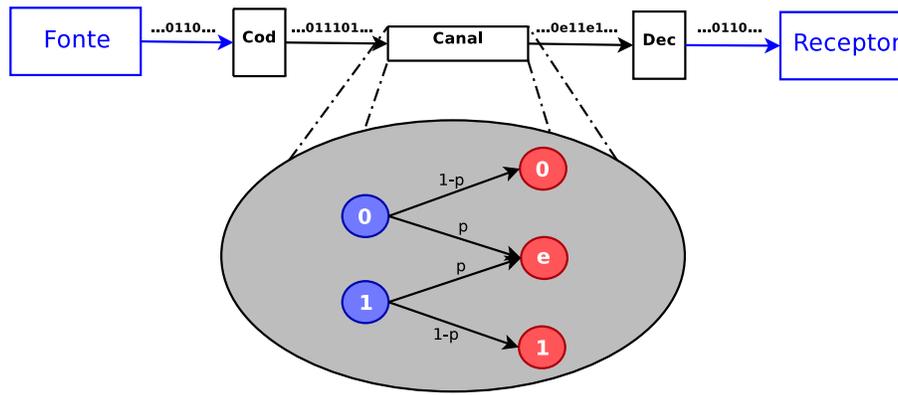


Figura 3.2: Estrutura geral do uso de um canal de informação.

## 3.2 Códigos de Bloco Lineares

Os códigos utilizados nos processos de de/codificação podem ser classificados em dois grupos: convolucionais e blocos lineares. Num código de bloco, objeto de estudo deste capítulo, o processamento da informação é realizado sobre sequências de símbolos de tamanho fixo  $k$  denominadas palavra ou vetor de informação ( $v_i \in \mathcal{A}^*$ ). A palavra codificada a ser transmitida é denominada palavra ou vetor código e é definida como  $v_c = c_1, c_2, \dots, c_k, \dots, c_n$ . Os  $n - k$  símbolos adicionais são denominados “símbolos de paridade”.

Um código  $(n, k)$  é produzido por uma matriz ao ser aplicada sobre os vetores  $v_i$  (de tamanho  $k$ ) para gerar o  $v_c$  correspondente (de tamanho  $n$ ). A matriz geradora  $G$  é construída a partir de  $k$  vetores linearmente independentes, tendo cada um deles o tamanho  $n$ . A matriz

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} \quad (3.1)$$

é geradora de um código  $(5,3)$  pois, além de suas linhas serem linearmente independentes, a multiplicação de vetores de informação de tamanho  $k = 3$  gera o correspondente vetor código de tamanho  $n = 5$ . Um exemplo de codificação pode ser visto na equação 3.2.

$$v_c = v_i G = (101) \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} = (10101) \quad (3.2)$$

Uma outra matriz deve ser considerada neste contexto: a matriz de teste de paridade  $H$ . Esta é utilizada para checar a pertinência de um determinado  $v_c$  ao código gerado por  $G$  através da equação  $v_c \cdot H^T = (0^*)$ , podendo assim ser a base para a detecção de erros no uso do canal.  $H$  pode ser construída a partir de  $G$  quando esta encontra-se em sua forma sistemática <sup>1</sup>, ou seja  $G = [I|P]$  onde  $I$  é a matriz identidade de tamanho  $k \times k$  e  $P$  é uma matriz de tamanho  $k \times (n - k)$ , que completa as dimensões definidas para  $G$ . Desta forma, a matriz de teste é definido como sendo  $H = [-P^T|I]$ . No exemplo aqui utilizado a matriz geradora (equação 3.1) já está em sua forma sistemática dando origem a matriz  $H$  mostrada na equação 3.3.

$$H = \left[ \begin{array}{ccc|cc} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{array} \right] \quad (3.3)$$

A seguir dois testes de paridade são apresentados. Na equação 3.4 é testado um vetor que pertence ao código, ou seja, o resultado do teste é igual ao vetor nulo.

$$(10101) \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{array} \right] = (1 + 0 + 1 + 0 + 0 \quad 0 + 0 + 1 + 0 + 1) = (00) \quad (3.4)$$

Já na equação 3.5 é utilizado um vetor que não pertence, ou seja, não pode ser gerada pela matriz da equação 3.1.

$$(01010) \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{array} \right] = (0 + 0 + 0 + 1 + 0 \quad 0 + 1 + 0 + 0 + 0) = (11) \quad (3.5)$$

---

<sup>1</sup>Toda matriz geradora possui uma matriz equivalente na forma sistemática obtida através das operações de transformação: permutações de colunas e operações elementares com linhas.

### 3.2.1 Decodificação

Ao tramitar por um canal, a palavra codificada  $v_c$  pode ser modificada. Matematicamente diz-se que o canal adicionou um *vetor erro* ( $e$ ) ao vetor código:  $v_e = v_c + e$ . Foi mostrado anteriormente como identificar se a palavra recebida pertence ou não ao código, mas em muitos casos também é possível recuperar a informação corrompida. No contexto de códigos de bloco lineares a decodificação é vista como a busca da palavra pertencente ao código ( $v_c$ ) “mais próxima” do vetor recebido ( $v_e$ ). O conceito de proximidade aqui utilizado é denominado *distância de Hamming*.

A distância  $d$  é normalmente o terceiro parâmetro de um código, o qual seria descrito por  $(n, k, d)$ . Este parâmetro define quão separadas estão os vetores  $v_c$ . O cálculo da distância de Hamming depende da distância entre duas palavras quaisquer do código. Neste cálculo define-se que duas palavras distam entre si exatamente a quantidade de símbolos que as diferenciam (vide equação 3.6). Além disso define o peso  $w$  de uma palavra como sendo a quantidade de símbolos não nulos da mesma, como por exemplo:  $w(01001) = 2$  ou  $w(11100) = 3$ .

$$d^*((01001), (11100)) = 3 \text{ ou } d^*((10001), (00000)) = 2 \quad (3.6)$$

Partindo destes conceitos de peso e distância das palavras de um código, define-se o cálculo da distância mínima através da fórmula:  $d = \min d^*(v_{c1}, v_{c2})$ , que significa a busca pela menor distância entre quaisquer duas palavras do código. No entanto, como todo código linear possui a palavra nula ( $(0^k)G = (0^n)$ ) e é fechado pela soma, este cálculo pode ser redefinido em termos do peso  $w$  de uma palavra como mostrado na equação 3.7.

