Universdade Federal de Pernambuco
Centro de Informática

Análise de Múltiplas Soluções do Algoritmo de Grover

Bacharelado em Engenharia da Computação

Débora Fortunato Dias

Recife, Dezembro de 2019

Universdade Federal de Pernambuco
Centro de Informática

Análise de Múltiplas Soluções do Algoritmo de Grover

Débora Fortunato Dias

Trabalho apresentado ao curso de Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Engenharia da Computação.

Orientador: Fernando Maciano de Paula Neto

Recife, Dezembro de 2019

Para minha querida e amada vovó Otília (In Memoriam).

Agradecimentos

Agradeço a todos que contribuíram direta e indiretamente para que eu pudesse concluir com sucesso essa primordial etapa da minha vida acadêmica.

Faço um agradecimento especial ao professor e orientador Fernando Maciano de Paula Neto por me conceder a oportunidade de realizar esta pesquisa sob sua orientação.

Aos meus pais e familiares que me apoiaram por todos esses anos. A todos os professores e funcionários do Centro de Informática. E finalmente aos meus colegas de turma e curso pelos momentos de conversa, descontração e incentivo.

Resumo

Algoritmos de busca são um dos pilares da computação. Com diferentes técnicas e aproximações, é possível realizar tarefas que transitam desde a simplicidade de uma busca em um único array, até a sofisticação de complexos códigos de quebra criptográfica. O tempo para realizar uma busca é comumente proporcional ao número de elementos que serão vasculhados. Considerando uma análise exaustiva, o algoritmo teria que verificar todos os elementos presentes para então achar o elemento desejado.

Em computadores quânticos um bit quântico pode assumir o valor de 0, 1, ou os dois ao mesmo tempo. Essa propriedade é chamada de superposição. Nesse estado especial, um algoritmo quântico poderia analisar em uma única execução todas as entradas possíveis para uma dada função. Devido a isso, um tal algoritmo pode vasculhar um conjunto de N elementos em um tempo proporcional a $\sqrt{N}$, o tornando um grande diferencial quando comparado a algoritmos clássicos de busca.

É possível também utilizar algoritmos quânticos de busca em problemas de otimização para resolver problemas complexos, cujo espaço de busca é exponencial no tamanho da entrada. Tendo em vista isso, o objetivo desta pesquisa é realizar uma análise do importante algoritmo quântico de busca proposto por Lov Grover, na década de 90. Será analisado o desempenho do algoritmo para diversos cenários propostos, com parâmetros variáveis. Seu desempenho em simulações é próximo aos resultados teóricos, no entanto, em máquinas reais observa-se a alta presença de ruído e taxas de erros à medida que aumentamos a complexidade do algoritmo. É concluído que na tecnologia atual, há uma grande limitação de hardware para problemas mais robustos.

Palavras-chave: Computação, quântica, algoritmos, algoritmos de busca, otimização.

Abstract

Search algorithms are one of the pillars of computing. With different techniques and approaches, you can perform tasks that range from the simplicity of a single array search to the sophistication of complex cryptographic cracking codes. The time to perform a search is commonly proportional to the number of elements that will be searched. Considering an exhaustive analysis, the algorithm would have to check all present elements to find the desired element.

In quantum computers a quantum bit can assume the value of 0, 1, or both at the same time. This property is called an overlay. In this special state, a quantum algorithm could execute in a single execution all possible inputs to a given function. Because of this, such an algorithm can search a set of N elements at a time proportional to $\sqrt{N}$, making it a great differential when compared to classic search algorithms.

It is also possible to use quantum search algorithms in optimization problems to solve complex problems whose search space is exponential in input size. Given this, the objective of this research is to perform an analysis of the important quantum search algorithm proposed by Lov Grover in the 90's. The performance of the algorithm for each proposed scenario will be analyzed, with variable parameters. Its performance in simulations is close to the theoretical results, however, in real machines there is a high presence of noise and error rates as we increase the complexity of the algorithm. It is concluded that in current technology, there is a major hardware limitation for more robust problems.

Key-words: Quantum computing, algorithm, search algorithm, optimization.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A computação quântica passou a ser estudada com maior interesse graças aos trabalhos de Paul Benioff e Richard Feynman, no início da década de 80. As pesquisas de Feynman o levaram à conclusão de que computadores clássicos não seriam eficientes para modelar sistemas mecânicos quânticos, uma vez que a dimensão do espaço nela tratado cresce exponencialmente em função do número de partículas presentes no sistema [1] . O questionamento do que poderia ser capaz de fazer tal tarefa, o fez sugerir o que seria o nascimento da computação quântica: computadores baseados na lei da mecânica quântica.

Em contrapartida, Benioff demonstrou que sistemas quânticos poderiam modelar uma máquina de Turing. Colocando em termos simples, ele provou que a computação quântica é, no mínimo, tão poderosa quanto a clássica [2]. No entanto, deixou em aberto a resposta se ela poderia ser mais rápida do que a clássica.

Ainda na década de 80, David Deutsch aproveitou algumas questões deixadas por Benioff para demonstrar que um computador quântico universal seria capaz de realizar tarefas que seriam impossíveis para um máquina de Turing universal, como por exemplo a geração de uma função randômica genuína (em computadores clássico isso é apenas pseudo randômico) e cálculos simultâneos em um único registrador (utilizando o paralelismo quântico) [3]. Alguns anos mais tarde já na década de 90, ele e Richard Jozsa foram os responsáveis pela criação do primeiro algoritmo quântico. O algoritmo proposto determina se uma função é constante para todas as entradas ou balanceada [4]. Ainda que não haja uma utilidade prática, este algoritmo foi capaz de voltar a atenção da comunidade científica para a computação quântica por ser mais rápido do que algum algoritmo clássico desenvolvido para o mesmo propósito.

Em 1994 aconteceu mais um grande avanço na computação quântica: a publicação do algoritmo quântico de Peter Shor. Esse algoritmo seria capaz de resolver um problema já bastante conhecido e ainda não resolvido em tempo eficiente nos computadores clássicos - a fatoração de grandes números [5]. A publicação desse artigo gerou uma onda de perspectiva e entusiasmo da comunidade científica e foi um dos grande marcos na área da computação quântica. E não apenas isso, o algoritmo de Shor levantou algumas questões como a da segurança criptográfica, uma vez que um dos métodos mais utilizados no encapsulamento de informações, o RSA, se baseia na dificuldade de fatoração de números grandes [6].

Apenas alguns anos depois, Lov Grover foi responsável por mais um marco história na computação quântica graças a sua publicação do primeiro algoritmo de busca quântico em um espaço desordenado [7]. Esse algoritmo tem um ganho quadrático quando comparado a qualquer equivalente clássico e serve como base para muitos outros algoritmos [8, 9, 10, 11, 12] que visam sua otimização, dado que algoritmos de busca são constante-

mente utilizados, servindo muitas vezes como sub rotina de códigos maiores.

Algoritmos quânticos precisam necessariamente possuir um desempenho melhor do que seu equivalente clássico, caso contrário, não há justificativa para sua implementação. Embora as áreas de informação e complexidade quântica tenham tido importantes avanços, o desenvolvimento de novos algoritmos quânticos não seguem o mesmo ritmo. Peter Shor, o autor do primeiro algoritmo quântico de fatoração, apontou algumas razões para isso [13]. Uma delas se baseia na dificuldade de pensar intuitivamente quando estamos em um meio quântico pelo fato de computadores quânticos funcionarem de uma maneira tão diferente dos computadores clássicos.

Algoritmos de busca são muitas vezes partes fundamentais de diversos códigos. Comumente, para realizar a busca de algum elemento, um algoritmo clássico levaria um tempo proporcional ao número de elementos que serão analisados até a busca ser concluída. Para um algoritmo quântico, a mesma tarefa seria realizada com um ganho quadrático de velocidade. O algoritmo de busca quântico desenvolvido por Lov Grover será o principal algoritmo de estudo desta pesquisa. Será analisado seu funcionamento para múltiplas soluções e performance em computadores quânticos real, algo ainda não explorado para os cenários propostos. Serão observados também a precisão e tempo de execução.

1.1 Objetivos

1.1.1 Gerais

O objetivo geral deste trabalho consiste em desenvolver e analisar simulações do algoritmo quântico de Grover para múltiplas soluções, em ambientes de simulação virtual e em um computador quântico de baixa escala ruidoso.

1.1.2 Específicos

Destacam-se os seguintes objetivos específicos:

- Detalhar e desenvolver as etapas do algoritmo de Grover clássico;
- Detalhar o algoritmo de Grover para múltiplas soluções;
- Realizar simulações em ambiente hipotético e em diferentes computadores quânticos reais, com variações paramétricas;
- Comparar resultados de simulações realizadas;
- Identificar potenciais pesquisas futuras no sentido de melhorar os resultados obtidos à medida que a tecnologia disponível da área avança.

Esta pesquisa está organizada primeiramente apresentando o os conceitos fundamentais e teóricos da computação quântica, dispostos na Seção 1.2 e 1.3. No Capítulo 2 é descrito o algoritmo de estudo desse trabalho, o algoritmo de Grover. São explicadas suas etapas, seu objetivo, algumas limitações e otimizações publicadas.

No Capítulo 3, são apresentadas as ferramentes utilizadas na metodologia e também os cenários de análises e simulações. Os resultados e comparações de todos experimentos realizados encontram-se no Capítulo 4. Finalmente, o Capítulo 5 expõe as conclusões dos resultados observados ao longo desta pesquisa, bem como possibilidades para o desenvolvimento de outras pesquisas relacionadas.

O Apêndice A conta com todos os oráculos utilizados nesse trabalho. O Apêndice

B e C possui os códigos desenvolvidos e usados nas simulações. E por fim, o Apêndice D conta com os cálculos teóricos e esperados para a precisão de diferentes cenários do algoritmo de Grover.

1.2 Mecânica Quântica

A Mecânica Quântica é um conjunto de regras matemáticas que servem para a construção de teorias físicas e consegue descrever com grande precisão eventos subatômico. Ela representa um dos ramos da Física originada pelos físicos Erwin Schrödinger [14] e Werner Heisenberg [15] no fim do século XIX e início do XX [16].

Desde a sua criação até os dias de hoje, a mecânica quântica tem sido aplicada em diversos ramos como na física de partículas, física atômica e molecular, na astrofísica e na matéria condensada [17]. Contudo, até o início da década de 1970 os experimentos feitos para se testar os modelos e teorias construídos a partir da mecânica quântica estavam restritos a sistemas com um número imenso de constituintes, o que tornava inviáveis as demonstrações. E foi a partir desta limitação que a computação quântica começou a criar forma, graças aos trabalhos iniciados por Feynman [1] e Benioff [2].

Alguns postulados da mecânica quântica são fundamentais para realizar computação quântica. O primeiro postulado trata da descrição matemática de um sistema quântico isolado. O segundo trata da evolução dos sistemas físicos quânticos. O terceiro descreve como ser possível extrair informações de um sistema quântico por medições. E o último postulado descreve a forma como sistemas quânticos diferentes podem ser combinados. Eles são explicitados abaixo [18]:

Postulado 1 (Espaço de estados) - Todo sistema físico tem a ele associado um espaço vetorial. Esse espaço é composto e formado por vetores complexos unitários $|\psi\rangle$ e seu complexo conjugado, $\langle\psi|$.

Esse postulado garante que todo o espaço de Hilbert H pode ser descrito por vetores da base ortonormal de H. Com isso, em um espaço finito n com uma base $\{|x_0\rangle, |x_1\rangle ..., |x_{n-1}\rangle\}$ pertencente a H, um estado $|\psi\rangle$ pode ser expresso através de uma combinação linear dada por:

$$|\psi\rangle = \sum_{i=0}^{n-1} a_i |x_i\rangle , \tag{1.1}$$

sendo a_i a amplitude do estado correspondente x_i e $\sum |a_i|^2 = 1$.

Postulado 2 (Evolução de estados) - A evolução temporal de um sistema quântico fechado é descrita por transformações unitárias. A mudança do estado $|\psi_{tx}\rangle$ em um instante x, para o estado $|\psi_{ty}\rangle$ em um instante y, se dá pela aplicação de um operador unitário U:

$$|\psi_{tx}\rangle = U |\psi_{ty}\rangle \tag{1.2}$$

U pode ser representado matricialmente e, por ser unitário, é verdade que:

$$UU^\dagger = U^\dagger U = I, \tag{1.3}$$

onde I é a matriz identidade e $U^\dagger$ é a transposta conjugada de U.

Esse postulado garante que todas as operações realizadas são fisicamente reversíveis - algo obrigatório para computação quântica. Também assegura a conservação do produto escalar, ou seja, após aplicar um operador unitário sobre um vetor, sua norma permanece a mesma.

Postulado 3 (Medição dos estados) - As medições quânticas são descritas por operadores de medições M_n, , onde índice n representa aos possíveis resultados da medição. A probabilidade de que o n seja observado após uma medição de um estado quântico $|\psi\rangle$ é dada por:

$$p(n) = \langle\psi| M_n^\dagger M_n |\psi\rangle , \tag{1.4}$$

e o estado do sistema após a medição

$$|\psi_n\rangle = \frac{M_n |\psi\rangle}{\sqrt{p(n)}} \tag{1.5}$$

Os operadores de medições obedecem a relação de completude, o que significa que

$$\sum_n M_n^\dagger M_n = I, \tag{1.6}$$

o que por consequência garante que a soma das probabilidades se equivalha a 1

$$\sum_n p(n) = \sum \langle\psi| M_n^\dagger M_n |\psi\rangle = 1. \tag{1.7}$$

Postulado 4 (Composição dos estados) - O espaço de estado de um sistema quântico composto é o produto tensorial, $\otimes$, dos espaço de estados componentes. Um sistema quântico de n elementos pode ser expresso por:

$$|\varphi\rangle = |\psi_0\rangle \otimes |\psi_1\rangle \otimes ... \otimes |\psi_{n-1}\rangle \tag{1.8}$$

A computação quântica utiliza tais postulados no seu funcionamento para aproveitar as vantagens que eles podem ofertar, como por exemplo o uso de superposição e emaranhamento.

1.3 Computação Quântica

Na década de 60, Gordon Moore percebeu que a quantidade de transistores presentes em um computador aumentavam significativamente em um intervalo curto de tempo. Essa observação ficou conhecida como Lei de Moore [19], e é válida até os dias atuais. Com isso, a tecnologia avança, os computadores se tornam menores, mais eficientes e rápidos. Porém, ainda há problemas que, ao que tudo indica, possuem solução inatingível ou no mínimo inviável utilizando o paradigma clássico. São os problemas chamados indecidíveis [20]. O estudo e desenvolvimento da computação quântica abrange a possibilidade de um maior poder computacional no futuro. Tal possibilidade propicia não apenas a resolução de problemas em tempo inviável em uma máquina clássica, mas também a melhora de soluções já existentes em ganhos até exponenciais.

Desde que Richard Feynman propôs os conceitos iniciais de um computador quântico em 1982, a computação quântica passou a ser observada com mais regularidade. Foi na

década seguinte que Lov Grover e Peter Shor publicaram importantes algoritmos quânticos que mudariam de vez o cenário da computação, seja ela clássica ou quântica. O algoritmo de busca de Grover e o de fatoração de Shor mostraram-se capazes de resolver problemas em tempo mais rápido que em uma máquina clássica, demonstrando a possibilidade de haver uma forma de computação mais poderosa que a atual.

Um computador quântico executa cálculos utilizando as propriedades da mecânica quântica, e isso já muda drasticamente a forma de enviar, manipular e ler informações em relação à computação clássica [18]. Analogamente a um computador clássico, que funciona a partir de circuitos elétricos e portas lógicas manipulando bits, o computador quântico opera a partir de circuitos quânticos baseados em portas lógicas quânticas, manipulando a sua unidade fundamental, o qbit.

