

BRUNO DEL BIANCO SOARES

COMPUTAÇÃO QUÂNTICA APLICADA À SOLUÇÃO DE  
PROBLEMAS DE BUSCA E CLASSIFICAÇÃO

São Paulo

2023

BRUNO DEL BIANCO SOARES

COMPUTAÇÃO QUÂNTICA APLICADA À SOLUÇÃO DE  
PROBLEMAS DE BUSCA E CLASSIFICAÇÃO

Trabalho de formatura apresentado à Escola Politécnica  
da Universidade de São Paulo para obtenção do título  
de Engenheiro Eletricista com ênfase em Computação

São Paulo

2023

BRUNO DEL BIANCO SOARES

COMPUTAÇÃO QUÂNTICA APLICADA À SOLUÇÃO DE  
PROBLEMAS DE BUSCA E CLASSIFICAÇÃO

Trabalho de formatura apresentado à Escola Politécnica  
da Universidade de São Paulo para obtenção do título  
de Engenheiro Eletricista com ênfase em Computação

Área de Concentração:  
Engenharia de Computação

Orientadora: Regina Melo Silveira

São Paulo  
2023

## **AGRADECIMENTOS**

Agradeço aos meus pais por batalharem desde meu nascimento para que tivesse acesso à educação e por todo carinho, paciência e amor.

Agradeço a todo corpo docente da Escola Politécnica da Universidade de São Paulo e, em especial, a professora Regina Melo Silveira.

## Resumo

O presente trabalho tem por objetivo estudar os fundamentos da computação quântica, estudar alguns algoritmos que são promissores candidatos a uma suposta vantagem da computação quântica sobre a computação clássica, aplicar esses algoritmos a resolução de problemas reais e fazer um estudo de seu desempenho, complexidade e aplicabilidade para problemas maiores e mais complexos. Os dois algoritmos escolhidos foram o Algoritmo de Grover, usado para solucionar problemas de busca não estruturada, e o algoritmo de Máquina de Vetor de Suporte Quântica, que é um algoritmo de aprendizado de máquina usado para solucionar problemas de classificação.

Palavras-chave: Computação Quântica. Algoritmo de Grover. Busca não estruturada. Máquina de Vetor de Suporte Quântica.

## **Abstract**

This project aims to study the fundamentals of Quantum Computing, study some quantum algorithms that are promising candidates to have an advantage over classical algorithms, apply those algorithms to solve real world problems and study their performance, complexity and applicability to solve more complex problems. The two chosen algorithms are the Grover Algorithm, which is a algorithm used to solve unstructured search problems, and the Quantum Support Vector Machine algorithm, which is a machine learning algorithm used solve classification problems.

Keywords: Quantum Computing, Grover's Algorithm, Unstructured Search. Quantum Support Vector Machine.

## Lista de Figuras

1	Esfera de Bloch. . . . .	27
2	Computadores Quânticos e Simuladores de Computadores Quânticos da IBM disponíveis . . . . .	36
3	Exemplo de Circuito Quântico criado com a biblioteca Qiskit, usando o Google Colaboratory . . . . .	37
4	Busca não ordenada . . . . .	38
5	Inversão de fase da amplitude associada a solução . . . . .	40
6	Inversão de todas as amplitudes ao redor da média das amplitudes . . . . .	40
7	Algoritmo de Grover - representação vetorial . . . . .	42
8	Algoritmo de Grover - representação vetorial (Inversão de fase da amplitude associada a solução) . . . . .	42
9	Algoritmo de Grover - representação vetorial (Inversão de todas as amplitudes ao redor da média das amplitudes) . . . . .	43
10	Circuito geral do Algoritmo de Grover . . . . .	43
11	Circuito do Algoritmo Grover em Qiskit, para encontrar o $ 11\rangle$ . . . . .	45
12	Máquina de vetor de Suporte . . . . .	47
13	Mapa de Características . . . . .	48