$$\begin{aligned} d &= \min d^*(v_{c1}, v_{c2}) \\ &= \min d^*(v_{c1} - v_{c2}, v_{c2} - v_{c2}) \\ &= \min d^*(v_{c3}, (0^n)) \\ &= \min w(v_{c3}) \end{aligned} \quad (3.7)$$

A figura 3.3 mostra a base para a decodificação através da distância mínima do código. Caso duas palavras quaisquer do código estejam separadas por  $d = 2t + 1$ , o código é capaz de recuperar  $t$  erros. Isso ocorre pois cada erro somado ao vetor  $v_c$  pode ser visto como a alteração de um símbolo, distanciando assim o  $v_e$  uma unidade por vez do centro

da esfera definida pelo  $v_i$  e raio  $t$ . Desta forma uma palavra incorreta no formato  $v_e = v_c + e$  sempre será encaixada na esfera definida pela palavra código original se  $w(e) \leq t$ . Este enquadramento define o processo de recuperação da palavra original, demonstra a importância do conhecimento de  $d$  e sua relação com a eficiência de decodificação de um código.

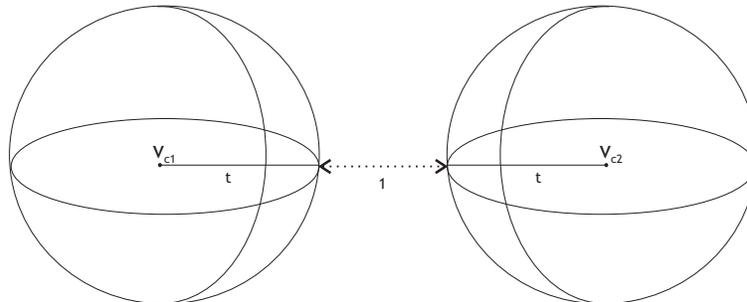


Figura 3.3: Duas palavras do código separadas por uma distância  $d = 2t + 1$ .

Com a definição da equação 3.7, o cálculo desta distância é diretamente proporcional ao número de palavras código. Porém este valor cresce exponencialmente com a quantidade de bits  $k$  da palavra de informação original, tornando-o um processo intratável pela computação clássica (processo com complexidade  $O(2^k)$  no caso do alfabeto binário). No restante deste capítulo será mostrado um algoritmo quântico que pode ser utilizado para tornar o cálculo de  $d$  mais eficiente.

### 3.3 Teste Quântico para Códigos de Bloco Lineares

Esta seção mostrará a utilização do algoritmo da Contagem Quântica, já definido no capítulo 2 - seção 2.4, para o teste de códigos de bloco lineares. Este teste baseia-se na busca da distância mínima de um dado código de bloco linear.

#### 3.3.1 Cálculo da Distância Mínima

Nesta subseção será demonstrado um algoritmo quântico para o cálculo da distância mínima de códigos de bloco lineares. A título de exemplo foi utilizado um Código de Hamming pois possui toda a estrutura matemática exposta anteriormente. Além disso, este tipo de

código define uma maneira simplista de montagem das matrizes  $H$  e  $G$  tornando a distância mínima ( $d$ ) conhecida antecipadamente e igual a 3. Se  $m$  é a quantidade de linhas da matriz  $H$ , a quantidade de colunas é definida pela quantidade de vetores não nulos e linearmente independentes  $n = 2^m - 1$ . Assim, um código de Hamming é definido genericamente como  $(2^m - 1, 2^m - 1 - m, 3)$ .

Um código de Hamming (7,4) será utilizado como exemplo. Sua matriz de teste de paridade foi construída com  $m = 3$ :

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3.8)$$

De  $H$  escreve-se a matriz geradora

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}. \quad (3.9)$$

O algoritmo aqui proposto é apresentado na forma de circuito e ilustrado na figura 3.4. A construção do mesmo baseia-se na matriz geradora do código  $G$  sendo portanto bastante geral.

A entrada do circuito é composta por 4 registradores. O primeiro possui  $k$  qubits que guardam os bits das palavras de informação. O segundo registrador possui  $n - k$  qubits para armazenar os bits das palavras código. O terceiro registrador possui  $\log n$  qubits para guardar o peso das palavras código. Por fim, o último registrador possui  $t$  qubits como definido na seção 2.4, utilizados no algoritmo da contagem quântica. Todos os qubits devem ser preparados no estado  $|0\rangle$ .

Há 4 fases no funcionamento do circuito. A primeira fase é a aplicação da porta Hadamard ao primeiro registrador, criando assim uma superposição de todas as palavras de informação de tamanho  $k$ . A segunda fase (porta  $G'$ ) codifica as palavras código e as guarda no segundo registrador. A terceira fase (porta  $C$ ) calcula o peso de todas as palavras guardando as informações no terceiro registrador. A última fase (porta  $Cont$ ) executa testes de existência de um determinado valor de peso. As portas mostradas abaixo farão uso do exemplo de

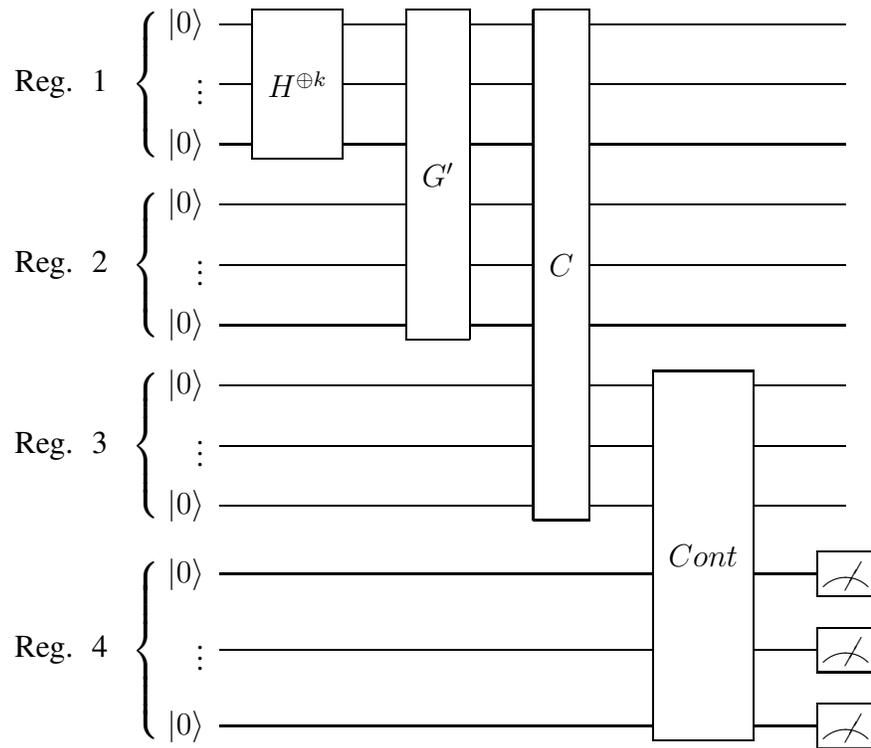


Figura 3.4: Circuito quântico para cálculo de  $d$  em códigos de blocos lineares.

código apresentado na equação 3.9.