1.3.1 Qbits

A unidade de informação clássica é o bit, cujos valores podem ser "0" ou "1". Nos sistemas clássicos, bits são fisicamente representados pela presença ou não de correntes elétricas em algum componente eletrônico, sendo a presença desta retratada pelo estado lógico 1 e a sua ausência o estado lógico 0. Obviamente os dois valores lógicos de um bit clássico são mutuamente excludentes. Analogamente, a unidade de informação quântica é o bit quântico, ou qbit. Um qbit pode ter os valores lógicos "0", "1" ou qualquer superposição deles [18]. Os vetores da base computacional de um qbit são representados pela seguinte notação matricial:

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

A superposição consiste em uma combinação linear desses estados. A forma mais fácil de se representar maticamente um estado quântico é utilizando os dois vetores das bases ortonormais $|0\rangle$ e $|1\rangle\rangle$:

$$|\psi\rangle = a|0\rangle + b|1\rangle$$

Chamamos a e b de amplitudes de probabilidade de respectivamente $|0\rangle$ e $|1\rangle$. O quadrado da norma desse valor pode ser entendido como a probabilidade do estado quântico colapsar para determinado valor clássico quando for medido. E por ser probabilístico, então $|\mathrm{a}|^2 + |\mathrm{b}|^2 = 1$.

Além da superposição, um outro fenômeno presente na computação quântica, em virtude do uso da mecânica quântica, é o emaranhamento. Emaranhamento é um fenômeno em que partículas quânticas são criadas e manipuladas de modo que nenhuma delas possa ser descrita sem referenciar a outra. Uma medida em um membro de um par emaranhado determinará imediatamente a medida do outro membro. Na computação quântica, essas partículas são os próprios qbits. Em um computador clássico, é possível saber sobre o estado de qualquer bit em memória, sem alterar o sistema. No computador quântico, a situação é diferente, qbits podem estar em estados sobrepostos, ou até mesmo emaranhados, e o simples ato de medir um estado quântico altera seu estado.

Podemos visualizar um bit quântico como um ponto sobre a superfície de uma esfera, a Esfera de Bloch (Figura 1.1). Para isso realizamos a parametrização de $|\psi\rangle$ com

auxílio de dois ângulos, θ e φ, além de $a = \cos\frac{\theta}{2}$ e $b = e^{i\varphi}\sin\frac{\theta}{2}$:

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle \tag{1.9}$$

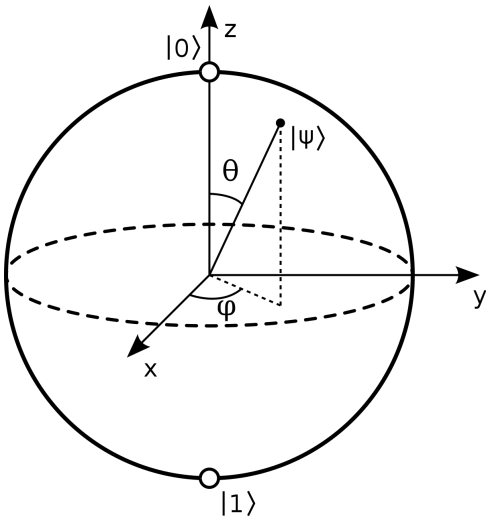


Figura 1.1: Esfera de Bloch

Alguns pontos especiais sobre a esfera de Bloch são mostrados na tabela Tabela 1.1.

θ	φ	$\|\psi\rangle$	Descrição
0	0	$\|0\rangle$	pólo norte da esfera de Bloch
π	0	$\|1\rangle$	pólo sul da esfera de Bloch
$\frac{\pi}{2}$	0	$\frac{(\|0\rangle+\|1\rangle)}{\sqrt{2}}$	equador sobre o eixo x
$\frac{\pi}{2}$	$\frac{\pi}{2}$	$\frac{(\|0\rangle+\|1\rangle)}{\sqrt{2}}$	equador sobre o eixo y

Tabela 1.1: Pontos especiais na Esfera de Bloch

Analisando a Figura 1.1 é válido pensar que é possível armazenar uma grande quantidade de informação - muitos estados - em um único qbit. No entanto, a mecânica quântica garante que ao realizar uma medida sobre um qubit, o estado, em superposição ou não, colapsa para um valor, ou seja, obtêm-se apenas um bit de informação.

1.3.2 Registrador Quântico

Os computadores quânticos utilizam registradores quânticos que são compostos de vários qbits. Quando em superposição, cada qbit de um registrador é uma superposição de $|0\rangle$ e $|1\rangle$. Por consequência, um registrador de n qbits é uma superposição de todas as 2^n possibilidades de estados:

$$|\psi_n\rangle = \sum_{i=0}^{2n-1} a_i |i\rangle$$

Um registrador quântico de 2 qbits seria expresso da seguinte forma:

$$|\psi_2\rangle = a\,|00\rangle + b\,|01\rangle + c\,|10\rangle + d\,|11\rangle$$

Para representar mais de um qbit no espaço de Hilbert, é realizada a operação de produto tensorial, representada por $\otimes$. Portanto, $|\psi\rangle$ nada mais é que a somatória de vários produtos tensoriais entre os qbits que compõem o estado:

$$|\psi\rangle = a\,|0\rangle \otimes |0\rangle + b\,|0\rangle \otimes |1\rangle + c\,|1\rangle \otimes |0\rangle + d\,|1\rangle \otimes |1\rangle$$

1.3.3 Operadores Quânticos

Operadores quânticos são representados matematicamente como matrizes de transformação aplicadas aos registradores quânticos através de um produto tensorial. Essas matrizes precisam necessariamente ser unitárias, ou seja, dado um operador $U : H \to H$, com H sendo o espaço de Hilbert, U é dito unitário quando seu inverso é igual ao seu conjugado transposto:

$$U^{\dagger}U = I$$

As matrizes unitárias garantem que a computação possa ser reversível, ou seja dado um operador X que será aplicado ao qbit $|\psi_1\rangle$ produzindo o resultado $|\psi_2\rangle$, quando aplicarmos a porta quântica inversa de X no qbit $|\psi_2\rangle$, teremos como resultado o qbit inicial $|\psi_1\rangle$.

Operadores unitários não alteram o produto interno entre dois vetores, e geometricamente preservam seus comprimentos e o ângulo formado entre eles. Quando aplicamos uma operação unitária a um único qbit, as mudanças podem ser interpretadas como rotações e reflexões nos eixos x, y e z na Esfera de Bloch. As matrizes de Pauli são importantes exemplos de transformações unitárias sobre um qbit.

Pauli I

É a porta identidade e o resultado de sua operação não altera o estado do qbit de entrada. A Figura 1.2 é a representação da porta Pauli I em um circuito.

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

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle$$
$$I\,|\psi\rangle = a\,|0\rangle + b\,|1\rangle = |\psi\rangle$$

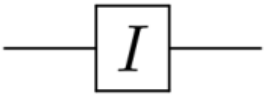


Figura 1.2: Representações da porta Pauli I - Identidade

Pauli X (NOT)

Essa porta realiza uma rotação de 180 graus sobre o eixo x e troca as amplitudes de $|0\rangle$ e $|1\rangle$. A Figura 1.3 são representações da porta Pauli X em um circuito. Ela é definida pela matriz:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle$$
$$X\,|\psi\rangle = a\,|1\rangle + b\,|0\rangle$$

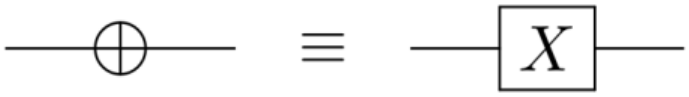


Figura 1.3: Representações da porta Pauli X

Pauli Y

Essa porta realiza uma rotação de 180 graus sobre o eixo y, troca as amplitudes de $|0\rangle$ e $|1\rangle$, nega $|1\rangle$ e depois multiplica todas amplitudes por i. A Figura 1.4 é a representação da porta Pauli Y em um circuito. É definida pela matriz:

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle$$
$$Y\,|\psi\rangle = ib\,|0\rangle - ia\,|1\rangle$$

Figura 1.4: Representações da porta Pauli Y

Pauli Z

Porta que gira os qbits em 180 graus em torno do eixo z, negando a amplitude de $|1\rangle$ apenas. A Figura 1.5 é a representação da porta Pauli Z em um circuito. Sua matriz de é definida por:

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

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle$$

$$Z\,|\psi\rangle = a\,|1\rangle - b\,|1\rangle$$

Figura 1.5: Representações da porta Pauli Z

Hadamard

Quando aplicada a $|0\rangle$ e $|1\rangle$, provoca um superposição idêntica para todos os estados, gerando por consequência uma igual probabilidade de observação para os kets $|0\rangle$ e $|1\rangle$. A utilização dessa porta é vasta e vários algoritmos quânticos começam pela aplicação do Hadamard. Geometricamente, essa porta realiza uma rotação de 90 graus sobre o eixo y, e uma rotação de 180 graus sobre o eixo x. É definida pela matriz:

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

$$H\,|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

$$H\,|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

A probabilidade de ambos estados após a aplicação da porta é exatamente a mesma, diferenciando apenas pela fase do qbit $|1\rangle$. A Figura 1.6 é a representação da porta Hadamard em um circuito.

Figura 1.6: Representações da porta Hadamard

Portas controladas: C-NOT e Toffoli

A Porta C-NOT atua em estados de 2 qbits de entrada, o controle e o alvo. Uma porta controlada age de acordo com o qbit de controle. Ela será ativada apenas quando o qbit de controle estiver no estado $|1\rangle$. Os qbits de controle e alvo podem ser estados superpostos, além disso, podem estar emaranhados. A representação matricial da porta quântica C-NOT é a dada por:

$$C - NOT = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}$$

$$C - NOT\,|00\rangle \rightarrow |00\rangle$$
$$C - NOT\,|01\rangle \rightarrow |01\rangle$$
$$C - NOT\,|10\rangle \rightarrow |11\rangle$$
$$C - NOT\,|11\rangle \rightarrow |10\rangle$$

Ela é equivalente à aplicação de uma porta X apenas quando o qbit de controle for $|1\rangle$:

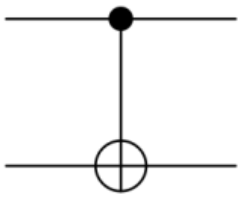


Figura 1.7: Representações da porta C-NOT

Na notação acima $\oplus$ representa a soma módulo 2. Operações com porta controladas podem usar qualquer porta - não apenas a X - desde que elas sejam unitárias. E também não são restritas a apenas dois qbits de entrada, como por exemplo a porta Toffoli.

A porta Toffoli é uma porta lógica universal, ou seja ela é útil para operações em circuitos clássicos, e por ser uma porta reversível, também é válida para circuitos quânticos. É bastante semelhante à porta C-NOT, diferindo por possuir um qbit de controle a mais. A porta Toffoli é bastante conveniente pois pode realizar operações lógicas de AND, XOR, NOT e até FANOUT, dependendo das entradas iniciais. Sua representação circuital é visualizada na Figura 1.8.

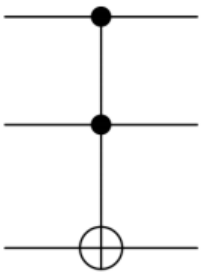


Figura 1.8: Representações da porta Toffoli

Na notação acima $\oplus$ representa a soma módulo 2. Caso os dois qbits de controle sejam $|1\rangle$, então a amplitude do qbit alvo é trocada.

Porta U_f

É uma porta muitas vezes chamada de "black box". É uma porta "genérica" onde se pode implementar qualquer operação, desde que esta obedeça às regras dos operadores unitários. A Figura 1.9 é a sua representação circuital. Ela é pré-definida da seguinte forma: $U_f\,|x, y\rangle = |x, y \oplus f(x\rangle$, sendo $f(x)$ um função definida e $\oplus$ a soma módulo 2 dos qbits.

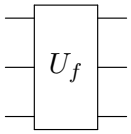


Figura 1.9: Representações da porta genérica Black Box

1.3.4 Circuitos Quânticos

A computação quântica de circuitos é análoga ao modelo computacional clássico baseado em circuitos. Dessa forma, a computação é realizada através de portas lógicas quânticas que executam uma sequência de transformações unitárias nos qbits. Algumas dificuldades inerentes ao hardware consistem em controlar os qbits, sendo necessário implementar algoritmos de correção de falhas que consequentemente precisarão de mais qbits extra.

Os circuitos quânticos são compostos por portas lógicas e "fios". Os fios não são necessariamente os objetos físicos costumeiros, eles apenas representam o transporte do qbit através do circuito. O fio pode representar um fóton, ou outra partícula, se movendo de um local para outro no espaço [18].

A computação quântica, assim como a clássica, manipula sua informação através de portas lógicas. Os circuitos quânticos fornecem uma forma simples para que se possa compreender o funcionamento de um algoritmo complexo ou mesmo uma operação de sobreposição. A Figura 1.10 é um exemplo de circuito quântico.

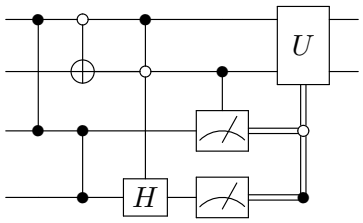


Figura 1.10: Exemplo de um circuito quântico

Capítulo 2

Algoritmos Quânticos de Busca

2.1 Algoritmos Quânticos

Um algoritmo quântico é um conjunto de instruções feitas através de consecutivas operações reversíveis (operadores unitários), seguidas por uma medição probabilística. Esses algoritmos funcionam de uma forma drasticamente diferente de algoritmos clássicos. Uma das principais características do paradigma quântico é o uso da superposição de estados. Um sistema quântico pode coexistir simultaneamente entre todos seus estados possíveis, ao contrário de um sistema clássico, onde é permitido apenas uma configuração bem definida em um dado instante. Uma outra característica especial é o paralelismo quântico. Em poucas palavras, o paralelismo quântico permite que um computador avalie uma função $f(x)$ para diferentes valores de x ao mesmo tempo [18]. Um dos primeiros algoritmos quânticos desenvolvidos, o algoritmo de Deutsch-Josza [4], abriu a possibilidade de que computadores quânticos seriam capazes de resolver problemas mais eficientes que um computador clássico. Algoritmos quânticos podem ser subdivididos em três tipos, segundo [18]. São eles: algoritmos que fazem uso da versão quântica da transformada de Fourier, como o próprio algoritmo de Deutsch-Josza e o algoritmo de Shor. Algoritmos de simulação de um sistema quântico. E por fim, algoritmos quânticos de busca, onde o mais conhecido exemplo é o algoritmo de Grover.

2.2 Algoritmo de Grover

O algoritmo de Grover realiza uma busca em um espaço desordenado de $N = 2^n$ itens, onde n é a quantidade de qbits, até achar o elemento desejado. O melhor algoritmo clássico leva no mínimo $\mathcal{O}(N)$ etapas para concluir a busca. Em contrapartida, o algoritmo de Grover, em um computador quântico, a realiza em $\mathcal{O}\sqrt{N}$ etapas, entregando um ganho quadrático de velocidade.

Além do paralelismo quântico, Grover utiliza a superposição de estados e a grande ideia do algoritmo é a mudança da probabilidade de cada elemento do espaço buscado, sem interferir no entanto no seu valor. Grover explora a propriedade de amplificação de amplitude para encontrar o elemento buscado.