14	Problema da coloração de mapas expresso em um grafo . . . . .	51
15	Grafo com 3 vértices . . . . .	52
16	Círculo em Qiskit do Oráculo . . . . .	53
17	Código Qiskit do oráculo . . . . .	53
18	Círculo em Qiskit do Difusor de Grover . . . . .	54
19	Amplitudes . . . . .	54
20	Código em Qiskit da criação do mapa de características Pauli X, treinamento do modelo e obtenção da precisão para o conjunto de teste . . . . .	57
21	Exemplo de imagens de biópsia por aspiração agulha fina usadas na construção do conjunto de dados. Imagem retirada de [13] . . . .	58
22	Diagrama de classificação - QSVM com conjunto de dados Scikit- Learn de reconhecimento de câncer de mama reduzido com PCA, com mapa de características Pauli X . . . . .	59
23	Diagrama de classificação - QSVM com conjunto de dados Scikit- Learn de reconhecimento de tipos de vinho reduzido com PCA, com mapa de características Pauli X . . . . .	60
24	Três tipos de Íris: Setosa, Versicolor e Virginica . . . . .	61



25	Diagrama de classificação - QSVM com conjunto de dados Scikit-Learn de reconhecimento de flores íris reduzido com PCA, com mapa de características Pauli Z. . . . .	61
----	---	----

## Lista de Tabelas

- 1 Tabela com comparação da precisão das previsões dos algoritmos SVM e QSVM para diferentes conjuntos de dados e diferentes mapas de características . . . . . 63
- 2 Tabela com comparação da precisão das previsões dos algoritmos SVM e QSVM Pauli X, variando o número de características . . . . 63

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>15</b>
1.1	HISTÓRIA DA COMPUTAÇÃO QUÂNTICA . . . . .	15
1.2	OBJETIVOS E METODOLOGIA . . . . .	16
<b>2</b>	<b>BASES TEÓRICAS DE MECÂNICA QUÂNTICA</b>	<b>18</b>
2.1	NOTAÇÃO DE DIRAC . . . . .	18
2.2	POSTULADOS DA MECÂNICA QUÂNTICA . . . . .	20
2.2.1	POSTULADO I - DESCRIÇÃO DOS ESTADOS QUÂNTICOS	20
2.2.2	POSTULADO II - DESCRIÇÃO QUÂNTICA DE OBSERVÁ- VEIS . . . . .	21
2.2.3	POSTULADO III - MEDIDAS SOBRE ESTADOS QUÂNTICOS	21
2.2.4	POSTULADO IV - PROBABILIDADE DE MEDIDA . . . . .	21
2.2.5	POSTULADO V - EVOLUÇÃO COM O TEMPO . . . . .	22
2.2.6	POSTULADO VI - SISTEMAS COMPOSTOS . . . . .	22
2.3	SUPERPOSIÇÃO . . . . .	23

2.4	<b>REGRA DE BORN</b>	23
2.5	<b>EMARANHAMENTO</b>	23
<b>3</b>	<b>COMPUTAÇÃO QUÂNTICA</b>	<b>24</b>
3.1	QUBIT	24
3.2	ESFERA DE BLOCH	26
3.3	OPERADORES QUÂNTICOS	27
3.3.1	OPERADORES UNÁRIOS	28
3.3.1.1	OPERADOR X (NOT)	28
3.3.1.2	OPERADOR Y	29
3.3.1.3	OPERADOR Z	29
3.3.1.4	OPERADOR $RZ_{\varphi}$	30
3.3.1.5	OPERADOR DE HADAMARD	31
3.3.2	OPERADORES BINÁRIOS	32
3.3.2.1	OPERADOR SWAP	33
3.3.2.2	OPERADOR CNOT	34
3.3.2.3	OPERADOR CZ	35

3.4	<b>AMBIENTE DE PROGRAMAÇÃO</b>	36
<b>4</b>	<b>ALGORITMO DE GROVER</b>	<b>38</b>
4.1	FUNCIONAMENTO DO ALGORITMO DE GROVER	39
<b>5</b>	<b>MÁQUINA DE VETOR DE SUPORTE QUÂNTICA</b>	<b>46</b>
5.1	IMPLEMENTAÇÃO DA MÁQUINA DE VETOR DE SUPORTE (SVM)	46
5.2	IMPLEMENTAÇÃO QUÂNTICA DA MÁQUINA DE VETOR DE SUPORTE (QSVM)	49
<b>6</b>	<b>ESTUDOS DE CASO</b>	<b>51</b>
6.1	COLORAÇÃO DE GRAFOS (GROVER)	51
6.2	CLASSIFICAÇÃO (QSVM)	56
6.2.1	CONJUNTOS DE DADOS UTILIZADOS	57
6.2.1.1	CONJUNTO DE DADOS 1 - CÂNCER DE MAMA	57
6.2.1.2	CONJUNTO DE DADOS 2 - VINHOS	59
6.2.1.3	CONJUNTO DE DADOS 3 - ÍRIS	60
6.2.2	COMPARAÇÃO DE RESULTADOS	62
<b>7</b>	<b>CONCLUSÃO</b>	<b>64</b>