### A porta $G'$

Esta porta quântica representa a matriz geradora do código  $\mathbf{G}$ . Dados os  $k$  qubits do primeiro registrador já preparados pela porta Hadamard, formando uma combinação probabilística de todas as palavras de informação, a ação de  $G'$  é gerar os  $n-k$  qubits redundantes no segundo registrador.

A entrada para o circuito é dada por:

$$|\Psi_0\rangle = |0000\rangle \otimes |000\rangle \otimes |000\rangle \otimes |0000\rangle \quad (3.10)$$

O estado do sistema, após a preparação do primeiro registrador pela porta Hadamard, é definido por:

$$|\Psi_1\rangle = \frac{1}{4} \sum_{i=0}^{15} |i\rangle \otimes |000\rangle \otimes |000\rangle \otimes |0000\rangle \quad (3.11)$$

significando que os quatro qubits do primeiro registrador estão em sobreposição equiprovável enquanto os demais permanecem inalterados. A porta  $G'$ , ao ser operada sobre o estado  $|\Psi_1\rangle$  cria, nos primeiro e segundo registradores, a sobreposição de todas as palavras código.

Já foi visto que a geração de uma palavra código é dada pela multiplicação de um vetor de informação pela matriz geradora (vide 3.2). Dessa forma, cada coluna da matriz  $G$  é responsável por gerar um bit da palavra código. Os qubits do primeiro registrador passam inalterados, pois a matriz  $G$  está em sua forma sistemática (matriz identidade nas primeiras colunas). Já os qubits do segundo registrador devem ser gerados de acordo com as colunas seguintes da matriz  $G$ , que neste caso são as colunas 5, 6 e 7.

A implementação da porta  $G'$  é baseada em simples portas NOT controladas (CNOT). O circuito da figura 3.5 realiza a codificação para o exemplo em questão.

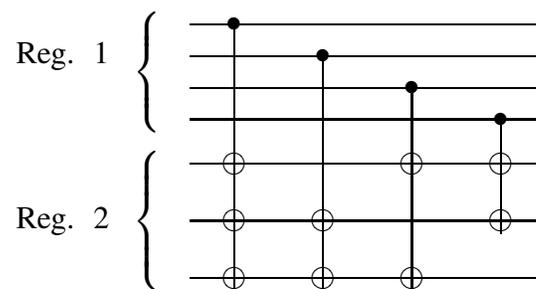


Figura 3.5: Circuito para codificação da porta  $G'$ .

Cada coluna da matriz  $G$  será representada por um qubit do circuito. As portas CNOT são controladas pelos qubits do primeiro registrador cujos valores não são alterados. Tomando como exemplo o qubit número cinco, vê-se que o mesmo é alterado de acordo com os valores da quinta coluna (1 0 1 1). Da mesma forma, o qubit seis é alterado pelos valores (1 1 0 1). Seguindo o mesmo processo para cada um dos qubits do segundo registrador, a porta  $G'$  estará completa.

### A porta C

A função da porta  $C$  é calcular o peso de Hamming de cada uma das palavras do código. Como todas as palavras código já estão armazenadas em sobreposição nos dois primeiros registradores, esta operação criará a sobreposição dos pesos no terceiro registrador. O estado

do sistema após a operação desta porta pode ser visto na equação 3.12.

$$\begin{aligned}
|\Psi_3\rangle = & \\
& \frac{1}{4} ( |0000\rangle \otimes |000\rangle \otimes |000\rangle \\
& + |0001\rangle \otimes |110\rangle \otimes |011\rangle \\
& + |0010\rangle \otimes |101\rangle \otimes |011\rangle \\
& + |0011\rangle \otimes |011\rangle \otimes |100\rangle \\
& + |0100\rangle \otimes |011\rangle \otimes |011\rangle \\
& + |0101\rangle \otimes |101\rangle \otimes |100\rangle \\
& + |0110\rangle \otimes |110\rangle \otimes |100\rangle \\
& + |0111\rangle \otimes |000\rangle \otimes |011\rangle \\
& + |1000\rangle \otimes |111\rangle \otimes |100\rangle \\
& + |1001\rangle \otimes |001\rangle \otimes |011\rangle \\
& + |1010\rangle \otimes |010\rangle \otimes |011\rangle \\
& + |1011\rangle \otimes |100\rangle \otimes |100\rangle \\
& + |1100\rangle \otimes |100\rangle \otimes |011\rangle \\
& + |1101\rangle \otimes |010\rangle \otimes |100\rangle \\
& + |1110\rangle \otimes |001\rangle \otimes |100\rangle \\
& + |1111\rangle \otimes |111\rangle \otimes |111\rangle \otimes |0000\rangle )
\end{aligned} \tag{3.12}$$

Os circuitos para executar esta tarefa são de construção simples na computação clássica. Uma realização possível utiliza uma estrutura simples, constituída apenas por portas CNOT, e que funciona como um contador. O circuito que implementa a porta C para o exemplo de código aqui seguido, pode ser visto na figura 3.6.