2.2.1 Funcionamento

Grover utiliza dois registradores a principio, com n qbits no primeiro e 1 qbit no segundo. O algoritmo começa ao inicializar todos os n qbits do registrador 1 no estado $|0\rangle$, e o registador 2 no estado $|1\rangle$:

$$|\varphi_1\rangle = |0...0\rangle$$
$$|\varphi_2\rangle = |1\rangle$$

Iniciado os registradores, aplica-se a porta Hadamard em todos os n estados possíveis do primeiro registrador, os deixando em igual superposição, isto é, em uma igual possibilidade de observação:

$$H^{\otimes n} |\varphi_1\rangle = H^{\otimes n} |0\rangle^{\otimes n} = |\psi\rangle$$

$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle$$

O segundo registrador, iniciado no estado $|1\rangle$, tem a seguinte configuração após a aplicação de H:

$$H |\varphi_2\rangle = H |1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} = |-\rangle$$

Os próximos passos são conhecidos como *Interação de Grover*, e é comumente referido como apenas G. Usaremos está notação posteriormente. Iniciamos com a chamada da função Oráculo (O). Essa função é capaz de identificar o elemento procurado na lista. Para entender O, definimos uma função $f : \{0...N-1\} \to \{0, 1\}$, onde:

$$f(x) = \begin{cases} 1, & \text{se } x \text{ for o estado procurado} \\ 0, & \text{caso contrário} \end{cases} \tag{2.1}$$

Não podemos utilizar f diretamente em um estado quântico. É necessário um operador unitário que podemos definir como dependente de f da seguinte forma:

$$U_f = |x\rangle |y \oplus f(x)\rangle ,$$

onde $|x\rangle$ representa o estado do primeiro registrador, $|y\rangle$ representa o estado do segundo registrador, e $\oplus$ é a soma módulo 2.

O oráculo irá marcar o estado procurado negando sua amplitude caso o qbit avaliado esteja no estado desejado. Senão, nada é modificado. Como a probabilidade de observação de um qbit é dada pela norma de sua amplitude ao quadrado, negar a amplitude não afetará esse valor. A aplicação do oráculo, tido agora pelo operador unitário U_f, é exemplificada por:

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

Vemos um exemplo do uso do paralelismo quântico, dado que a função U_f é aplicada a todos estados do espaço procurado em uma única chamada de U_f. A implementação de U_f no entanto difere para cada problema especifico.

As etapas de *inversão sobre a média* e *inversão de fase*, combinadas e repetidas consecutivamente, ocasionarão no aumento de amplitude do elemento procurado, e diminuição das amplitudes dos demais itens do espaço. Novamente é aplicado Hadamard, e em seguida o operador unitário $(2\left|0\right\rangle\left\langle 0\right| - I)$, e por fim mais uma aplicação de Hadamard:

$$H^{\otimes n}(2\left|0\right\rangle\left\langle 0\right| - I)H^{\otimes n}$$
$$2H^{\otimes n}\left|0\right\rangle\left\langle 0\right|H^{\otimes n} - I,$$

e como sabemos que

$$\left|\psi\right\rangle = H^{\otimes n}\left|0\right\rangle^{\otimes n},$$

$$H^{\otimes n}(2\left|0\right\rangle\left\langle 0\right| - I)H^{\otimes n} = (2\left|\psi\right\rangle\left\langle\psi\right| - I) \tag{2.2}$$

Finalmente, a interação de Grover é dada por:

$$G = (2\left|\psi\right\rangle\left\langle\psi\right| - I)U_f \tag{2.3}$$

A interação de Grover será repetida r vezes, sendo $r = \frac{\pi}{4}\sqrt{2^n}$. Então, é feita uma medição no sistema que constará em um alta probabilidade da solução ser observada. A Figura 2.1 demonstra o esquema geral de funcionamento do algoritmo de Grover.

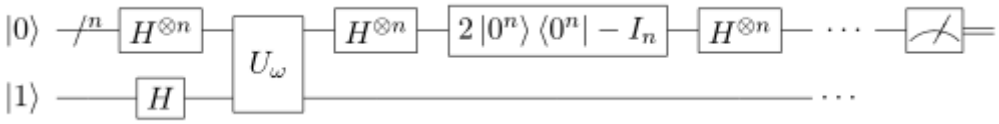


Figura 2.1: Esquema circuital do algoritmo de Grover

2.2.2 Interpretação Geométrica

Supondo que há um espaço de busca N com [0...N-1] elementos, e um subespaço $M \in N$ de soluções, sendo t o número de soluções no espaço.

Começamos por definir os seguintes vetores:

$$\left|\alpha\right\rangle = \frac{\sqrt{N-t}}{N}\sum_x \left|x\right\rangle, \tag{2.4}$$

$$\left|\beta\right\rangle = \left|x_0\right\rangle, \tag{2.5}$$

onde $|\alpha\rangle$ representa a somatória de todos os estados que não são soluções, e β o estado procurado. Caso haja um número t de soluções, $|\beta\rangle$ seria representado por $\frac{1}{\sqrt{t}}\sum_{x'}|x'\rangle$. Vemos então que:

$$|\psi\rangle = \frac{1}{\sqrt{N-t}}|\alpha\rangle + \frac{1}{\sqrt{N}}|\beta\rangle \tag{2.6}$$

Ao aplicarmos a função de Oráculo O no estado $|\psi\rangle$, notamos que ocorre uma reflexão:

$$O|\psi\rangle = O\left[\frac{1}{\sqrt{N-t}}|\alpha\rangle + \frac{1}{\sqrt{N}}|\beta\rangle\right] \tag{2.7}$$

$$\frac{1}{\sqrt{N-t}}|\alpha\rangle - \frac{1}{\sqrt{N}}|\beta\rangle \tag{2.8}$$

Em seguida, o operador unitário $(2|\psi\rangle\langle\psi| - I)$ é aplicado ao estado genérico $|\phi\rangle$, o que resulta em mais uma reflexão sobre o estado $|\psi\rangle$. Das propriedades da álgebra linear, o resultado do produto de duas reflexões ocasiona em uma rotação. Ou seja, a aplicação de G, r vezes sobre um estado analisado, provoca sucessivas rotações do vetor. Esse processo é melhor exemplificado na Figura 2.2.

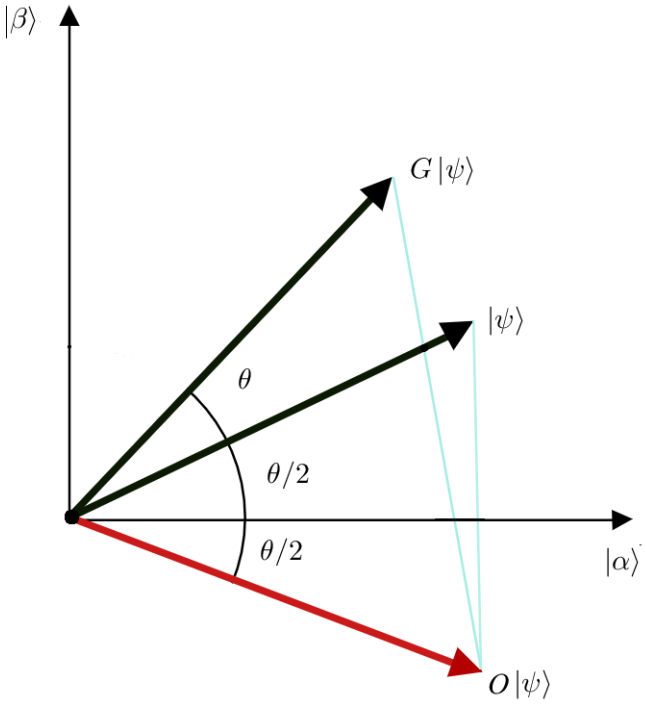


Figura 2.2: Rotações do Grover

Fazendo um pequeno truque geométrico ao definir $\cos\left(\frac{\theta}{2}\right) = \sqrt{\frac{N-1}{N}}$, podemos reescrever $|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|\alpha\rangle + \sin\left(\frac{\theta}{2}\right)|\beta\rangle$. Quando aplicamos G em $|\psi\rangle$ temos:

$$G|\psi\rangle = \cos\left(\frac{3\theta}{2}\right)|\alpha\rangle + \sin\left(\frac{3\theta}{2}\right)|\beta\rangle \tag{2.9}$$

E para um número k de rotações temos:

$$G\left|\psi\right\rangle = \cos\left(\frac{2k+1}{2}\theta\right)\left|\alpha\right\rangle + \sin\left(\frac{2k+1}{2}\theta\right)\left|\beta\right\rangle \tag{2.10}$$

Olhando novamente para Figura 2.2 fica evidente que estamos realizando rotações de $|\psi\rangle$ até a aproximação no eixo $|\beta\rangle$. É necessário realizar um número ótimo de vezes essa rotação, caso contrário $|\psi\rangle$ passará do eixo desejado $|\beta\rangle$. Realizamos então r rotações:

$$r = IP\left(\frac{\arccos\sqrt{\frac{t}{N}}}{\theta}\right) \tag{2.11}$$

,

sendo IP o *inteiro mais próximo*.

É perceptível que $r \leq \lceil \frac{\pi}{2\theta} \rceil$, e podemos limitar um valor para r. Ainda é possível notar que

$$\frac{\theta}{2} \geq \sin\left(\frac{2k+1}{\theta}\right) = \sqrt{\frac{t}{N}}, \tag{2.12}$$

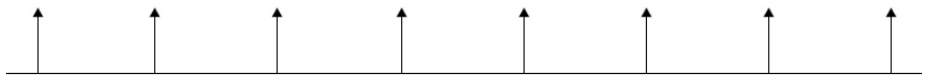
resultando em $r \leq \left\lceil \frac{\pi}{4}\sqrt{\frac{N}{t}} \right\rceil$, provando então o ganho quadrático do algoritmo de Grover ao realizar uma busca em $\sqrt{N}$ etapas.

2.2.3 Exemplo

Para exemplificar o funcionamento do algoritmo de Grover, é tido um espaço com N elementos e 3 qbits, isto é $2^3 = 8 = N$. Para simplicidade, definiremos $t = 1$ e estado buscado $x_0 = |011\rangle$. Em um sistema completo de 3 bits, o estado genérico é representado por:

$$\left|x\right\rangle = \alpha_0\left|000\right\rangle + \alpha_1\left|001\right\rangle + \alpha_2\left|010\right\rangle + \alpha_3\left|011\right\rangle + \alpha_4\left|100\right\rangle + \alpha_5\left|101\right\rangle + \alpha_6\left|110\right\rangle + \alpha_7\left|111\right\rangle \tag{2.13}$$

Como visto na Seção 2.2.1, os estados são iniciados em $|000\rangle$, seguido da aplicação de Hadamard a fim de gerar amplitudes iguais para todos estados de $|x\rangle$. As amplitudes dos estados podem ser representadas graficamente por linhas cujo comprimento representa sua magnitude. A aplicação de Hadamard resulta na seguinte configuração do sistema:



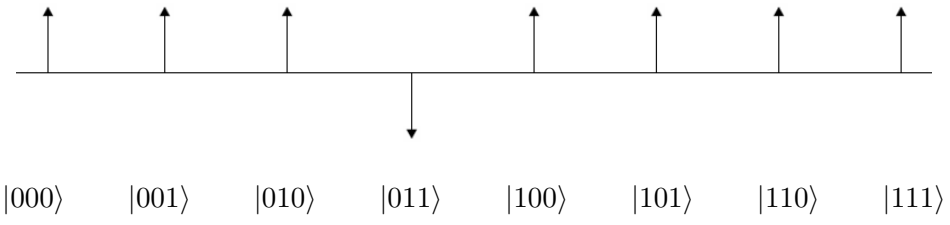
$|000\rangle$ $|001\rangle$ $|010\rangle$ $|011\rangle$ $|100\rangle$ $|101\rangle$ $|110\rangle$ $|111\rangle$

$$H^{\otimes n}\left|000\right\rangle = \frac{1}{\sqrt{8}}\sum_{x=0}^{7}\left|x\right\rangle = \left|\psi\right\rangle \tag{2.14}$$

Como está sendo avaliado um sistema com 3 qbits e apenas uma solução (t = 1), é necessário realizar $r = \frac{\pi}{4}\sqrt{8} \approx 2.23$ iterações, cujo inteiro mais próximo é 2.

Inicia-se a *Interação de Grover*. Primeiro é chamado o oráculo O, que negará a amplitude do estado procurado $x_0 = |011\rangle$:

$$|x\rangle = \frac{1}{2\sqrt{2}}|000\rangle + \frac{1}{2\sqrt{2}}|001\rangle + \frac{1}{2\sqrt{2}}|010\rangle - \frac{1}{2\sqrt{2}}|011\rangle + \frac{1}{2\sqrt{2}}|100\rangle + \frac{1}{2\sqrt{2}}|101\rangle + \frac{1}{2\sqrt{2}}|110\rangle + \frac{1}{2\sqrt{2}}|111\rangle$$



$$H^{\otimes n}|000\rangle = \frac{1}{\sqrt{8}}\sum_{x=0}^{7}|x\rangle = |\psi\rangle \tag{2.15}$$

Aplica-se agora o operador unitário $(2|\psi\rangle\langle\psi| - I)$, realizando a inversão sobre a média, o que afetará a amplitude dos estados.

$$(2|\psi\rangle\langle\psi| - I)|x\rangle$$

$$(2|\psi\rangle\langle\psi| - I)\left(|\psi\rangle - \frac{2}{2\sqrt{2}}|011\rangle\right)$$

$$2|\psi\rangle\langle\psi|\psi\rangle - |\psi\rangle - \frac{2}{\sqrt{2}}|\psi\rangle\langle\psi|011\rangle + \frac{1}{\sqrt{2}}|011\rangle$$

$$2|\psi\rangle - |\psi\rangle - \frac{2}{\sqrt{2}}|\psi\rangle\left(\frac{1}{2\sqrt{2}}\right) + \frac{1}{\sqrt{2}}|011\rangle$$

$$\frac{1}{2}|\psi\rangle + \frac{1}{\sqrt{2}}|011\rangle \tag{2.16}$$

Da equação 2.14, temos:

$$\frac{1}{2}\left(\frac{1}{\sqrt{8}}\sum_{x=0}^{7}|x\rangle\right) + \frac{1}{\sqrt{2}}|011\rangle$$

$$\frac{1}{4\sqrt{2}}\sum_{x\neq 3}^{7}|x\rangle + \frac{1}{4\sqrt{2}}|011\rangle + \frac{1}{\sqrt{2}}|011\rangle$$

$$\frac{1}{4\sqrt{2}}\sum_{x\neq 3}^{7}|x\rangle + \frac{5}{4\sqrt{2}}|011\rangle \tag{2.17}$$

Expandindo os estados de $|x\rangle$:

$$|x\rangle = \frac{1}{4\sqrt{2}}|000\rangle + \frac{1}{4\sqrt{2}}|001\rangle + \frac{1}{4\sqrt{2}}|010\rangle - \frac{5}{4\sqrt{2}}|011\rangle + ... + \frac{1}{4\sqrt{2}}|111\rangle$$

$|000\rangle$ $|001\rangle$ $|010\rangle$ $|011\rangle$ $|100\rangle$ $|101\rangle$ $|110\rangle$ $|111\rangle$

Com o aumento da amplitude do elemento procurado, e diminuição das amplitudes dos demais itens, a primeira interação é terminada. Como visto no início desta seção, é necessário duas interações para que o algoritmo tenha desempenho ótimo.