# 1 INTRODUÇÃO

## 1.1 HISTÓRIA DA COMPUTAÇÃO QUÂNTICA

A computação quântica tem seu início na década de 1980 com o físico teórico norte-americano Richard Feynman. Feynman notou a grande dificuldade que era simular um sistema de partículas quânticas em um computador clássico, e então propôs um novo conceito, e uma mudança no paradigma da computação. Feynman propôs utilizar as características das partículas quânticas como superposição e entrelaçamento quântico para fazer computação.

Essa teoria de computação quântica teve poucos avanços durante anos, até que em 1994, recebeu um novo capítulo com a criação do algoritmo de Shor, por Peter Shor. Esse é um algoritmo quântico para a fatoração de números inteiros. Problema que é reconhecidamente complexo para computadores clássicos. Dado um número inteiro  $n$  com pelo menos dois divisores primos distintos, o algoritmo de Shor consegue calcular um divisor não trivial de  $n$  de forma relativamente simples, algo que não é possível de ser feito em um computador clássico, tanto que essa dificuldade é uma das bases de um dos principais algoritmos criptográficos utilizados atualmente, o RSA.

Em 1996 o cientista Lov Grover publica um artigo descrevendo seu algoritmo, demonstrando que a computação quântica poderia ser usada para reduzir a complexidade computacional de uma busca em um banco de dados não ordenado. A complexidade computacional de seu algoritmo é  $O(\sqrt{N})$  passos quânticos, em oposição a uma complexidade computacional de  $O(N)$  passos clássicos,

executando a busca por força bruta em um computador clássico, ou seja, um ganho quadrático na complexidade.

Desde então, foram desenvolvidos algoritmos quânticos cada vez mais eficientes e mais complexos para solucionar diversas tarefas que são consideradas difíceis de serem executadas em computadores clássicos, como por exemplo: simulação de sistemas físico-químicos, ciência dos materiais, solução de sistemas de equações lineares e inteligência artificial.

## 1.2 OBJETIVOS E METODOLOGIA

Neste trabalho, será realizado um estudo das bases da mecânica quântica através de livros e artigos, com o intuito de compreender os princípios dessa área, como operadores hermitianos, estados quânticos, princípio de superposição, princípio de emaranhamento ou emaranhamento quântico, etc. Em seguida, será realizado um estudo sobre como a mecânica quântica é utilizada na computação quântica, explanando sobre os bits quânticos, as portas lógicas quânticas, e como podemos montar circuitos quânticos para realizar determinado algoritmo. Conhecendo as bases tanto da mecânica quântica como da computação quântica, será feito um estudo do Algoritmo de Grover e Algoritmo de Máquina de Vetro de Suporte Quântica.

Serão, então, estudados problemas de busca em dados não estruturados, como por exemplo, o problema de coloração de grafos. A partir disso, serão implementado um circuito quântico, utilizando o Algoritmo de Grover para solucionar esse problema. Esses circuitos serão executados em simuladores de com-



putadores quânticos, através da plataforma IBM Quantum, e usando a biblioteca, de Python, Qiskit.

Além disso, faremos uma análise do algoritmo Máquina de Vetor de Suporte Quântica. Usaremos três conjuntos de dados diferentes, e iremos classificar esses conjuntos de dados variando alguns parâmetros, e comparando a precisão obtida.

## 2 BASES TEÓRICAS DE MECÂNICA QUÂNTICA

A Mecânica Quântica é a área da física responsável por estudar o comportamento de objetos em escala atômica e sub-atômica. Sua história nos remete ao século 19, quando o problema da radiação de corpo negro intrigava a comunidade científica. Um corpo negro seria um objeto que absorveria toda radiação eletromagnética incidente sobre ele, e não refletiria radiação alguma. O corpo negro é um objeto ideal e hipotético, que não existe na natureza, mas nos auxilia a compreender o comportamento de objetos reais, com estrelas. O corpo negro emitiria radiação na mesma taxa que absorveira radiação, e essa taxa de emissão poderia ser usada para determinar a temperatura do objeto. Esse problema só foi resolvido em 1900 quando Max Planck descreve o comportamento da emissão de radiação em corpo negro através da Lei de Planck da Radiação de Corpo Negro, onde ele assume que a energia deveria estar quantificada, ou seja, a energia seria emitida em pacotes de energia, e não poderia assumir valores intermediários a esses pacotes de energia. Esses pacotes de energia foram chamados de quantum, ou quanta, no plural, essa expressão vem do latim e significa "quantidade" ou "quanto". Assim surge o nome física quântica, ou mecânica quântica. Uma teoria que quebra completamente com o paradigma da mecânica clássica Newtoniana.