### A porta *Cont*

A leitura do terceiro registrador no estado  $|\Psi_3\rangle$  é similar a um experimento aleatório que, de acordo com a distribuição probabilística, resulta em um dos possíveis valores de peso. A execução de tal leitura uma determinada quantidade de vezes leva a uma estimativa do polinômio enumerador de pesos. Tal polinômio tem a estrutura

$$A(x) = A_0 + A_1x + A_2x^2 + \dots + A_nx^n, \tag{3.13}$$

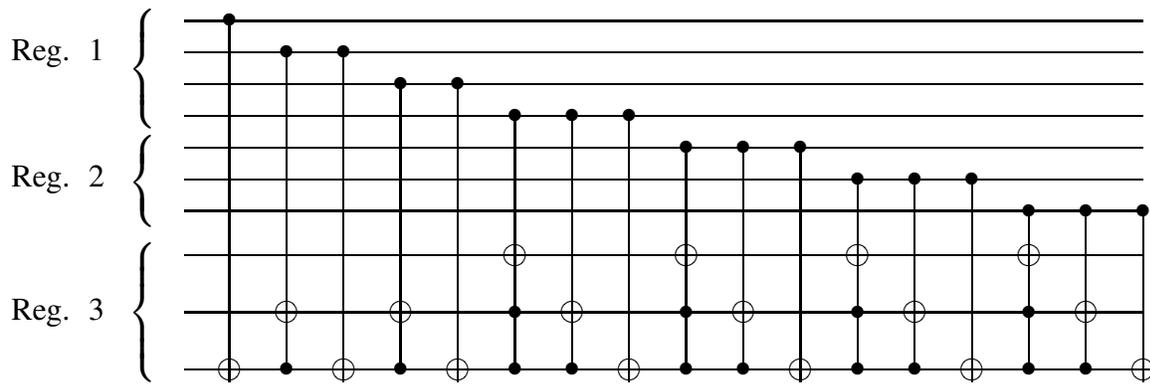


Figura 3.6: Circuito para cálculo da porta C.

onde o índice  $A_i$  é a quantidade de palavras de peso  $i$ . Logo,  $A_0 = 1$  pois um código linear sempre contém a palavra nula, e  $A_1, \dots, A_{d-1}$  são iguais a zero. O cálculo do peso mínimo consiste na busca do primeiro  $A_i$  cujo valor é não nulo, ou seja,  $A_d$ . A forma utilizada neste trabalho para a busca de  $d$  baseia-se na contagem de cada  $A_i$  utilizando o algoritmo quântico de contagem já exposto na seção 2.4. Portanto, a porta  $Cont$  deve ser vista como a implementação do algoritmo quântico de contagem (vide figura 2.8) apenas estando os registradores 3 e 4 (no circuito da figura 3.4) em ordem invertida.

Para o funcionamento interno, a contagem de cada  $A_i$  requer a existência de oráculos que identifiquem os valor  $i$  que estão escritos no terceiro registrador. Na figura 3.7 estão descritos todos os oráculos para o exemplo utilizado neste capítulo, onde o mais a esquerda reconhece  $i = 1$  e sucessivamente até  $i = 7$ .

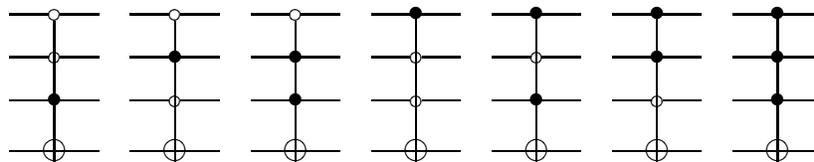


Figura 3.7: Conjunto de testes para a etapa de contagem.

### 3.3.2 Algoritmo para Encontrar $d$

Composto de acordo com a descrição anterior, o circuito é capaz de dizer se existem palavras códigos com um determinado peso. Isso acontece pois, como o algoritmo de contagem deixará registrada a quantidade de palavras com peso  $i$  no quarto registrador, a leitura do mesmo

com valor zero definirá a inexistência deste peso dentre as palavras do código. Realizando iterações para todos os  $i < d$  resultará na leitura do valor nulo, até que a contagem de  $A_d$  seja feita e resulte em valor distinto de zero.

O algoritmo para encontrar o peso  $d$  pode ser resumido da seguinte forma:

1. faça  $i \leftarrow 1$ ;
2. faça  $d \leftarrow 0$ ;
3. enquanto  $d = 0$  faça:
  - (a) execute contagem para  $A_i$ ;
  - (b) faça  $d \leftarrow$  leitura do registrador 4;
  - (c)  $i \leftarrow i + 1$ ;
4. retorne  $d$ .

### 3.3.3 Avaliação de Desempenho e Recursos

Dado que o desempenho do algoritmo clássico para a busca de  $d$  baseia-se no teste das  $N$  palavras do código, considera-se que o mesmo é da ordem de  $O(N)$ . Será mostrado nesta seção que o algoritmo desenvolvido neste trabalho é da ordem de  $O(\log N \sqrt{\log N})$ .

Os dois primeiros registradores possuem  $n$  qubits para armazenar os  $N$  vetores. O terceiro registrador possui  $\log n$  qubits, pois deve guardar os valores binários dos pesos de palavras do código que variam de 0 a  $n$ . Tal registrador servirá de entrada para o circuito da contagem, correspondendo ao segundo registrador do circuito da figura 2.8, e seu tamanho será denotado por  $n' = \log n$ .

Na conclusão da seção 2.4 foi visto que o algoritmo da contagem pode executar em  $\Theta(\sqrt{N'})$ , onde  $N'$  é a quantidade de qubits de entrada do circuito. A equação 3.14 mostra a dedução da relação existente entre a quantidade inicial de elementos  $N$  e a quantidade de elementos de entrada na porta  $Cont(N')$ .

$$\begin{aligned}
N' &= 2^{n'} = 2^{\log n} = 2^{\log \log N} & (3.14) \\
\log N' &= \log 2^{\log \log N} \\
\log N' &= (\log \log N) \log 2 \\
\log N' &= \log \log N \\
N' &= \log N
\end{aligned}$$

Com esta redução da quantidade de elementos na entrada do circuito da contagem, o circuito 3.4 executa na ordem de  $\Theta(\sqrt{\log N})$  chamadas ao oráculo. Como o algoritmo completo executa no mínimo  $d$  vezes este circuito e  $d < n = \log N$ , então sua complexidade é da ordem de  $O(\log N \sqrt{\log N})$ . Dado que estes resultados são probabilísticos e cada execução do circuito tem uma probabilidade de erro de  $\epsilon = 1/6$ , a iteração para cada teste de  $A_i$  pode ser repetida levando à redução do erro a  $\epsilon = 0.004$  com apenas 3 repetições.

Em relação aos recursos necessários para a execução algoritmo proposto, pode-se calcular facilmente a quantidade de qubits utilizados. Este valor equivale à soma de  $\log N$  qubits para os dois primeiros registradores,  $\log \log N$  do terceiro, e ainda  $t = \lceil n/2 \rceil + 3 \equiv O(\log N)$  do último registrador. Claramente este valor é da ordem de  $O(\log N)$  qubits.