Aplica-se o Oráculo, negando a amplitude de $x_0 = |011\rangle$:

$$|x\rangle = \frac{1}{4\sqrt{2}}|000\rangle + \frac{1}{4\sqrt{2}}|001\rangle + \frac{1}{4\sqrt{2}}|010\rangle - \frac{5}{4\sqrt{2}}|011\rangle + ... + \frac{1}{4\sqrt{2}}|111\rangle$$

$$\frac{1}{4\sqrt{2}}\sum_{x\neq 3}^{7} -\frac{5}{4\sqrt{2}}|011\rangle - \frac{1}{4\sqrt{2}}|011\rangle$$

$$\frac{1}{4\sqrt{2}}\sum_{x\neq 3}^{7} -\frac{3}{2\sqrt{2}}|011\rangle$$

$$\frac{1}{2}|\psi\rangle - \frac{3}{2\sqrt{2}}|011\rangle \tag{2.18}$$

Aplica-se o operador unitário $(2|\psi\rangle\langle\psi| - I)$:

Da expansão de $|x\rangle$ vem:

$$|x\rangle = -\frac{1}{8\sqrt{2}}|000\rangle - \frac{1}{8\sqrt{2}}|001\rangle - \frac{1}{8\sqrt{2}}|010\rangle + \frac{11}{8\sqrt{2}}|011\rangle - ... - \frac{1}{8\sqrt{2}}|111\rangle$$

$|000\rangle$ $|001\rangle$ $|010\rangle$ $|011\rangle$ $|100\rangle$ $|101\rangle$ $|110\rangle$ $|111\rangle$

Com o fim da segunda e última interação, o algoritmo chega ao final. A partir disso, realizar uma medida no sistema garante uma probabilidade de $|\frac{11}{8\sqrt{2}}|^2 \approx 94.6\%$ da solução correta, $x_0 = |011\rangle$ ser observada. E deixando com $|-\frac{7}{8\sqrt{2}}|^2 \approx 4.4\%$ a probabilidade de uma medição errada. A eficiência do algoritmo de Grover tende a crescer quando N se torna maior, ocasionando em uma taxa de erro extremamente pequena.

2.2.4 Limitações

A probabilidade de observar t itens marcados após r rotações de Grover é dada por:

$$g_r(p) = \sin^2\left[(2r+1)\arcsin\sqrt{\frac{N}{t}}\right], \tag{2.19}$$

No entanto, o algoritmo de Grover apresenta claras limitações desde a sua publicação e diversos trabalhos buscaram diferentes formas de resolução para tais lacunas. A primeira é que nem sempre o número de soluções t é conhecido antes do inicio do algoritmo. Esse cenário não é muito realístico, tendo em vista que problemas de buscas não possuem essa informação geralmente. Podemos combinar a estratégia de Grover com um algoritmo de *contagem quântica*, a fim de estimar um valor para t e então ter uma algoritmo de busca mais robusto e próximo à realidade. O segundo surge ainda que t seja conhecido. Note que a Eq. 2.19 possui comportamento sinodal, isso resulta em valores reais para r que no entanto, deverá possuir um valor inteiro. Uma aproximação incorreta para o inteiro mais próximo pode ocasionar resultados insatisfatórios ou até incorretos. Algoritmos que usam a busca de Grover precisam lidar com esses problemas. Nas seguintes seções são apresentadas otimizações ao algoritmo de Grover, cujo foco se volta para o cenário de múltiplas soluções. Também são mencionados problemas que utilizam Grover como subrotina para resolução de problemas de otimização global.

2.3 Generalizações e Otimizações

Baseando-se nas limitações do Grover, alinharemos o foco desta pesquisa a partir de agora para a análise de múltiplas soluções, um cenário que, apesar de mencionado, não foi proposto por Grover em seu algoritmo original. Visto isso, diversas pesquisas foram dedicadas à análise e generalização de seu algoritmo para o caso onde $t > 1$ [21, 22, 23]. Realizaremos uma breve análise sobre estes trabalhos e outras generalizações onde o algoritmo de Grover é utilizado.

2.3.1 Múltiplas soluções

Iniciaremos o desenvolvimento de múltiplas soluções através de um outro fator de grande importância no algoritmo de Grover: o número de iterações r. Em [9] é proposto equações para o cálculo de amplitude na *i-ésima* iteração. Dado que $k_0 = l_0 = \frac{1}{\sqrt{N}}$, e sendo k_i a amplitude do estado procurado e l_i a amplitude dos outros estados, então:

$$k_{i+1} = \frac{N-2}{N}k_i + \frac{2(N-1)}{N}l_i \tag{2.20}$$

$$l_{i+1} = \frac{N-2}{N}l_i - \frac{2}{N}k_i \tag{2.21}$$

Essas equações podem ainda ser escritas em função do ângulo θ definido por $\sin^2\theta = \frac{1}{N}$:

$$k_i = \sin(2j+1)\theta \tag{2.22}$$

$$l_i = \frac{1}{\sqrt{N-1}} \cos(2j+1)\theta \tag{2.23}$$

Foi calculado que o número ótimo de iterações r deverá ser $\frac{\pi}{4}\sqrt{N}$ para $t = 1$. Intuitivamente pode-se pensar que aumentando o número de iterações, ocasionará um aumento da taxa de sucesso. No entanto, isso não é verdade. Grover explicitou que após i iterações, a probabilidade de sucesso deverá ser no mínimo 50%, o que é o mesmo valor após a aplicação de metade do número ótimo de rotações, $\frac{\pi}{8}\sqrt{N}$. Dobrando r, ou seja $r = \frac{\pi}{2}\sqrt{N}$, resultará no decaimento da taxa de sucesso.

Agora supondo que haja t soluções, nos quais as amplitudes dos itens marcados seja k e dos não marcados seja l. Analogamente ao que é feito para uma única solução, as amplitudes k_i e j_i na *i-ésima* iteração são:

$$k_i = \frac{1}{\sqrt{t}} \sin(2j+1)\theta \tag{2.24}$$

$$l_i = \frac{1}{\sqrt{t}} \cos(2j+1)\theta \tag{2.25}$$

Assim como é feito para t unitário, o objetivo é elevar os valores de k à medida que se reduz l ao máximo possível. O tempo de execução desse algoritmo é $\mathcal{O}\sqrt{\frac{N}{t}}$. Portanto similar a proposta inicial de Grover, adicionando o parâmetro t. Por consequência, o número ótimo de rotações será agora $\frac{\pi}{4}\sqrt{\frac{N}{t}}$.

2.3.2 Algoritmo BBHT

Em 1996, Boyer, Brassard, Hoyer e Tapp elaboraram uma série de otimizações do algoritmo de Grover, o que nomearemos de algoritmo BBHT [9]. Como discutido anteriormente, uma das limitações do algoritmo de Grover é não saber previamente quantas soluções buscar. Uma das propostas do BBHT é unir técnicas de fatoração quântica proposta por Shor [5] juntamente com o algoritmo de Grover para estimar o número de soluções. O objetivo neste caso é saber o número ótimo de rotações pois se o algoritmo executar menos ou mais rotações que o devido, a probabilidade de encontrar os elementos corretos decresce. O primeiro passo é realizar um número pequeno de iterações G seguido de uma medição do elemento $x \in 0...N-1$, e então verifica-se se *f(x) = 1*. Caso a condição seja falsa, uma nova iteração do *loop* é feita, atualizando o número de rotações para um valor maior.

A fim de encontrar uma ou mais soluções quando não se sabe sua quantidade exata, é necessário partir de duas propostas provadas algebricamente, explicadas com maior detalhe em [9]:

Teorema 1: Para qualquer número inteiro k, e quaisquer números reais a e b, da expansão geométrica vêm:

$$\sum_{j=0}^{k-1} \cos a + 2jb = \frac{[\sin(kb)][\cos a + b(k-1)]}{\sin b} \tag{2.26}$$

$$\sum_{j=0}^{k-1} \cos a(2k+1) = \frac{\sin(2ka)}{2\sin a} \tag{2.27}$$

Teorema 2: Seja M o número de soluções procuradas, t o conjunto de elementos do espaço buscado, k um inteiro positivo, e θ um ângulo tal que

$$\sin\theta^2 = M/t, \tag{2.28}$$

ao realizar uma medição no sistema após r interações de Grover, sendo $r \leq k-1$, a probabilidade de uma solução ser observada é:

$$P_k = \frac{1}{2} - \frac{\sin(4k\theta)}{4k\sin(2\theta)} \tag{2.29}$$

O algoritmo BBHT segue os seguintes passos:

1. Inicializa-se a constante $m = 1$ e o parâmetro $0 < \lambda \leq \frac{8}{7}$

2. Escolhe-se um inteiro r aleatoriamente, tal que $0 \leq r < m$

3. Aplica-se r interações de Grover e observa-se a saída x do registrador.

i. Caso $f(x) = 1$, o algoritmo retorna x.

ii. Caso contrário, é estabelecido um novo valor para m, dado por:

$$m = min(\lambda m, \sqrt{N}), \tag{2.30}$$

e volta-se ao passo 2.

Esse algoritmo funcionará apenas quando $1 \leq t \leq \frac{3N}{4}$, e para $t > \frac{3N}{4}$ é sugerido técnicas clássicas de amostragem. Caso t seja desconhecido, há uma forma de estimar o seu valor combinando o algoritmo de Grover com o algoritmo de fatoração quântica de Shor [24]. Ele é comumente referido por *Contagem quântica*. Esse algoritmo não será desenvolvido neste trabalho, mas é encorajada a leitura de [25] e [26] para um melhor entendimento do tema.

2.3.3 Outras generalizações

O algoritmo de Grover é frequentemente utilizado como subrotina na otimização global de algoritmos quânticos. Ele é um algoritmo do tipo amplificador de amplitudes, através de r iterações do operador G, ocasionando finalmente numa alta probabilidade de observação do estado correto. É evidente a importância de um valor ótimo para r, dado que algum valor diferente desse limite, ocasiona no decaimento da probabilidade do algoritmo encontrar o elemento marcado.

Bulger, Baritompa e Wood propuseram uma implementação do algoritmo de busca adaptativa pura (Pure adaptive search - PAS [27]) utilizando o paradigma de busca quântica publicados por Grover [28]. Os autores uniram os dois conceitos para propor uma nova heurística de otimização global, a Busca Adaptativa de Grover - do inglês Grover Adaptive Search (GAS). Baseia-se na escolha de um valor fixo para r para todas as iterações de Grover necessárias.

Durr e Hoyer elaboraram um algoritmo que visa encontrar o mínimo valor x em uma espaço desordenado N [8], utilizando o método de busca de Grover e a otimização proposta no algoritmo BBHT [9]. Têm-se um espaço de itens T[0...N-1], o algoritmo retornará o índice i onde $T[i] = x$, tal que x seja o menor valor entre os itens do espaço T. O primeiro passo é escolher ao acaso um elemento y da lista T e o tem como menor elemento do

espaço, em seguida é realizada a busca por um elemento y' tal que $y' \in T : y' < y$ através da chamada do BBHT. Caso $y' < y$, y' é então tido como menor valor da lista e o BBHT é chamado novamente. Este processo é repetido até que a probabilidade do valor mínimo ser encontrado seja maior que $\frac{1}{2}$.

Baritompa, Bulger e Wood, baseando-se no algoritmo de Durr e Hoyer, propuseram um novo método de otimização quântica global mais genérico (BBW) para uma função $f : 0...N-1$ e uma lista de elementos finita $T[0...N-1]$ [10]. Os autores fizeram uso das ideias de busca adaptativa na elaboração de um esquema para computar o número de rotações a serem feitas a cada iteração de G. Em outras palavras, há uma lista L tal que o valor do elemento $L[i]$ armazena o número de rotações r para a *i-ésima* chamada de G.

Liu e Koehler utilizaram o método de busca BBW e desenvolveram duas importantes modificações. A primeira se trata de promover um ganho de velocidade em tempo de execução que permitiu aumentar os pontos antes sugeridos pelo BBW de 33 para 43, aumentando assim sua aplicabilidade [11]. Essa modificação também buscou evitar cálculos desnecessários utilizando os polinômios de Chebyshev para calcular distribuições. Já a segunda modificação converte o algoritmo de um método estático para um dinâmico. A segunda melhoria utilizou as lei de Bayes para atualizar a distribuição após cada iteração de Grover, dependendo se a pesquisa falhou ou conseguiu encontrar um ponto de melhoria.

Lara, Portugal e Lavor apresentaram um novo método híbrido, que utiliza um algoritmo clássico para encontrar um mínimo local e um algoritmo quântico (GAS) [12]. As simulações numéricas mostraram que o DHB, o BBW e o novo método têm comportamento assintótico muito diferente, onde o novo método apresentou melhor desempenho.

Em [29] é proposto um algoritmo quântico otimizado para buscar máximos e mínimos locais, tomando como base o método proposto por Durr e Hoyer (corrigido por Baritompa), o DHB. Em um cenário em que se pode estimar a razão entre o número de soluções m e o espaço pesquisado N, o método proposto pode melhorar a probabilidade de sucesso em números significativos. O algoritmo mostra uma vantagem em relação ao DHB na complexidade de busca em grandes bancos de dados e na complexidade de construção de oráculos.

Na proposta original do algoritmo de Grover, uma busca precisa, de modo que a probabilidade de sucesso de observação dos itens marcados seja infinitesimalmente próxima à 100%, é possível apenas para determinados valores da razão $\frac{m}{N}$ em que m é o número de itens marcados em um banco de dados de N itens. Existem vários algoritmos modificados com uma ou mais ajustes de fase. Entre eles, o algoritmo proposto por Long é o mais simples no sentido de que possui apenas uma fase ajustável [30]. Em se mostra que outros algoritmos com fases adicionais não são mais eficientes do que a versão de Long com uma fase única [31].

Capítulo 3

Experimentos e Simulações

Neste capítulo é explicado as escolhas de linguagem, bibliotecas e frameworks utilizados no desenvolvimento e simulações do algoritmos revisados nesse trabalho. Por fim, é apresentado os experimentos realizados com base nas ferramentas escolhidas e análise comparativa de seus resultados.

3.1 Metodologia

3.1.1 Linguagem

Python foi a linguagem utilizada por possuir todos os requisitos necessários para o desenvolvimento do projeto, além de contar com bibliotecas como o QuTip e QisKit, para desenvolvimento e simulação de algoritmos e circuitos quânticos. Aqui serão usadas ideias e técnicas tradicionais de programação para elaboração dos códigos fonte. Python é uma linguagem de fácil entendimento, o que simplificado o trabalho de refatoração dos algoritmos anteriormente vistos e também revisão de conceitos matemáticos e álgebra linear, altamente necessários para o entendimento da computação quântica.

3.1.2 Anaconda

O Anaconda é um software de código aberto para as linguagens *Python* e *R* com foco em computação científica [32]. Visa simplificar o gerenciamento de pacotes e instalação de dependências através de comandos simples. A distribuição do Anaconda conta com mais de 1.500 pacotes, e também inclui uma GUI, o Anaconda Navigator, e uma alternativa gráfica à interface da linha de comandos (CLI).

3.1.3 Jupyter Notebook

O Jupyter Notebook é uma aplicação web de código aberto que permite criar e compartilhar documentos que contêm códigos em tempo real, equações, gráficos e texto narrativo [33]. Neste projeto foi utilizado para simulação numérica dos algoritmos quânticos, tanto para o método local, como para máquinas reais. Também é possível realizar modelagem estatística, plotagem de gráficos, aprendizagem de máquina entre outras atividades.

3.1.4 QisKit

O Qiskit é um software escrito em Python para formular circuitos quânticos. Ele foi desenvolvido para facilitar a interação de qualquer pessoa com um computador quântico bem como seus conceitos de computação quântica. A interface fornece acesso a processadores quânticos reais e também acesso a um simulador local para executar simulações. O simulador pode ser configurado para utilizar por exemplo o ibmqx5, um computador quântico de 16 qbits.

Na elaboração dos circuitos quânticos dos algoritmos previamente descritos nesse projeto, o QisKit oferece uma variedade de portas disponíveis. Usaremos amplamente as de Pauli, Hadarmar, CNOT, Toffoli, T, S e U. Porém, comumente na construção de oráculos, a combinação das mesmas serão vistas como um operador unitário, omitindo todas as portas utilizadas para uma visualização gráfica mais simples.