### 2.1 NOTAÇÃO DE DIRAC

Em Mecânica Quântica podemos representar os estados de um determinado sistema quântico através da Notação de Dirac [8]. Nessa notação os estados de um

determinado sistema serão representados por um "ket" ( $|\Psi\rangle$ ).

Tomemos um espaço vetorial de 3 dimensões  $E = \{|\Psi_1\rangle, |\Psi_2\rangle, |\Psi_3\rangle\}$ , podemos representar um estado  $|\Psi\rangle$  nessa base vetorial com:

$$|\Psi\rangle = \alpha_1|\Psi_1\rangle + \alpha_2|\Psi_2\rangle + \alpha_3|\Psi_3\rangle \quad (1)$$

Podemos também representar esse estado em uma matriz do tipo coluna:

$$|\Psi\rangle = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix}$$

Além disso, podemos definir "bra"  $\langle\Psi|$  como sendo o conjugado transposto de  $|\Psi\rangle$ .

$$\langle\Psi| = |\Psi\rangle^\dagger$$

$$\langle\Psi| = \left( \alpha_1^* \quad \alpha_2^* \quad \alpha_3^* \right)$$

## 2.2 POSTULADOS DA MECÂNICA QUÂNTICA

Nesta seção serão descritos os principais postulados da mecânica quântica, baseado em [6].

### 2.2.1 POSTULADO I - DESCRIÇÃO DOS ESTADOS QUÂNTICOS

Para cada instante de  $t$ , o estado de um determinado sistema físico pode ser representado por um ket  $|\Psi\rangle$ , que representa um vetor no espaço de Hilbert. Dessa forma, a superposição de dois ou mais estados é também um estado do sistema.

$$|\Psi\rangle = \alpha_1 |\Psi_1\rangle + \alpha_2 |\Psi_2\rangle$$

Onde  $\alpha_1$  e  $\alpha_2$  são números complexos, chamados de amplitude. Considerando  $|\Psi_1\rangle$  o estado de uma partícula em uma determinada posição e  $|\Psi_2\rangle$  o estado dessa mesma partícula em uma outra posição,  $|\Psi\rangle$  indica o estado de superposição entre  $|\Psi_1\rangle$  e  $|\Psi_2\rangle$ . Nesse caso, a partícula estaria em duas posições ao mesmo tempo, algo que vai na contramão da nossa intuição e da física clássica, mas que é observado em experimentos como o da fenda dupla em eletrons.

O espaço de Hilbert é um espaço vetorial sobre os números complexos, com dimensão (vetores na base) finita ou infinita e dotado da operação de produto interno. O produto interno entre um bra  $\langle\Psi|$  e um ket  $|\phi\rangle$  é dado por  $\langle\Psi|\phi\rangle$ .

## 2.2.2 POSTULADO II - DESCRIÇÃO QUÂNTICA DE OBSERVÁVEIS

Cada atributo observável de um sistema físico, como momento linear ou posição, por exemplo, pode ser descrito por um operador linear Hermitiano correspondente que age sobre os ket do sistema. A matriz  $H = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$  é um exemplo de operador hermitiano.

Operadores Hermitianos são operadores lineares em que o conjugado transposto (adjunto) do operador é igual ao próprio operador. Para um operador  $\hat{A}$ , segue que:  $\hat{A} = \hat{A}^\dagger$ .

## 2.2.3 POSTULADO III - MEDIDAS SOBRE ESTADOS QUÂNTICOS

Os únicos resultados possíveis ao medir um observável  $A$ , são os autovalores associados ao operador  $\hat{A}$  que descrevem esse operador.

## 2.2.4 POSTULADO IV - PROBABILIDADE DE MEDIDA

Ao medir um observável  $A$  em um estado  $|\Psi\rangle$ , a probabilidade associada de se obter um autovalor  $a_n$  é dada pelo quadrado do produto interno de  $|\Psi\rangle$  com o autoestado  $|\Psi\rangle, \langle a_n|\Psi\rangle$ .