Com relação a quantidade de portas necessárias, tem-se:

- para a fase de preparação são necessárias  $O(n)$  portas Hadamard;
- para a geração das palavras código são utilizadas  $O(n)$  portas *CNOT*;
- na fase de cálculo dos pesos das palavras código são necessárias  $\Theta(n \log n)$ ;
- e finalmente na fase de contagem quântica são utilizadas  $\sqrt{\log n}$  execuções da iteração de Grover (G), onde G é da ordem de  $O(\log n)$ , seguida da QFT inversa que requer  $\Theta(t^2) = \Theta((\log n)^2)$  portas.

Assim sendo, a quantidade de portas do circuito é da ordem de  $O((\log n)^2)$ .

# Capítulo 4

## Ataque Quântico ao Gerador

### Pseudo-Aleatório de Blum-Micali

Um gerador pseudo-aleatório é um algoritmo determinístico que utiliza aritmética para gerar sequências numéricas simulando a aleatoriedade. Estes algoritmos são de fundamental importância devido à dificuldade em encontrar fontes realmente aleatórias. Por serem menos custosos que hardwares geradores de números realmente aleatórios, estes algoritmos tornam-se a principal alternativa de uso, como por exemplo, em simulações de sistemas físicos com características não-determinísticas. Os principais usos dos geradores pseudo-aleatórios na Teoria da Informação e Computação estão ligados à segurança da informação, já que os principais algoritmos de criptografia utilizam-se destes geradores para realizar suas tarefas.

Na primeira parte deste capítulo será feita uma introdução ao algoritmo de Blum-Micali. Na segunda parte será definido um ataque ao mesmo que utiliza ferramental quântico para melhorar o ataque bruta-força clássico.

#### 4.1 Gerador de Blum-Micali

O gerador pseudo-aleatório de Blum-Micali [BM84] possui uma estrutura iterativa bastante simples. Para gerar um número  $b$  com  $j$  bits o algoritmo inicia com uma semente, gera um número  $x_1$  e deste é retirado o primeiro bit ( $b_1$ ) de  $b$ . Iterativamente, o processo segue gerando novos números  $x_i$  baseados nos valores anteriores ( $x_{i-1}$ ) e escolhendo novos bits.

A semente definida por Blum e Micali é composta por 3 valores:  $(p, g, x_0)$ . O número  $p$

é um primo qualquer, normalmente muito grande, que define o conjunto  $G = \mathbb{Z}_p^*$  (conjunto dos inteiros  $\text{mod } p$ ) no qual serão efetuados os cálculos abaixo apresentados. O valor  $g$  é um gerador para  $G$ , ou seja,  $g^y \text{ mod } p$  gera todos os elementos de  $G$ , onde  $y \in \mathbb{Z}_{p-1}$ . Ambos os números  $p$  e  $g$  são públicos e podem ser usados inúmeras vezes na tentativa de ataque ao gerador. A chave secreta do gerador é o valor  $x_0 \in \{x | y^2 \equiv x \text{ mod } p\}$  (o conjunto de todos os quadrados  $\text{mod } p$ ).

Para o bit  $i$  ( $1 \leq i \leq x$ ), a palavra  $b$  é gerada de acordo com a equação 4.1, onde a equação 4.2 significa que  $b_i = 1$  se e somente se  $x_i > (p-1)/2$  e  $b_i = 0$  caso contrário.

$$x_i := g^{x_{i-1}} \text{ mod } p \quad (4.1)$$

$$b_i := \delta_{x_i > (p-1)/2} \quad (4.2)$$

Um exemplo pode ser dado ao gerar um número de quatro bits a partir da seguinte semente:  $p = 19$ ,  $g = 2$  e  $x_0 = 4$ . A equação 4.3 mostra os passos iterativos da geração de  $b = 1010$ . Cada passo do algoritmo é definido como um estado interno do sistema cujo estado inicial é baseado na semente.

$$\begin{aligned} x_1 = 2^4 = 16 &\Rightarrow b_1 = 1 \\ x_2 = 2^{16} = 5 &\Rightarrow b_2 = 0 \\ x_3 = 2^5 = 13 &\Rightarrow b_3 = 1 \\ x_4 = 2^{13} = 3 &\Rightarrow b_4 = 0 \end{aligned} \quad (4.3)$$

A questão aqui examinada é se há como descobrir a chave secreta ( $x_0$ ) utilizando-se os dados conhecidos da semente ( $p$  e  $g$ ) e algumas palavras geradas pelo algoritmo ( $b$ 's).

### 4.1.1 Ataque a Geradores Pseudo-aleatórios

Os geradores pseudo-aleatórios devem passar por alguns testes avaliadores para que sejam considerados fortes o suficiente para determinados usos. Dentre estes testes os mais importantes estão relacionados com encontrar padrões nos valores gerados, como exemplo, buscar elementos (bits) consecutivos que ocorrem de forma frequente. Como estes algoritmos se baseiam em estruturas aritméticas, vários problemas podem expor algum padrão do gerador. Alguns deles são: sementes fracas, ou seja, estados iniciais que levam o gerador a produzir

saídas com repetições periódicas acima do esperado; e não uniformidade na distribuição dos valores gerados.

Desta forma, algumas regras são definidas para que um gerador deste tipo possa ser utilizado. Uma dessas exigências define que, dada uma sequência gerada, um atacante não deve poder prever bits anteriores ou posteriores, ou ainda descobrir um estado interno do gerador. Uma outra regra define que um ataque deve ser computacionalmente intratável para a busca de um estado interno anterior dado um estado posterior.

No caso do algoritmo de Blum-Micali, a segurança está baseada na dificuldade de descobrir, dada qualquer sequência gerada, a componente  $x_0$  da semente. Fica claro que um algoritmo força-bruta é possível, pois o algoritmo gerador pode ser executado para cada  $x_0 \in \mathbb{Z}_{p-1}$  e sua saída comparada com uma ou mais palavras geradas. Mas esta abordagem é computacionalmente intratável dada a explosão na quantidade de valores a serem testados quando  $p$  torna-se suficientemente grande.