3.1.5 IBM Q Experience

O IBM Q Experience é uma plataforma online que fornece aos usuários em geral acesso a um conjunto de processadores quânticos por meio da nuvem, fóruns para discutir tópicos relevantes da computação quântica, um conjunto de tutoriais sobre como desenvolver e simular algoritmos quânticos assim como materiais educativos sobre computação quântica [34]. O IBM Quantum Experience tem como objetivo ajudar no aprendizado sobre o mundo quântico lendo o guia de usuário da ferramenta, realizando seus próprios experimentos, simulando-os e executando-os em diferentes processadores quânticos disponíveis na IBM Cloud. Conta com ferramentas úteis como:

- Quantum Composer, uma interface gráfica para criação de circuitos quânticos;
- Quantum Simulator, para testes de algoritmos;
- acesso a processadores quânticos que rodam no laboratório da IBM;
- Quantum Community, um fórum onde ideias e experiências dos usuários são compartilhadas e discutidas.

3.2 Cenário de análise

Nesta seção serão definidos os cenários analisados e posteriormente comparados. Faremos simulações locais, utilizando o QisKit, e em máquina reais, providas pelo serviço de cloud da IBM. Para o algoritmo base, há variações do número de qbits na implementação do Grover: serão analisados os desempenhos para 2 qbits, 3 qbits e por fim 4 qbits. Por consequência, e para simplificar o cenário de análise, o espaço de busca N será um inteiro de base 2, portanto teremos 4, 8 e 16 estados, para respectivamente um Grover de 2, 3 e 4 qbits. Simularemos também a possibilidade múltiplas soluções, com $1 \leq t \leq 3$ para cada configuração do Grover. E por fim, analisaremos o desempenho e precisão dos algoritmos para diferentes *backends* disponibilizados pela IBM. Vale ressaltar a operabilidade do backend à época da escrita deste trabalho. A tabela abaixo resume os parâmetros abordados nesta pesquisa:

Qbits (n)	Soluções (t)	Método de Análise	Máquina real
2	1	Teórico	Vigo
3	2	QisKit	London
4	3	Máquina real	Yorktown
			Ourense

Tabela 3.1: Parâmetros abrangidos nas simulações dos algoritmos

3.3 Implementação

O algoritmo de Grover consiste em quatro etapas, como visto na seção específica de funcionamento do método. O circuito quântico seguirá as mesmas etapas e ordem de implementação. Seu primeiro passo é a aplicação da porta Hadamard em todos os qbits, gerando uma superposição e igualando todas suas probabilidades de observação:

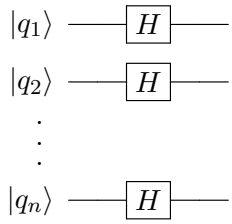


Figura 3.1: Aplicação de H em um circuito de n qbits

Em seguida, apenas os estados buscados serão marcados pelo oráculo, através da negação de sua amplitude. Há diversas maneiras de projetarmos um oráculo. Neste trabalho utilizaremos constantemente portas controladas do tipo Z e Pauli X. Todos oráculos utilizados neste trabalho para marcação de estados podem ser encontrados no apêndice. Na configuração abaixo vemos a construção de um oráculo para o estado $|1100\rangle$ em um Grover de 4 qbits:

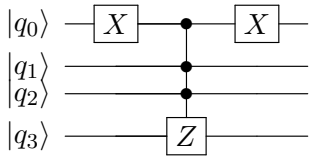


Figura 3.2: Oráculo aplicado no estado $|1100\rangle$

A próxima etapa ampliará as amplitudes dos alvos. Usamos novamente os operadores H e Pauli X, em conjunto com operações controladas tipo Z. Abaixo um exemplo de amplificação:

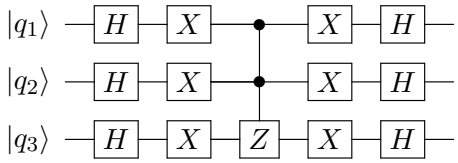


Figura 3.3: Amplificação de amplitude em um Grover de 3 qbits

Finalmente, é tomada uma medição e observado um valor clássico ao término do algoritmo:

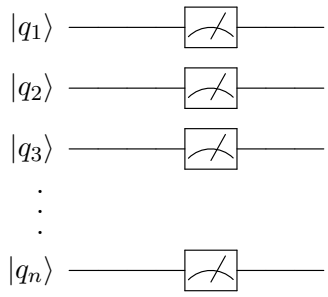


Figura 3.4: Medição ao término do algoritmo

O circuito completo da implementação do algoritmo de Grover e sua otimização BBHT se encontram na seção de apêndice. Em modo geral, o Grover pode ser resumido graficamente por:

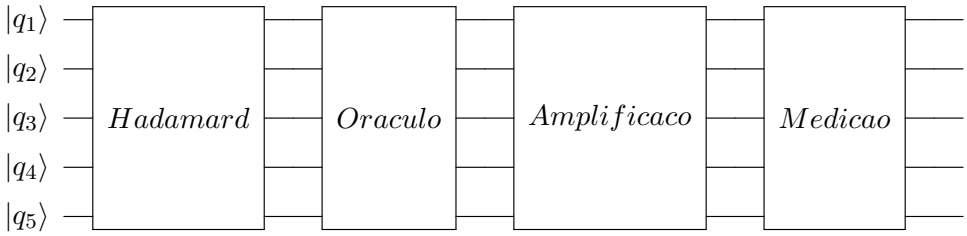


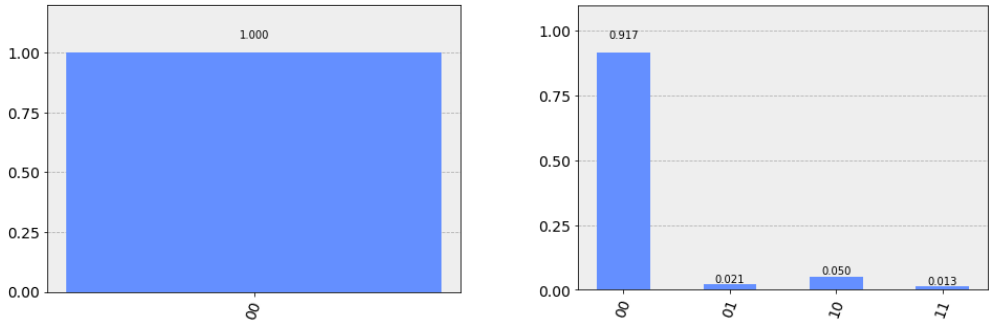
Figura 3.5: Esquema do algoritmo de Grover

3.4 Simulações

Aqui serão detalhados os três diferentes cenários de simulações realizadas no Qis-Kit e nos dispositivos quânticos da IBM. Em cada experimento diferentes parâmetros são alterados, a fim de fazer uma análise robusta de desempenho, precisão e tempo de execução. Além das simulações, também é visto o resultado teoricamente esperado, através dos cálculos previamente definidos na seção 2.2. Todos os resultados são dispostos em tabelas comparativas.

3.4.1 Variação de qbits

Neste cenário variaremos n e analisaremos o desempenho para o Grover de 2 qbits, 3 qbits e 4 qbits. O método teórico consta na realização de cálculos puros e verificação da probabilidade de observação do item marcado ao final das rotações estabelecidas. Fixaremos $t = 1$ neste primeiro caso, e utilizaremos o Vigo como computador quântico. Ele consiste em uma máquina real de 5 qbits, suportando até 75 experimentos e 8192 execuções.

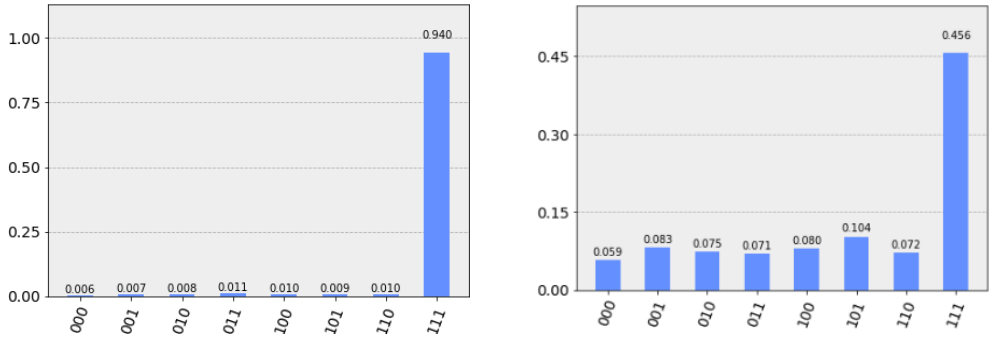


(a) Simulação local (b) Simulação Vigo

Figura 3.6: Simulações Grover 2 qbits para estado $|00\rangle$

Qbits (n)	Soluções (t)	Método	Precisão média	Tempo de execução (s)
2	1	Teórico	100%	-
		QisKit	100.0%	0.011
		Vigo	91.7%	8.33

Tabela 3.2: Grover de 2 qbits, execuções = 1024, Máquina real = Vigo

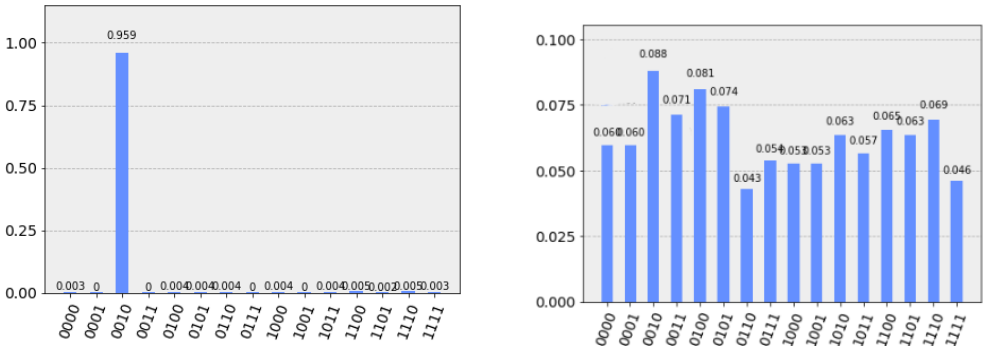


(a) Simulação local (b) Simulação Vigo

Figura 3.7: Simulações Grover 3 qbits para estado $|111\rangle$

Qbits (n)	Soluções (t)	Método	Precisão média	Tempo de execução (s)
3	1	Teórico	95%	-
		QisKit	94%	0.032
		Vigo	45.6%	8.61

Tabela 3.3: Grover de 3 qbits, execuções = 1024, Máquina real = Vigo



(a) Simulação local (b) Simulação Vigo

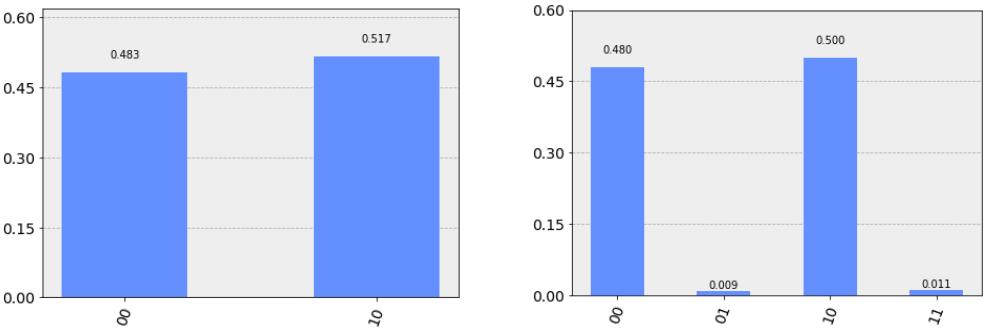
Figura 3.8: Simulações Grover 4 qbits para estado $|0010\rangle$

Qbits (n)	Soluções (t)	Método	Precisão média	Tempo de execução (s)
4	1	Teórico	98%	-
		QisKit	95.9%	0.088
		Vigo	8.8%	8.59

Tabela 3.4: Grover de 4 qbits, execuções = 1024, Máquina real = Vigo

3.4.2 Variação de número de soluções

Agora variaremos t e analisaremos o desempenho para o Grover de 2 qbits, 3 qbits e 4 qbits, analogamente ao experimento anterior. Serão omitidos os cálculos para os valores teóricos, porém estes se encontram no apêndice ao fim deste trabalho. Mais uma vez utilizaremos o *Vigo* como *backend*.

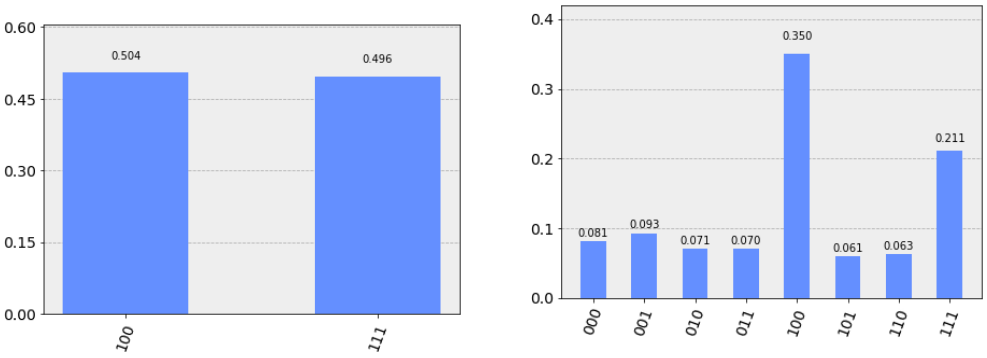


(a) Simulação local (b) Simulação Vigo

Figura 3.9: Simulações Grover 2 qbits para os estados $|00\rangle$ e $|10\rangle$

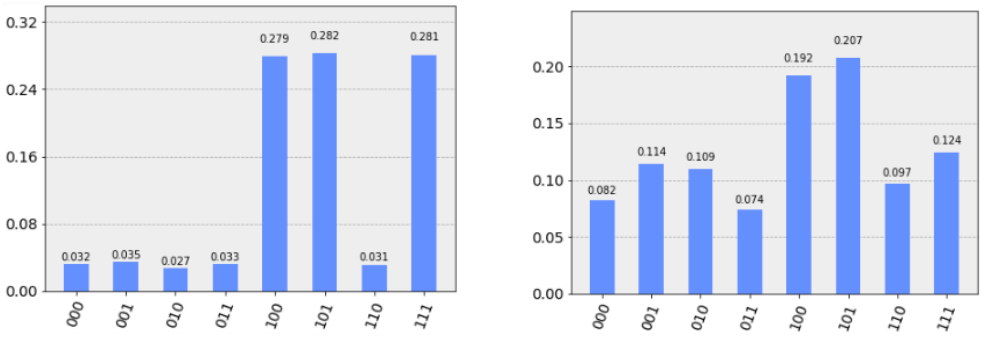
Qbits (n)	Soluções (t)	Método	Precisão	Tempo de execução (s)
2	1	Teórico	100%	-
		Qiskit	100%	0.011
		Vigo	91.7%	8.33
	2	Teórico	100%	-
		Qiskit	100%	0.028
		Vigo	98%	8.29

Tabela 3.5: Resultados de simulações com t variável para um Grover de 2 Qbits e backend Vigo.