## 2.2.5 POSTULADO V - EVOLUÇÃO COM O TEMPO

A evolução com o tempo de um sistema quântico preserva a norma do ket. Ou seja, a evolução com o tempo de um sistema quântico é dada por  $|\Psi(t)\rangle = \hat{U}(t, t_0) |\Psi(t_0)\rangle$  para um operador unitário  $\hat{U}$ .

A preservação da norma do estado está associada à conservação da probabilidade. Assim, para que possamos conservar amplitudes de probabilidade na computação quântica, usamos matrizes unitárias para constituir as portas lógicas quânticas.

## 2.2.6 POSTULADO VI - SISTEMAS COMPOSTOS

Alguns eventos decorridos em sistemas quânticos fazem uso de dois ou mais vetores de estados unidos no espaço de Hilbert, isto significa que sistemas deste tipo comportam-se como um sistema composto. Sejam dois vetores de estados  $|\Psi_1\rangle$  e  $|\Psi_2\rangle$ , o espaço de Hilbert  $|\Phi\rangle$  associado ao sistema composto entre  $|\Psi_1\rangle$  e  $|\Psi_2\rangle$ , é o produto tensorial dos espaços de Hilbert associados aos sistemas simples  $|\Psi_1\rangle$  e  $|\Psi_2\rangle$ .

$$|\Phi\rangle = |\Psi_1\rangle \otimes |\Psi_2\rangle$$

Esse postulado é extremamente útil em computação quântica pois nos permite representar sistemas com mais de um qubit. Exploraremos mais afundo suas consequências na seção 3.1.

### 2.3 SUPERPOSIÇÃO

O princípio da superposição quântica diz que a combinação linear de dois ou mais vetores de estados no mesmo espaço de Hilbert, também é um estado do sistema, ou em outras palavras, se existem diversos estados possíveis para um sistema, também existe um estado em que esses estados coexistem, ou existem como combinação desses diversos estados [1]. Como descrito no Postulado I.

### 2.4 REGRA DE BORN

Se temos um sistema em superposição de estados, cada um com uma amplitude associada, regra de Born diz que o módulo ao quadrado da amplitude de determinado estado, dá a probabilidade de se obter esse estado após a medição do sistema [1]. Ou seja, a soma do quadrado do módulo da amplitude de todos os estados deve ser sempre igual a 1. Para o  $|\Psi\rangle = \alpha |0\rangle + \beta |1\rangle$ , temos:

$$|\alpha|^2 + |\beta|^2 = 1 \quad (2)$$

### 2.5 EMARANHAMENTO

O emaranhamento (ou entrelaçamento) é um caso de superposição quântica especial, em que dois sistemas diferentes estão relacionados de tal forma que a medição em um dos sistemas afeta diretamente o valor do outro sistema [1], ou seja, o valor dos dois sistemas são inseparáveis.

### 3 COMPUTAÇÃO QUÂNTICA

A Computação Quântica pode ser definida como a utilização das propriedades quânticas, como superposição e emaranhamento quântica, além do formalismo matemático da física quântica, para fazer computação, ou seja, criar algoritmos que consigam utilizar essas propriedades quânticas para fazer operações e manipular a informação.

Em computação clássica, forma de computação que utiliza as leis da física clássica para fazer computação, a menor unidade de informação está contida em uma unidade chamada bit (simplificação para dígito binário, "*Binary digit*", em inglês. Essa unidade pode assumir dois valores possíveis, 0 ou 1. Com isso, podemos representar os números em base 2 e fazer diversas operações utilizando a lógica booleana. Já em computação quântica, a menor unidade de informação é o qubits, ou bit quântico. O qubit pode assumir os valores  $|0\rangle$  e  $|1\rangle$ , como na computação clássica, mas pode também assumir um estado de superposição dos os estados  $|0\rangle$  e  $|1\rangle$ . O capítulo 3 foi feito baseado em [1].

#### 3.1 QUBIT

Os computadores quânticos utilizam uma base no espaço de Hilbert  $B = \{|0\rangle, |1\rangle\}$  de dimensão igual a 2. Como visto anteriormente, o qubit pode ser representado utilizando a notação de Dirac, além disso, um estado genérico  $|\Psi\rangle$  será dado por:

$$|\Psi\rangle = \alpha_1 |0\rangle + \alpha_2 |1\rangle$$



Onde  $\alpha_1$  e  $\alpha_2$  são números complexos, e representam as amplitudes associadas aos estados  $|0\rangle$  e  $|1\rangle$ . Além disso,  $\alpha_1$  e  $\alpha_2$  deve respeitar a relação  $|\alpha_1|^2 + |\alpha_2|^2 = 1$ .