## 4.2 Busca Quântica de $x_j$

Nesta seção serão analisadas algumas características de um ataque quântico ao gerador de Blum-Micali. O ataque pretende descobrir o elemento  $x_j \in G$  que é o elemento gerador do  $j$ -ésimo bit do valor  $b$  gerado. A ideia base do algoritmo é a reconstrução do número gerado  $b$ , começando com o conjunto  $G$  e reduzindo-o através de testes em busca das possíveis sequências geradoras. Obviamente que classicamente esta solução força-bruta não leva a bons resultados devido à explosão exponencial de possibilidades. No caso da utilização de um computador quântico pode-se lançar mão de paralelismo real para eliminar tal dificuldade. Mostra-se nesta seção uma aplicação do algoritmo quântico de Grover para a busca de  $x_j$  com a utilização de  $(O(\log p \sqrt{\log p}))$  passos.

O atacante do gerador tem conhecimento dos seguintes dados:

- o número primo  $p$  gerador do espaço de busca;
- a raiz  $g$  do grupo finito;
- e uma palavra pseudo-aleatória  $b$  gerada pelo algoritmo.

Na estrutura do circuito da figura 4.1, destacam-se os quatro registradores necessários para o funcionamento do algoritmo e suas quatro fases. O primeiro registrador possui  $j$  bits sequenciais de um número gerado, enquanto o segundo possui  $\lceil \log p \rceil$  qubits necessários para gerar os elementos do conjunto  $G$ . O terceiro e o quarto registradores possuem qubits auxiliares, onde o terceiro recebe os mesmos  $j$  qubits do primeiro registrador e o quarto possui apenas um qubit utilizado pelo oráculo no algoritmo de Grover.

As quatro fases do circuito podem ser resumidas em:

- preparação da superposição no segundo registrador com portas Hadamard;
- iterações de filtragem (porta  $F(b_i)$ ) e permutação (porta  $P$ ) dos elementos do segundo registrador, marcando o resultado no terceiro registrador;
- aplicação do algoritmo de Grover (porta  $G^*$ ), para amplificar o elemento marcado;
- leitura dos qubits do segundo registrador.

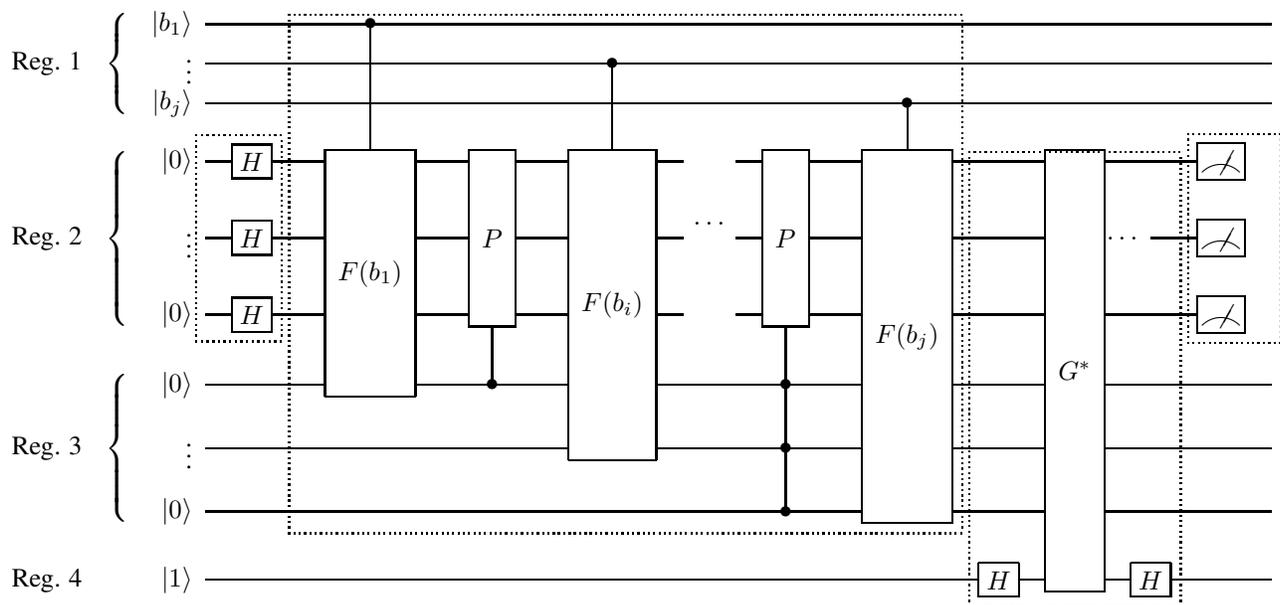


Figura 4.1: Circuito de ataque ao gerador pseudo-aleatório de Blum-Micali.

### 4.2.1 Simulação do Ataque Clássico utilizando o Paralelismo Quântico

A segunda fase do algoritmo simula o ataque clássico força-bruta no que diz respeito a aplicar o algoritmo de Blum-Micali em todos os elementos do grupo. A diferença clara é que a

utilização da superposição quântica torna estes cálculos muito mais eficientes. Assim como o gerador pseudo-aleatório, o circuito mostrado é iterativo, buscando filtrar, dentro do conjunto de possibilidades, aqueles elementos capazes de gerar o bit ( $b_i$ ) em questão. Um exemplo do seu funcionamento pode ser visto na tabela 4.1 com  $p = 19$ ,  $g = 2$  e  $b = 10001000$ .

$G_1 = \{1, 2, 3, \dots, 18\}$	$F(b_1) \Rightarrow \{16, 13, 14, 18, 17, 15, 11, 12, 10\}$	$G_2 = P(G_1)$
$G_2 = \{5, 3, 6, 1, 10, 12, 15, 11, 17\}$	$F(b_2) \Rightarrow \{5, 3, 6, 1\}$	$G_3 = P(G_2)$
$G_3 = \{13, 8, 7, 2\}$	$F(b_3) \Rightarrow \{8, 7, 2\}$	$G_4 = P(G_3)$
$G_4 = \{9, 14, 4\}$	$F(b_4) \Rightarrow \{9, 4\}$	$G_5 = P(G_4)$
$G_5 = \{18, 16\}$	$F(b_5) \Rightarrow \{18, 16\}$	$G_6 = P(G_5)$
$G_6 = \{1, 5\}$	$F(b_6) \Rightarrow \{1, 5\}$	$G_7 = P(G_6)$
$G_7 = \{2, 13\}$	$F(b_7) \Rightarrow \{2\}$	

Tabela 4.1: Funcionamento do circuito para  $p = 19$ ,  $g = 2$  e  $b = 10001000$

Cada linha da tabela 4.1 mostra uma iteração de filtragem do circuito. A operação  $P(G_n)$  permuta os elementos do subconjunto  $G_n$  do conjunto inicial  $G$ , utilizando a equação 4.1. O operador  $F(b_i)$  implementa a equação 4.2, realizando assim um teste relacionado ao bit  $b_i$ , escolhendo dentro de  $G_n$  os elementos que geram este bit. No caso do algoritmo aqui apresentado, em cada iteração, um qubit auxiliar ( $q_i$ ) do terceiro registrador é associado aqueles elementos que satisfazem ao teste ( $F(b_i)$ ).