(a) Simulação local (b) Simulação Vigo

Figura 3.10: Simulações Grover 3 qbits para os estados $|100\rangle$ e $|111\rangle$

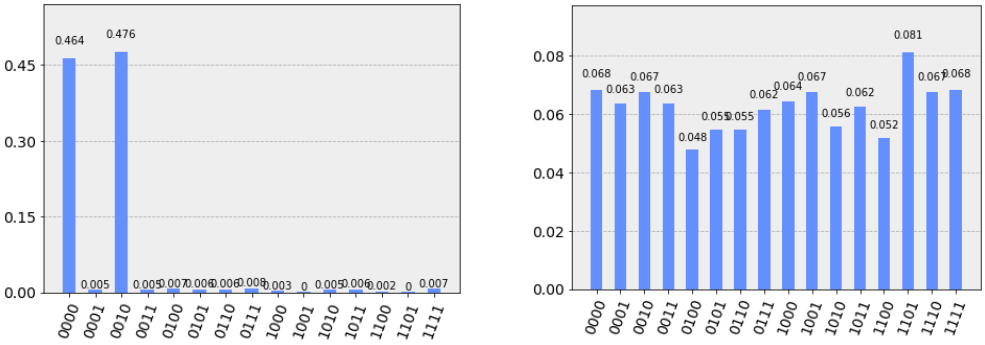


(a) Simulação local (b) Simulação Vigo

Figura 3.11: Simulações Grover 3 qbits para os estados $|100\rangle$, $|101\rangle$ e $|111\rangle$

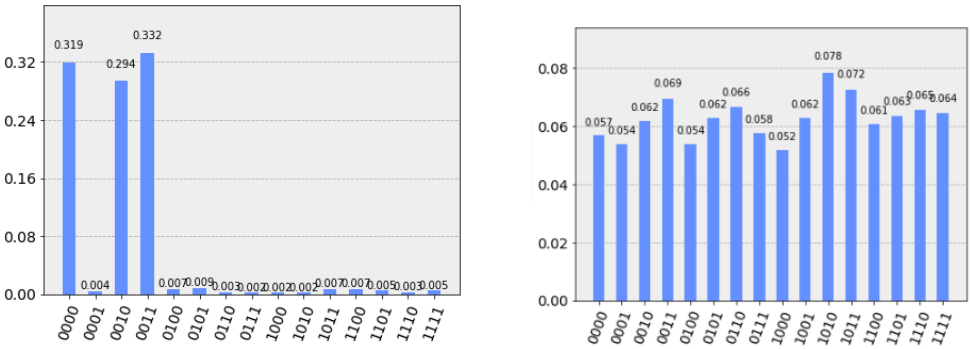
Qbits (n)	Soluções (t)	Método	Precisão	Tempo de execução (s)
3	1	Teórico	94.5%	-
		Qiskit	94%	0.032
		Vigo	51.6%	8.61
	2	Teórico	100%	-
		Qiskit	100%	0.04
		Vigo	56.1%	8.37
	3	Teórico	92%	-
		Qiskit	84.2%	0.14
		Vigo	67.3%	8.31

Tabela 3.6: Resultados de simulações com t variável para um Grover de 3 Qbits e backend Vigo.



(a) Simulação local (b) Simulação Vigo

Figura 3.12: Simulações Grover 4 qbits para os estados $|0000\rangle$ e $|0010\rangle$



(a) Simulação local (b) Simulação Vigo

Figura 3.13: Simulações Grover 4 qbits para os estados $|0000\rangle$, $|0010\rangle$ e $|0011\rangle$

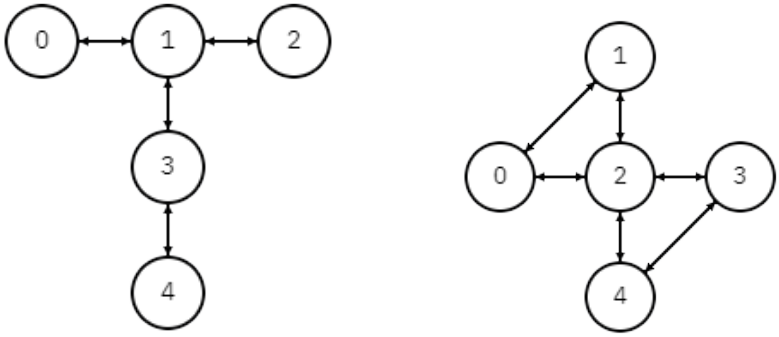
Qbits (n)	Soluções (t)	Método	Precisão	Tempo de execução (s)
4	1	Teórico	98.0%	-
		Qiskit	95.9%	0.088
		Vigo	8.8%	8.59
	2	Teórico	97.2%	-
		Qiskit	94%	0.076
		Vigo	13.5%	8.65
	3	Teórico	97.4%	-
		Qiskit	94.5%	0.082
		Vigo	18.8%	8.71

Tabela 3.7: Resultados de simulações com t variável para um Grover de 4 Qbits e backend Vigo

3.4.3 Variação de máquinas reais

Finalmente variaremos o *backend* e analisaremos o desempenho para o Grover de 2 qbits, 3 qbits e 4 qbits para múltiplas soluções. Aqui não será analisado o resultado teórico, visto que esta simulação é realizada em uma máquina quântica real. Quatro computadores reais foram selecionados para verificação: *Vigo*, *London*, *Yorktown* e *Ourense*. Todos são máquinas de 5 qbits, permitem até 75 experimentos e um máximo de 8192 *execuções*. Três delas possuem a mesma disposição de qbits, porém com taxas de erros diferentes [34]. A

tabela 3.8 resume tais taxas de erros para portas U_3 e $CNOT$, e a Figura 3.14 ilustra a configuração dos qbits nas máquinas.

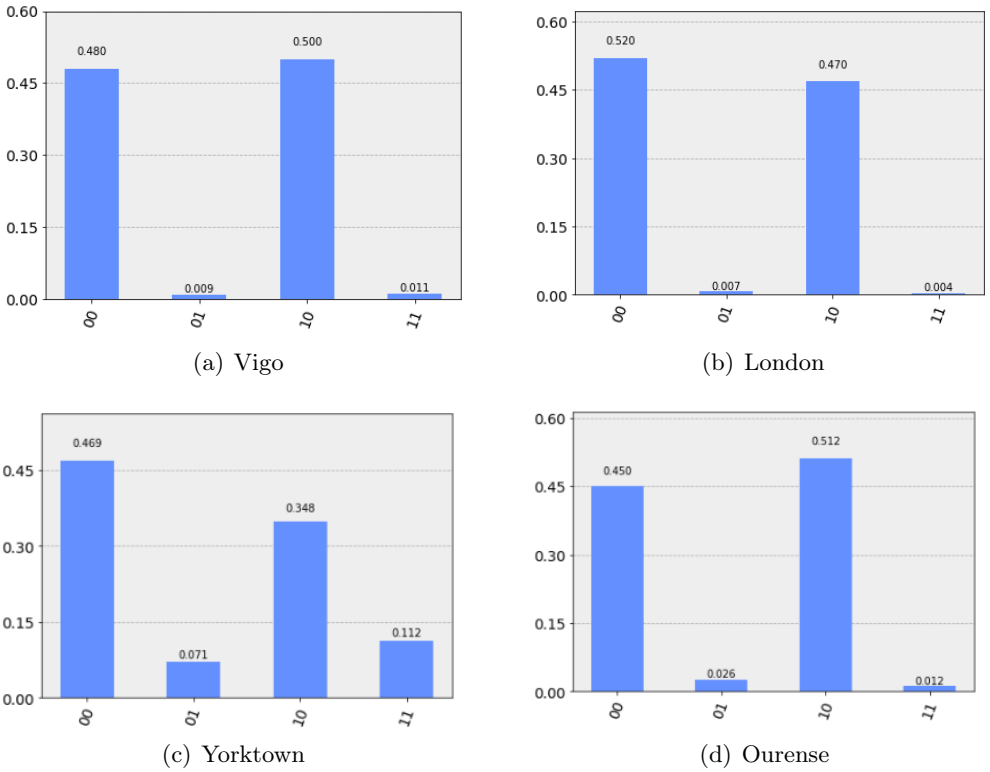


(a) Vigo, London, Ourense (b) Yorktown

Figura 3.14: Disposição dos qbits em diferentes dispositivos da IBM

Máquina real	Taxa de erro U_3	Taxa de erro $CNOT$
Vigo	$6.573e^{-4}$ a $1.264e^{-3}$	$6.865e^{-3}$ a $1.206e^{-2}$
London	$5.702e^{-4}$ a $3.172e^{-3}$	$9.985e^{-3}$ a $4.367e^{-2}$
Ourense	$4.930e^{-4}$ a $6.712e^{-4}$	$5.682e^{-3}$ a $8.441e^{-3}$
Yorktown	$8.455e^{-4}$ a $1.375e^{-3}$	$1.346e^{-2}$ a $2.204e^{-2}$

Tabela 3.8: Especificações dos computadores quânticos escolhidos



(a) Vigo (b) London

(c) Yorktown (d) Ourense

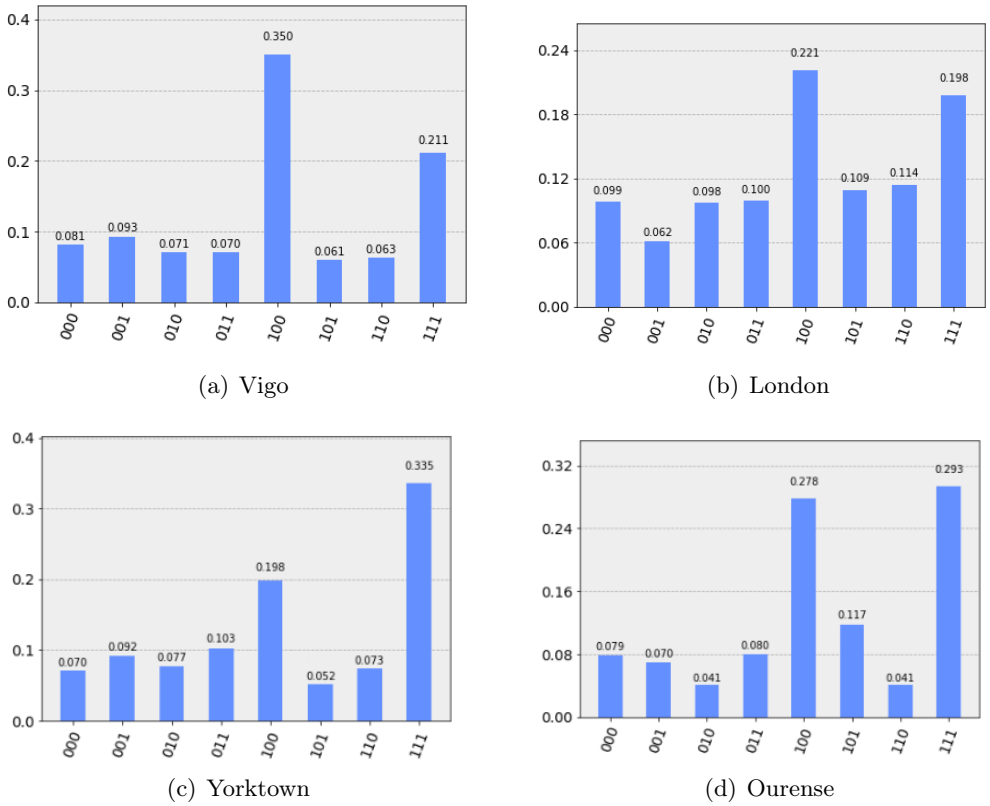
Figura 3.15: Simulações para diferentes computadores quânticos. Estados: $|00\rangle$ e $|10\rangle$

Grover 2 qbits, t = 1		
IBM Backend	Precisão média	Tempo de execução (s)
Vigo	90.97%	8.18
London	-%	-
Yorktown	86.55%	7.13
Ourense	92.61%	8.39

Tabela 3.9: Resultados de simulações em diferentes computadores quânticos com execuções = 1024.

Grover 2 qbits, t = 2		
Máquina real	Precisão média	Tempo de execução (s)
Vigo	97.25%	8.35
London	99.90%	8.77
Yorktown	89.71%	7.09
Ourense	92.23%	8.34

Tabela 3.10: Resultados de simulações em diferentes computadores quânticos com execuções = 1024.



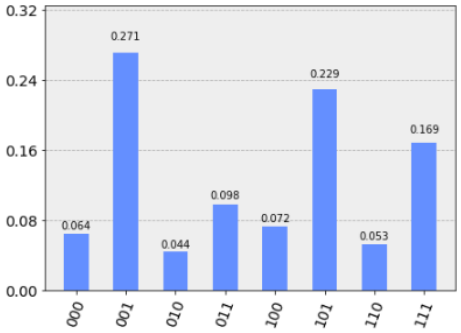
(a) Vigo (b) London

(c) Yorktown (d) Ourense

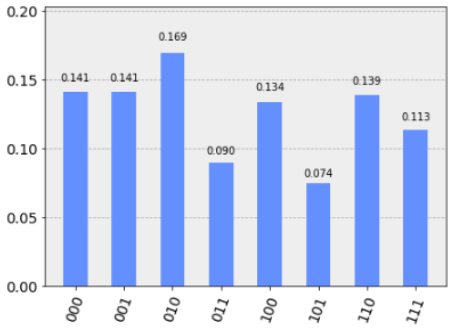
Figura 3.16: Simulações para diferentes computadores quânticos. Estados: $|100\rangle$ e $|111\rangle$

Grover 3 qbits, t = 2		
Máquina real	Precisão média	Tempo de execução (s)
Vigo	56.27%	8.37
London	41.98%	9.32
Yorktown	53.34%	7.38
Ourense	57.18%	8.56

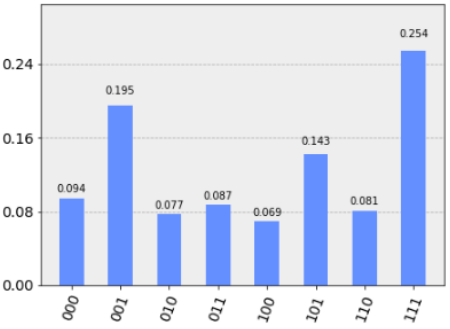
Tabela 3.11: Resultados de simulações em diferentes computadores quânticos com execuções = 1024.



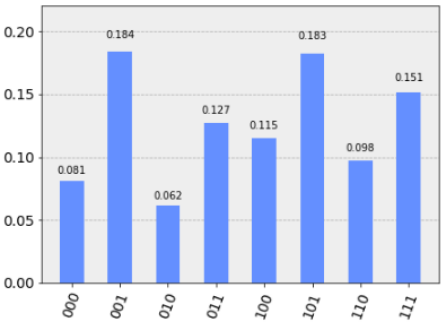
(a) Vigo



(b) London



(c) Yorktown



(d) Ourense

Figura 3.17: Simulações para diferentes computadores quânticos. Estados: $|001\rangle$, $|101\rangle$ e $|111\rangle$

Grover 3 qbits, t = 3		
Máquina real	Precisão média	Tempo de execução (s)
Vigo	66.97%	8.59
London	35.6%	10.15
Yorktown	59.22%	7.42
Ourense	51.83%	8.45

Tabela 3.12: Resultados de simulações em diferentes computadores quânticos com execuções = 1024.

As simulações do Grover de 4 qbits para $t = 1$, $t = 2$ e $t = 3$ se encontram nas tabelas 3.13, 3.14 e 3.15.

Grover 4 qbits, t = 1		
Máquina real	Precisão	Tempo de execução (s)
Vigo	7.5%	8.72
London	5.5%	10.37
Yorktown	9.1%	7.25
Ourense	8.7%	8.43

Tabela 3.13: Resultados de simulações em diferentes computadores quânticos com execuções = 1024.

Grover 4 qbits, t = 2		
Máquina real	Precisão	Tempo de execução (s)
Vigo	15.7%	8.70
London	13.8%	10.10
Yorktown	14.4%	7.89
Ourense	16.9%	8.78

Tabela 3.14: Resultados de simulações em diferentes computadores quânticos com execuções = 1024.

Grover 4 qbits, t = 3		
Máquina real	Precisão	Tempo de execução (s)
Vigo	17.2%	8.45
London	15.6%	10.08
Yorktown	18.3%	7.54
Ourense	19.1%	8.63

Tabela 3.15: Resultados de simulações em diferentes computadores quânticos com execuções = 1024.

3.4.4 BBHT

Nos experimentos anteriores, utilizamos o Grover em sua forma original para diferentes cenários onde t é conhecido, embora variável. Agora escolheremos os melhores resultados para simular o algoritmo BBHT, quando t é desconhecido e atende à $1 \leq t \leq \frac{3N}{4}$. O BBHT será testado em um Grover de 3 qbits, pois julgamos que 2 qbits é um espaço muito pequeno, e o de 4 qbits resulta em alto ruído em máquinas reais. Também selecionamos como *backend* para testes o Vigo por demonstrar melhores resultados principalmente pra um espaço $N = 8$. E por fim, variaremos t.