### A Porta $F$

A operação  $F$  é responsável por filtrar os elementos  $x_i$  capazes de gerar o bit  $b_i$  em cada aplicação. A filtragem é feita utilizando um bit  $q_i$  do terceiro registrador, que deve ser invertido do seu estado inicial  $|0\rangle$  para  $|1\rangle$  caso o teste tenha sucesso. Como já foi definido anteriormente, o teste divide os elementos em dois sub-conjuntos: um com os elementos maiores que  $(p - 1)/2$ , e outro com os elementos restantes. Sabe-se que um circuito para este tipo de comparação existe e é eficiente na computação clássica, e segundo [Ben73] todo algoritmo polinomial no modelo clássico pode ser implementado por um algoritmo polinomial no modelo quântico.

O circuito da figura 4.2 mostra uma porta similar ao operador  $F$  no caso de se considerar a aproximação entre a função  $(p - 1)/2$  e o teste do bit mais significativo.

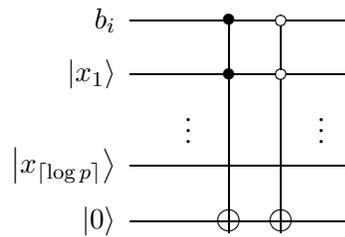


Figura 4.2: Porta similar à porta  $F$  utilizando o bit mais significativo de  $x_i$ .

### A Porta $P$

A porta  $P$  implementa o cálculo  $x_i = g^{x_{i-1}} \bmod p$ . Como os elementos  $x_i$  pertencem a um grupo finito e  $g$  é um gerador do grupo, seu cálculo é uma função injetora que cria uma permutação dos elementos. A tabela 4.2 apresenta a permutação dos elementos do exemplo utilizando  $p = 19$  e  $g = 2$ .

$i$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$2^i$	2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1	2

Tabela 4.2: Permutação dos elementos utilizando  $p = 19$  e  $g = 2$

A definição da função como uma permutação auxilia na implementação da porta. O operador  $P$  deve ser definido como o somatório de vários projetores no formato  $|y\rangle\langle x|$  onde  $y = 2^i$  e  $x = i$ . No caso do exemplo, o operador deve ser:

$$P = |2\rangle\langle 1| + |4\rangle\langle 2| + \dots + |1\rangle\langle 18|. \quad (4.4)$$

Porém, para que o operador  $P$  seja unitário, é necessário complementar o mesmo com o elemento  $|0\rangle\langle 0|$  e  $|x'\rangle\langle x'|$  onde  $p \leq x' < 2^{\lceil \log p \rceil}$ , que são gerados inevitavelmente na primeira fase do circuito, mas que não pertencem ao conjunto  $\mathbb{Z}_p$ . Tais elementos serão “eliminados” do conjunto de elementos válidos pelos testes sucessivos desta fase do circuito.

### 4.2.2 Utilização do Algoritmo de Grover

Nesta terceira fase do algoritmo são executadas algumas iterações de Grover para que o elemento marcado no segundo registrador possa ter sua probabilidade de leitura amplificada. O oráculo para esta tarefa é muito simples (vide circuito da figura 4.3) e marca o qubit do

quarto registrador de acordo com os qubits do terceiro. Como já explicado na subseção anterior, o terceiro registrador terá todos os qubits com valor  $|1\rangle$  apenas para o elemento que satisfizerem os testes da fase anterior do circuito. Desta forma, toda amplificação ocorrida no elemento  $|11\dots 1\rangle$  do terceiro registrador será espelhada no valor  $x_j$  do segundo registrador.

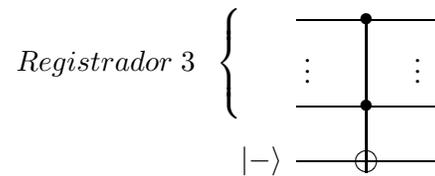


Figura 4.3: Estrutura do oráculo utilizado para as iterações de Grover da porta  $G^*$ .

### 4.2.3 Conclusões

Esta seção tece sobre a segurança do gerador pseudo-aleatório perante um ataque utilizando o circuito proposto neste capítulo. Uma segunda questão a ser analisada diz respeito a quantidade de bits  $j$  necessária para a filtragem de um único elemento  $x_j \in G$  como resultado do circuito.

O circuito proposto computa o valor de  $x_j$  em  $\Theta(\sqrt{\log p})$  apenas quando recebe uma quantidade  $j$  de bits necessária para marcar um único elemento no segundo registrador. Uma análise aprofundada deste valor é delegada para trabalhos futuros (vide capítulo 5). No entanto, é possível criar uma estimativa de  $j$  baseada no fato de que geradores pseudo-aleatórios visam utilizar sementes que espalhem o melhor possível os valores ao realizar a permutação dada pela equação 4.1. Desta forma, a aplicação do filtro implementado pela porta *P particiona* o conjunto  $G$ , e subsequentemente os subconjuntos de  $G$ , pela metade. Dado que o circuito escolhe apenas um dos dois sub-conjuntos a cada passo de filtragem, o valor  $j$  é da ordem de  $O(\log p)$ . Se tal hipótese for verdadeira, a etapa de filtragem é realizada em  $O(\log p)$  passos e o algoritmo de Grover necessitaria de  $O(\sqrt{\log p})$  aplicações do oráculo.

Na seção 4.1.1 foram enumeradas algumas regras que definem um gerador pseudo-aleatório seguro. O circuito apresentado neste capítulo auxilia a quebrar a primeira delas, já que é demonstrada uma forma de computar o valor  $x_j$ , um estado interno do gerador, a partir de uma palavra gerada. É fácil ver que, de posse do valor de  $x_j$ , é possível determinar os bits posteriores ao mesmo, assim a segunda regra é quebrada. Por fim, utilizando-se a já

mencionada Transformada Quântica de Fourier (seção 2.4), é possível o cálculo eficiente do logaritmo discreto [Sho94], e assim computar sequencialmente os demais estados internos  $x_i$ , onde  $i \geq 0$ . Com a descoberta do valor  $x_0$  a semente é desvendada e o atacante passa a poder simular o gerador.