Aqui, usaremos $execues = 1$ em vez de 1024 e rodaremos o algoritmo inteiro 50 vezes para cada cenário de t possível, a fim de calcular a média de loops principais e rotações de Grover realizadas. Queremos medir a saída do registrador imediatamente após k iterações de Grover e verificar se aquele estado é um dos buscados. O pseudocódigo do algoritmo BBHT se encontra na seção 2.3.2 e sua implementação em Python pode ser acessada no apêndice. Omitiremos os histogramas das simulações e resumiremos os resultados nas tabelas abaixo. O *loop* principal irá parar quando ao menos uma solução for encontrada após k aplicações de Grover.

BBHT Grover 3 qbits, QisKit			
Soluções (t)	Média de loop principal	Média rotações Grover	Tempo médio de execução (s)
1	1	1	0.006
2	1	1	0.008
3	1.5	1.3	0.026

Tabela 3.16: BBHT Grover 3 qbits, QisKit

BBHT Grover 3 qbits, IBM			
Soluções (t)	Média de loop principal	Média rotações Grover	Tempo médio de execução (s)
1	1.2	1.1	3.21
2	1	1	3.44
3	2.3	1.7	8.78

Tabela 3.17: BBHT Grover 3 qbits, IBM

Capítulo 4

Resultados

Neste capítulo os resultados das simulações realizadas serão discutidos e comparados. Serão feitas também observações de desempenho esperado e comportamentos atípicos, visto o cenário analisado.

4.1 Análises e comparações

Teoricamente, à medida que aumentamos o espaço de busca N, a probabilidade de encontrarmos uma única solução aumenta. Isso é válido nos cálculos teóricos e comprovado nas simulações do QisKit, com resultados de 94% a 98%. No entanto, ao realizar a mesma busca em uma máquina real, nota-se a presença de ruído significativo. Tal ruído cresce conforme o número de qbits necessários devido a um fenômeno chamado de decoerência. Decoerência é a perda de propriedades da mecânica quântica causada pelas interações com o ambiente, e esse efeito aumento à medida que o sistema se torna mais complexo.

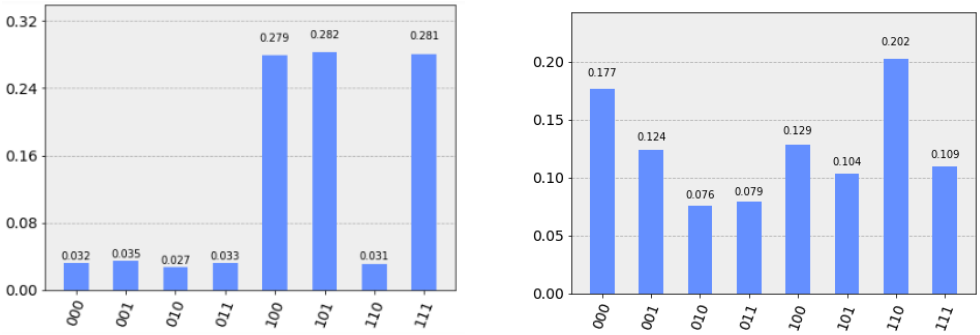
Por consequência da presença de ruído, ou seja, do aumento das amplitudes de itens não marcados, a probabilidade de achar os estados procurados decresce. Para um Grover de 2 qbits, os resultados em todos *backends* testados ainda são considerados muito bons, tanto para $t = 1$ e $t = 2$.

Visão geral Grover 2 qbits		
Soluções (t)	Precisão QisKit	Precisão IBM
1	100%	90.58%
2	100%	94.75%

Tabela 4.1: Visão geral simulações Grover 2 qbits.

É interessante notar que para o caso de $t = 2$, a precisão média nos dispositivos da IBM chegou a superar o caso de $t = 1$. Com o aumento de t, há menos ruído distribuído para os itens não marcados. Este comportamento será repetido nas análises seguintes, visto que a soma de todas probabilidades de estados deverá ser sempre 100%. Porém é importante relembrar que conforme se aumenta o número de qbits, aumenta-se o ruído, ocasionando em alguns casos resultados não satisfatórios.

Ao analisar o Grover de 3 qbits para múltiplas soluções, constata-se o aumento significativo de ruído, impactando a precisão de ambas simulações propostas. Averiguando seus resultados em uma máquina real, essa perturbação ruidosa se torna ainda mais evidente, e em alguns casos para estados e backends específicos, o ruído é tão alto que a probabilidade de encontrar um estado não marcado supera a de item que é buscado.



(a) QisKit (b) ibmq_london

Figura 4.1: Efeitos da presença de alto ruído

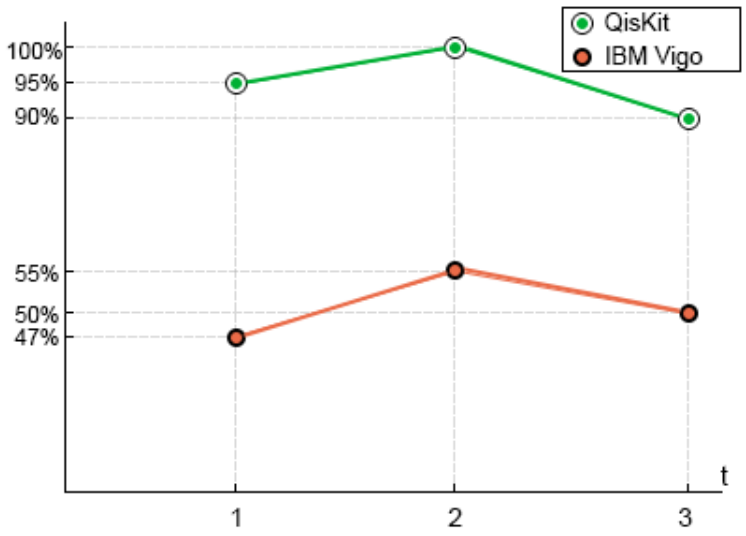


Figura 4.2: Número de soluções x Precisão para Grover 3 qbits

O *ibmq_london* no geral não obteve bons resultados para um Grover de 3 qbits e múltiplas soluções, na maioria das vezes permitindo que estados não marcados obtivessem probabilidade próxima dos marcados. Ainda assim, o comportamento da figura 4.1 não foi visto com frequência, e é tido como atípico.

Visão geral Grover 3 qbits		
Soluções (t)	Precisão QisKit	Precisão IBM
1	95%	47.51%
2	100%	52.59%
3	85%	55.43%

Tabela 4.2: Visão geral simulações Grover 3 qbits.

No Grover de 4 qbits as simulações no QisKit são boas para única e múltiplas soluções. Porém, ao simular os cenários nos dispositivos da IBM, observa-se o alto índice de ruído, tornando uma tarefa impossível saber quais estados foram marcados. Os resultados não são bons para nenhum backend testado, mesmo quando $t = 1$. Foram tentadas diversas formas para melhorar os simulações como aumento do número de portas, reescrita da função oráculo - visto que dispositivos utilizados não suportam portas CCCZ, fazendo uma decomposição em operações mais simples - e por fim, modificação do parâmetro de otimi-

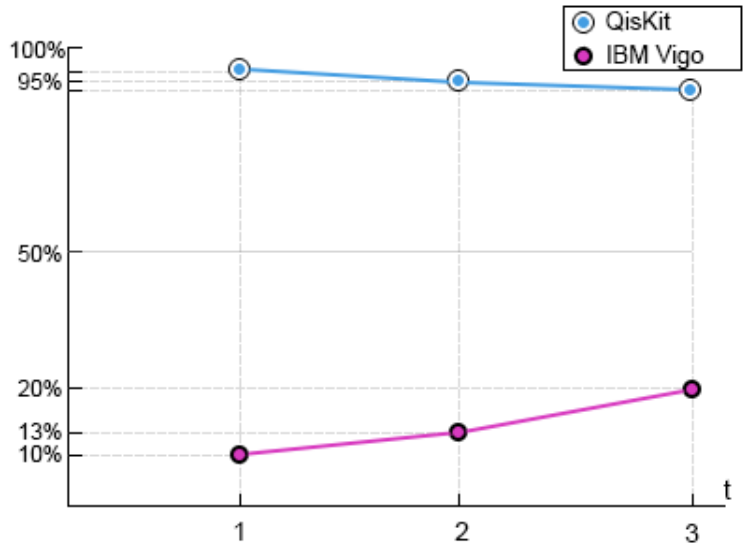


Figura 4.3: Número de soluções x Precisão para Grover 4 qbits

zação ao seu máximo. Nenhuma delas ocasionou efeito observável. Vale ressaltar porém que os chips quânticos atualmente são muito ruidosos e o circuito de 4 qbits é relativamente complexo e profundo.

Visão geral Grover 4 qbits		
Soluções (t)	Precisão QisKit	Precisão IBM
1	96%	9.26%
2	94%	14.06%
3	95%	19.12%

Tabela 4.3: Visão geral simulações Grover 4 qbits.

Fazendo uma averiguação dos backends usados nas simulações, o *Vigo* demonstrou um melhor desempenho na maioria dos cenários testados em termos de precisão quanto de tempo de execução. Observando a tabela 3.8, esse dispositivo possui a menor variação de erro em portas CNOT. O *Yorktown* demonstrou resultados satisfatórios e sempre obteve o menor tempo de execução em todas análise, nunca ultrapassando 8s. Ele é o único a ter uma distribuição de qbits diferente dos demais dispositivos testados. Porém seu desempenho em um Grover de 2 qbits para $1 \leq t \leq 3$, apesar de ainda ser bom, não ultrapassou os 90%. Já o *Ourense* teve desempenho similar ao *Vigo*, em alguns casos, ultrapassando a precisão deste. Seu destaque foi na simulação do Grover de 4 qbits de múltiplas soluções, onde, ainda que com o alto ruído, obteve precisão melhor que os demais backends usados. O *Ourense* possui a menor faixa de erros de porta U_3.

Múltiplas soluções demonstraram uma melhora na precisão dos estados marcados, mais por uma questão probabilística do que por acurácia das máquinas reais. Essa afirmação pode ser comprovada ainda que observando os resultados para um Grover de 4 qbits, o único cujas simulações reais não chegaram minimamente perto dos resultados do QisKit.

Finalmente, ao analisar o cenário onde t é desconhecido, é possível que se realize mais iterações de Grover que o número ótimo, mas é válido lembrar que estamos lidando com um algoritmo probabilístico. O algoritmo foi rodado em média 50 vezes para cada cenário de t possível. É destacado quando $t = 3$, onde em média é preciso duas iterações do loop principal, ou seja, duas tentativas para encontrar um dos valores de t e a possibilidade de duas iterações de Grover, quando seu valor teórico e ótimo é 1. Ressalta-se também a

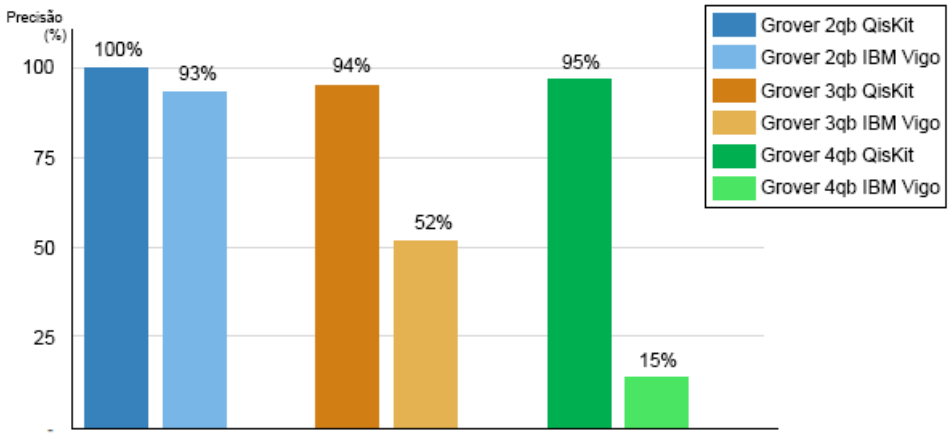


Figura 4.4: Precisões médias para QisKit x IBM Vigo

aproximação de análise diferente para o BBHT, onde é utilizado $execues = 1$ e o Grover puro, que utilizamos $execues = 1024$, já que estamos querendo medir a saída de um registrador.

Capítulo 5

Conclusão

5.1 Considerações finais

O algoritmo de Grover entrega um excelente resultado teórico para casos de solução unitária e múltiplas, como demonstrado por cálculos e pela simulação no QisKit. Porém, simulações em máquinas quânticas reais se mostraram próximas dos valores esperados apenas quando o espaço de busca é pequeno (N = 4), algo longe de um cenário real e que não possui uma aplicação prática. Ao aumentarmos para 4 qbits, o ruído se torna tão alto que impossibilita visualizar o resultado esperado, seja para $t = 1$ ou $t > 1$. Embora o algoritmo seja ótimo para solução única, sua confiabilidade pode decair para múltiplas soluções ainda mais quando simulado em dispositivo quântico. Com isso, o algoritmo de Grover poder não ser adequado para soluções reais no atual momento, mais por uma limitação de hardware.

A partir da análise dos experimentos realizados, nota-se um decaimento drástico de precisão quando $t = 1$ para um Grover de 2 qbits, para um de 3 qbits e por fim, um de 4 qbits. Os resultados para 2 qbits são próximos do esperado, para 3 qbits há um decaimento de cerca de 50%, algo bastante notório. Já para 4 qbits os resultados se tornam impráticos, sendo impossível discernir qual o estado marcado. Quando aumentamos t, notamos uma melhoria de precisão, mas isso é explicado por uma questão meramente probabilística, e não da eficiência do chip quântico. Os resultados do Grover de 4 qbits para solução unitária ou múltipla foram, de certa forma, surpreendentes, dada a precisão teórica na faixa de 97% (*t = 2,3*) a 98% (*t = 1*). Variando uma precisão na faixa dos 10% para *t = 1*, significa um decaimento de quase 90% em relação ao valor esperado. Em contrapartida, um computador clássico realizaria uma busca linear de um item em um espaço de $N = 16$ em até $\frac{N}{2}$ etapas com uma precisão de 100%. Isso só demonstra que ainda há uma perceptível limitação de hardware em computadores quânticos, o que por consequência afeta sua precisão e eficiência.

Com os resultados observados neste trabalho, é concluído que ainda há uma notável limitação de hardware o que ocasiona a insuficiência de precisão dos resultados simulados em máquinas quânticas reais. A complexidade de implementação de circuitos mais profundos, como por exemplo um Grover de 4 qbits, gera um aumento de ruído, tempo de execução e erros de portas. Todos esses fatores afetam diretamente a precisão e desempenho dos algoritmos desenvolvidos. Como as simulações do QisKit trabalham em um cenário "perfeito", tais erros de cenário real não são simulados nos resultados, e temos a ilusória que os resultados reais serão similares, mas isso só foi observado para algoritmos mais simples, como o Grover de 2 qbits para $t = 1$ e $t = 2$.

5.2 Pesquisas futuras

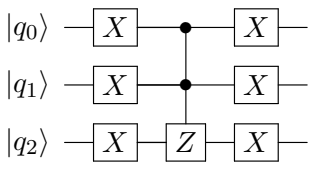
Para trabalhos futuros, simular em máquinas reais o desempenho da *contagem quântica* um algoritmo capaz de estimar o número desconhecido de soluções t antes de executar o algoritmo de Grover. Também seria válido simulações de algoritmos que utilizam o Grover como sub rotina para outros problemas, além de apenas busca, como por exemplo, algoritmos para encontrar o mínimo e máximo locais, e também a implementação de busca adaptativa pura.

Vale salientar a importância de acompanhar os avanços dos dispositivos quânticos. Por se tratar de uma área relativamente recente e de potencial para novas descobertas, é válido se atualizar quanto às frequentes melhorias e atualizações dos chips quânticos atuais, além da possibilidade de lançamento de novos computadores quânticos que podem possuir um desempenho melhor ou próximo ao esperado teórico. Será importante no futuro reavaliar este e outros trabalhos com a tecnologia mais recente e comparar os avanços alcançados.

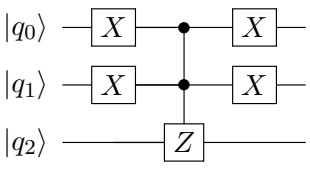
Apêndice A

Oráculos

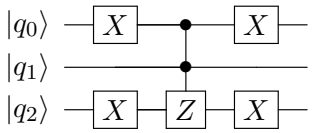
A.1 3 Qbit Grover



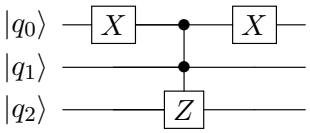
000



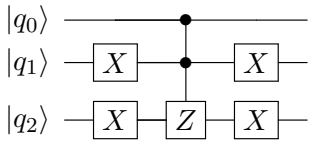
001



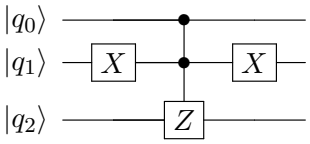
010



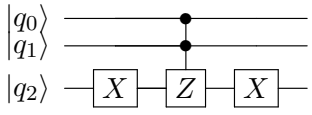
011



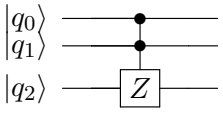
100



101

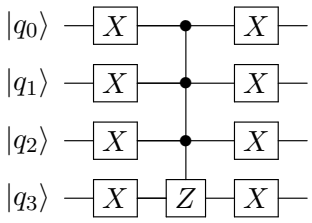


110

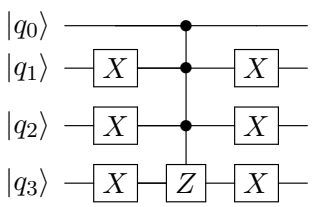


111

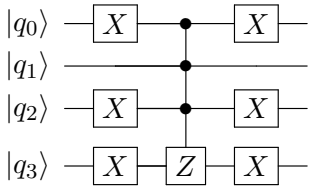
A.2 4 Qbit Grover



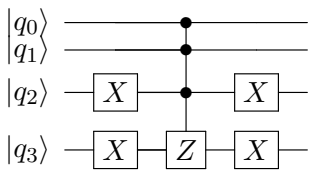
0000



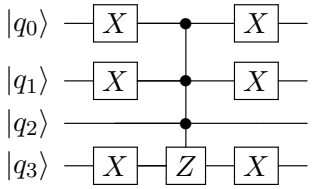
0001



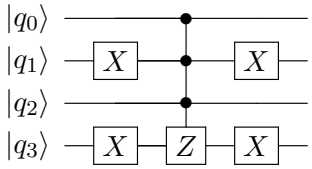
0010



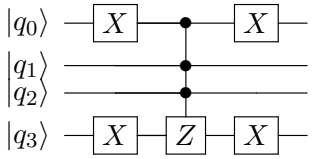
0011



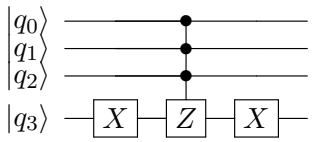
0100



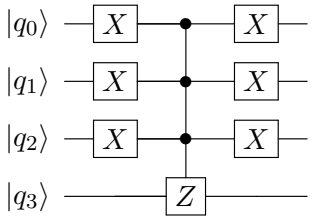
0101



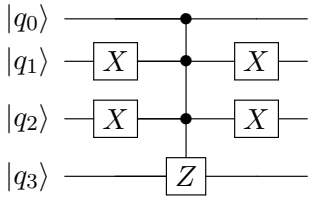
0110



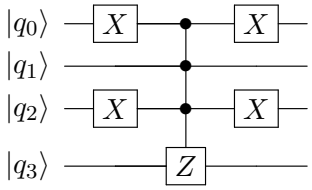
0111



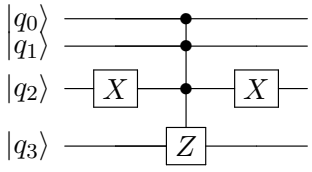
1000



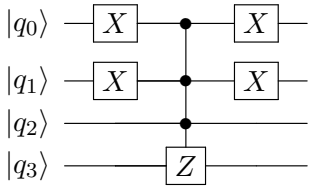
1001



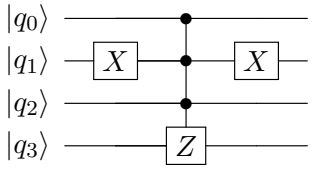
1010



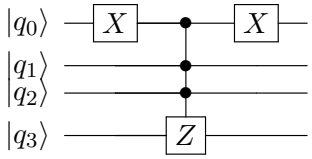
1011



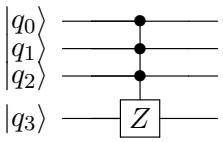
1100



1101

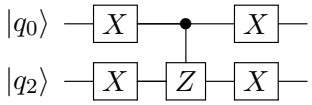


1110

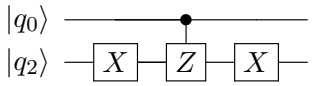


1111

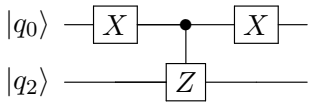
A.3 2 Qbit Grover



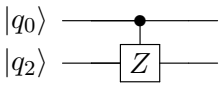
00



01



10



11

Apêndice B

Algoritmo de Grover

Também disponível online neste link.

```
from qiskit import IBMQ, BasicAer
from qiskit import QuantumCircuit, ClassicalRegister, QuantumRegister
import numpy as np
import math as m

def oraculo(qc, qr):

    #Defina seu oracle#
    #Verifique appendix#


def nCZ(circuit, controls, target):
    if (len(controls) > 4):
        raise ValueError('Nao implementado')
    elif (len(controls) == 1):
        circuit.h(target)
        circuit.cx(controls[0], target)
        circuit.h(target)
    elif (len(controls) == 2):
        circuit.h(target)
        circuit.ccx(controls[0], controls[1], target)
        circuit.h(target)
    elif (len(controls) == 3):
        circuit.h(target)
        circuit.cccx(controls[0], controls[1], controls[2], target)
        circuit.h(target)


def inversao(qc, qr, n):
    qc.h(qr)
    qc.x(qr)

    nCZ(qc, [qr[j] for j in range(n-1)], qr[n-1])
```

```
    qc.x(qr)
    qc.h(qr)

n = #VALOR#
N = 2**n
M = #VALOR#

qr = QuantumRegister(n)
cr = ClassicalRegister(n)

groverCircuit = QuantumCircuit(qr,cr)
groverCircuit.h(qr)

iteracoes = m.floor(m.pi *(N/M)**(1/2) / 4 )

for i in range(iteracoes):
    oraculo(groverCircuit, qr)
    groverCircuit.barrier()
    inversao(groverCircuit, qr, n)
    groverCircuit.barrier()

print(iteracoes)

groverCircuit.measure(qr,cr)
```

Apêndice C

Algoritmo BBHT

Também disponível online neste link.

```
from qiskit import QuantumCircuit, ClassicalRegister, QuantumRegister

def oraculo(qc, qr):


def nCZ(circuit, controls, target):
    if (len(controls) > 4):
        raise ValueError('Nao implementado')
    elif (len(controls) == 1):
        circuit.h(target)
        circuit.cx(controls[0], target)
        circuit.h(target)
    elif (len(controls) == 2):
        circuit.h(target)
        circuit.ccx(controls[0], controls[1], target)
        circuit.h(target)
    elif (len(controls) == 3):
        circuit.h(target)
        circuit.cccx(controls[0], controls[1], controls[2], target)
        circuit.h(target)


def inversao(qc, qr, n):
    qc.h(qr)
    qc.x(qr)

    nCZ(qc, [qr[j] for j in range(n-1)], qr[n-1])

    qc.x(qr)
    qc.h(qr)

def groverIteracao(groverCircuit, qr, n):
    oraculo(groverCircuit, qr)
    inversao(groverCircuit, qr, n)
```

```
pi = math.pi
n = 3
N = 2**n
m = 1
lmb = 1.34

qr = QuantumRegister(n, 'q')
cr = ClassicalRegister(n, 'c')

groverCircuit = QuantumCircuit(qr,cr)
groverCircuit.h(qr)

contador = 1
achou = 0

while (achou == 0):
    print("====== iteracao : ", contador, "======")
    kLista = list(range(1, math.floor(m+1)))
    print("lista de k : ", kLista)
    kAleatorio = random.choice(kLista)
    print("iteracoes de grover : ", kAleatorio)

    for i in range(kAleatorio):
        groverIteracao(groverCircuit, qr, n)

    groverCircuit.measure(qr,cr)
    result = execute(groverCircuit, backend, shots = 1,  optimization_l
    counts = result.get_counts()

    for j in counts:
        if (j in ['111']):
            print(j, " - ok")
            achou = 1
        else:
            print(j, " - nope")
            m = (m*lmb)
            print(m)

    print(counts)
    contador = contador + 1
```

Apêndice D

Cálculos teóricos Grover

A fórmula utilizada para o cálculo de sucesso de observação foi proposta por Bassard, Boyer e Hoyer em [9] e é definida por:

$$p = \sin^2\left((2 * r + 1)\arcsin\sqrt{\frac{t}{N}}\right) \tag{D.1}$$

Sendo r o número de rotações definido por:

$$r = \frac{\pi}{4}\sqrt{\frac{N}{t}} \tag{D.2}$$

,

onde N é o tamanho do espaço de busca e t a quantidade de soluções procuradas.

2 Qbits, t = 1

$r = \frac{\pi}{4}\sqrt{\frac{4}{1}} = 1.57 \approx 1$

$\sin^2\left((2 * 1 + 1)\sin^{-1}\sqrt{\frac{1}{4}}\right) = 1$

2 Qbits, t = 2

$r = \frac{\pi}{4}\sqrt{\frac{4}{1}} = 1.11 \approx 1$

$\sin^2\left((2 * 1 + 1)\sin^{-1}\sqrt{\frac{2}{4}}\right) = 1$

3 Qbits, t = 1

$r = \frac{\pi}{4}\sqrt{\frac{8}{1}} = 2.22 \approx 2$

$\sin^2\left((2 * 2 + 1)\sin^{-1}\sqrt{\frac{1}{8}}\right) = 0.94$

3 Qbits, t = 2

$r = \frac{\pi}{4}\sqrt{\frac{8}{2}} = 1.57 \approx 1$

$\sin^2\left((2 * 1 + 1)\sin^{-1}\sqrt{\frac{2}{8}}\right) = 1$

3 Qbits, t = 3

$r = \frac{\pi}{4}\sqrt{\frac{8}{3}} = 1.28 \approx 1$

$\sin^2\left((2*1+1)\sin^{-1}\sqrt{\frac{3}{8}}\right) = 0.91$

4 Qbits, t = 1

$r = \frac{\pi}{4}\sqrt{\frac{16}{1}} = 3.14 \approx 3$

$\sin^2\left((2*3+1)\sin^{-1}\sqrt{\frac{1}{16}}\right) = 0.98$

4 Qbits, t = 2

$r = \frac{\pi}{4}\sqrt{\frac{16}{2}} = 2.22 \approx 2$

$\sin^2\left((2*2+1)\sin^{-1}\sqrt{\frac{2}{16}}\right) = 0.97$

4 Qbits, t = 3

$r = \frac{\pi}{4}\sqrt{\frac{16}{3}} = 1.81 \approx 1$

$\sin^2\left((2*1+1)\sin^{-1}\sqrt{\frac{3}{16}}\right) = 0.97$

Bibliografia

[1] R.P. Feynman. *Simulating physics with computers*. International Journal of Theoretical Physics, 1982.

[2] Paul Benioff. Quantum mechanical models of turing machines that dissipate no energy. *Physical Review Letters*, 48:1581–1585, 1982.

[3] David Deutsch. Quantum theory, the church–turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400:117 – 97, 1985.

[4] D. Deutsch and R. Jozsa. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 439:553 – 558, 1992.

[5] P. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[6] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[7] Lov K. Grover. Quantum mechanics helps in searching for a needle in a haystack. *Physical Review Letters*, 79(2):325–328, Jul 1997.

[8] Christoph Durr and Peter Hoyer. A quantum algorithm for finding the minimum, 1996.

[9] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik*, 46(4-5):493–505, Jun 1998.

[10] William Baritompa, D. Bulger, and Graham Wood. Grover's quantum algorithm applied to global optimization. *SIAM Journal on Optimization*, 15:1170–1184, 01 2005.

[11] Yipeng Liu and Gary Koehler. Using modifications to grover's search algorithm for quantum global optimization. *European Journal of Operational Research*, 207:620–632, 12 2010.

[12] Pedro Lara, Renato Portugal, and Carlile Lavor. A new hybrid classical-quantum algorithm for continuous global optimization problems, 2013.

[13] Peter W. Shor. Why haven't more quantum algorithms been found? *J. ACM*, 50(1):87–90, January 2003.

[14] E. Z. Schrödinger. Über eine bemerkenswerte eigenschaft der quantenbahnen eines einzelnen elektrons. *Zeitschrift für Physik*, 1923.

[15] W. Z. Heisenberg. Über quantentheoretische umdeutung kinematischer und mechanischer beziehungen. *Zeitschrift für Physik*, 1925.

[16] W. a Heisenberg. Uber den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik. *Z. Phys.*, 43:172–198, 1927.

[17] J. Matson. What is quantum mechanics good for? *Scientific American*, 2010.

[18] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, New York, NY, USA, 10th edition, 2011.

[19] G.E. Moore. Cramming more components onto integrated circuits. *Proceedings of the IEEE*, 86:82–85, 02 1998.

[20] Michael Sipser. *Introduction to the Theory of Computation*. Course Technology, second edition, 2006.

[21] Goong Chen, Stephen A. Fulling, and Jeesen Chen. Generalization of grover's algorithm to multiobject search in quantum computing, part i: Continuous time and discrete time, 2000.

[22] G. Chen and Sun S. Generalization of grover's algorithm to multiobject search in quantum computing, part ii: General unitary transformations, 2000.

[23] G. Chen, S. A. Fulling, and M. O. Scully. Grover's algorithm for multiobject search in quantum computing, 1999.

[24] Gilles Brassard, Peter HØyer, and Alain Tapp. Quantum counting. *Lecture Notes in Computer Science*, page 820–831, 1998.

[25] Michele Mosca. Quantum searching counting and amplitude ampli cation by eigenvector analysis. 1998.

[26] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation, 2000.

[27] Nitin Patel, Zelda Zabinsky, and Robert Smith. Pure adaptive search in monte carlo optimization. *Math Program*, 43, 01 1988.

[28] D. Bulger, W. Baritompa, and Graham Wood. Implementing pure adaptive search with grover's quantum algorithm. *Journal of Optimization Theory and Applications*, 116:517–529, 03 2003.

[29] Yanhu Chen, Shijie Wei, Xiong Gao, Cen Wang, Jian Wu, and Hongxiang Guo. An optimized quantum maximum or minimum searching algorithm and its circuits, 2019.

[30] F. Toyama, Wytse van Dijk, and Yuki Nogami. Quantum search with certainty based on modified grover algorithms: Optimum choice of parameters. *Quantum Information Processing*, 12, 05 2013.

[31] G. L. Long. Grover algorithm with zero theoretical failure rate. *Physical Review A*, 64(2), Jul 2001.

[32] Anaconda software distribution. https://anaconda.com/.

[33] Project jupyter. https://jupyter.org/.

[34] IBM Quantum devices. https://quantum-computing.ibm.com/.