# Capítulo 5

## Conclusões e Trabalhos Futuros

Neste capítulo são resumidas algumas realizações deste trabalho e também são levantadas algumas linhas de pesquisa a serem desenvolvidas futuramente.

### 5.1 Conclusões

Este trabalho apresentou dois resultados pertencentes à área da Teoria da Informação atestando a eficiência do uso de algoritmos quânticos quando comparados com os algoritmos clássicos desenvolvidos para o mesmo fim.

O primeiro resultado foi o desenvolvimento de um algoritmo para o cálculo da distância mínima de um código de bloco linear, publicado em anais de eventos nacionais da área [OA06; OA07]. A complexidade do algoritmo proposto é da ordem de  $O(\log N \sqrt{\log N})$  sendo mais eficiente que o algoritmo clássico, cuja ordem de complexidade é  $O(N)$  (para  $N$  igual a quantidade de palavras do código a serem analisadas).

O segundo algoritmo criado é um ataque quântico ao gerador pseudo-aleatório de Blum-Micali também aceito para publicação [ONA07]. O algoritmo mostrou-se capaz de quebrar a segurança do gerador com  $O(\sqrt{N})$  aplicações do oráculo onde  $N$  pode chegar a ser igual a  $\log p$  ( $p$  número primo que define um espaço de testes exponencialmente grande) dadas hipóteses que buscarão ser provadas em trabalhos futuros.

Com tais resultados, o desejo inicial de contribuir para expandir a utilização da Computação Quântica foi alcançado e espera-se que novas pesquisas e pesquisadores na área possam encontrar nesse texto recursos úteis.

## 5.2 Trabalhos Futuros

No processo de desenvolvimento das soluções apresentadas neste trabalho, algumas linhas de pesquisa se mostraram promissoras. Na área de Códigos para Controle de Erros, referente ao capítulo 3, são sugeridos os seguintes estudos posteriores:

- Aplicação do arcabouço desenvolvido neste trabalho não somente para teste, mas também para a geração de códigos lineares, principalmente códigos expressos em grafos [LMSS01];
- Estudo similar para códigos não lineares.

Já na área dos Geradores Pseudo-aleatórios, referente às análises realizadas no capítulo 4, três perspectivas em especial podem ser levadas em consideração:

- Estudo mais aprofundado da quantidade de bits  $j$  da palavra  $b$  necessários para o ótimo funcionamento do algoritmo [CH06];
- Aplicação da estrutura apresentada para quebra de outros geradores similares ao Blum-Micali;
- Avaliação de um algoritmo descrito no trabalho [Wat06] com a intenção de melhorar os ganhos do algoritmo proposto.

# Bibliografia

- [BB92] Andre Berthiaume and Gilles Brassard. Oracle quantum computing. In *Proceedings of the Workshop on Physics of Computation: PhysComp '92*, pages 195–199, Los Alamitos, CA, 1992. Institute of Electrical and Electronic Engineers Computer Society Press.
- [Ben73] Charles Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17:525–532, 1973.
- [BHMT00] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation, 2000.
- [BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum counting. *Lecture Notes in Computer Science*, 1443:820+, 1998.
- [BM84] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984.
- [BV97] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997.
- [CH06] Daniel R. Cloutier and Joshua Holden. Mapping the discrete logarithm. Apr 2006.
- [Deu85] David Deutsch. Quantum theory, the church-turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 400(1818):97–117, 1985.
- [Dir58] Paul Adrien Maurice Dirac. *The Principles of Quantum Mechanics*. Oxford University Press, Oxford, 1958.

- [DJ92] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. In *Proceedings Royal Society London*, volume 439A, pages 553–558, 1992.
- [Fey82] Richard Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6&7):467–488, 1982.
- [Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pages 212–219, 1996.
- [Hei25] Werner Heisenberg. Über quantentheoretische umdeutung kinematischer und mechanischer beziehungen. 33:879–893, 1925. English title: Quantum-Theoretical Re-interpretation of Kinematic and Mechanical Relations.
- [Her85] Nick Herbert. *A Realidade Quântica*. Livraria Francisco Alves Editora S.A., 1985. Traduzido em 1989.
- [Joz91] Richard Jozsa. Characterizing classes of functions computable by quantum parallelism. In *Proceedings Royal Society London*, volume A435, pages 563–574, 1991.
- [LLM05] Carlile Lavor, Leo Liberti, and Nelson Maculan. Grover’s algorithm applied to the molecular distance geometry problem. In *VII Brazilian Congress of Neural Networks*, 2005.
- [LMSS01] Michael G. Luby, Michael Mitzenmacher, Amin M. Shokrollahi, and Daniel A. Spielman. Efficient erasure correcting codes. *IEEE Transactions on Information Theory*, 47(2):569–584, February 2001.
- [NC00] Michael A. Nielsen and Issac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, UK, ISBN 0-521-63235-8, 2000.
- [OA06] Nigini A. Oliveira and Francisco de Assis. Aplicação do algoritmo de grover para estimação da distância mínima de códigos de bloco lineares. In *Workshop-Escola de Computação e Informação Quântica*, pages 229–237, 2006.

- [OA07] Nigini A. Oliveira and Francisco de Assis. Algoritmo quântico para cálculo da distância mínima de códigos de bloco lineares. In *Workshop-Escola de Computação e Informação Quântica*, pages 13–20, 2007.
- [ONA07] Nigini Abilio Oliveira, Edmar José Nascimento, and Francisco Marcos de Assis. Ataques quânticos ao gerador pseudo-aleatório de blum-micali. In *XXV Simpósio Brasileiro de Telecomunicações (SBrT'07)*, 2007.
- [Sha48] Claude E. Shannon. A mathematical theory of communications. *Bell Systems Technical Journal*, 27:379–423 and 623–656, July and October 1948.
- [Sho94] Peter Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.
- [Sim94] David R. Simon. On the power of quantum computation. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 116–123, Los Alamitos, CA, 1994. Institute of Electrical and Electronic Engineers Computer Society Press.
- [Wat06] John Watrous. Zero-knowledge against quantum attacks. In *38th Annual ACM Symposium on Theory of Computing*, pages 296–305, May 2006.
- [Wil98] Colin P. Williams. *Explorations in Quantum Computing*. Springer-Verlag, New York, NY, 1998.