Também podemos representar  $|0\rangle$  e  $|1\rangle$  através de matrizes.

$$|0\rangle = 1|0\rangle + 0|1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$|1\rangle = 0|0\rangle + 1|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Para um estado de superposição máxima, temos:

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Podemos obter o valor de um sistema quântico de dois qubits a partir do produto tensorial dos dois qubits que formam esse sistema. Tomemos como exemplo um sistema formado por  $|0\rangle$  e  $|1\rangle$

$$|0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ 0 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = |01\rangle$$

### 3.2 ESFERA DE BLOCH

Existem diversas maneira de representar um qubit. Podemos representar através da notação de Dirac, como vimos anteriormente, mas podemos também representar-lo através da esfera de bloch. Um estado de um qubit qualquer  $|\Psi\rangle$  pode ser representado como um vetor que começa no centro e termina na superfície da esfera de bloch.

A esfera de bloch tem raio igual a 1, e fazendo uma analogia com o planeta Terra, tem a representação do estado  $|1\rangle$  no polo norte da esfera, e a representação do estado  $|0\rangle$  no polo sul da esfera. Tomando um estado  $|\Psi\rangle$  qualquer, dado por:

$$|\Psi\rangle = \alpha_1 |0\rangle + \alpha_2 |1\rangle$$

Podemos representa-lo em notação polar como:

$$|\Psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right)e^{i\phi}|1\rangle \quad (3)$$

Com  $\theta \in [0, \pi]$  e  $\phi \in [0, 2\pi[$ . Ainda com a analogia do planeta Terra, temos o angulo  $\theta$  como a latitude da esfera e o angulo  $\phi$  como a longitude da esfera.

Essa representação é interessante, já que nos permite ter uma intuição geométrica das manipulações que faremos no qubit, ainda que seja contraintuitiva, já que estaremos representando visualmente em 3 dimensões, estados de

um espaço de Hilbert de dimensão igual a 2. Assim, temos estados que são fisicamente idênticos, como os estados  $|0\rangle$  e  $-|0\rangle$ .

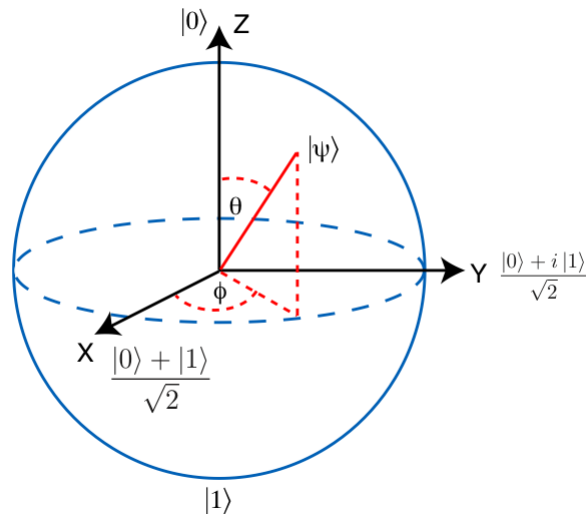


Figura 1: Esfera de Bloch.

### 3.3 OPERADORES QUÂNTICOS

Em Computação Quântica, utilizamos operadores lineares unitários, para manipular os nossos vetores de estado dos nossos qubits, realizando assim operações lógicas diversas. Operadores unitários (não confundir com operadores unários) são operadores lineares que preservam a norma (produto interno) do vetor que ele atua, conservando assim a Regra de Born.

Podemos fazer analogias entre esses operadores com as portas lógicas clássicas, utilizadas em computação clássica. Por isso, costumamos chamar esses operadores de portas lógicas quânticas e costumamos representá-los através

de circuitos muito parecidos com os circuitos lógicos utilizados em computação clássica, como veremos na seção abaixo.

### 3.3.1 OPERADORES UNÁRIOS

Os operadores unários são operadores aplicados a um qubit e são representados por uma matriz  $2 \times 2$ . Os três primeiros operadores estudados são os operadores de Pauli (X, Y e Z) que fazem rotações na esfera de Bloch nos eixos x, y e z, respectivamente.

#### 3.3.1.1 OPERADOR X (NOT)

O primeiro operador estudado é o operador X, que equivale ao operador NOT em computação clássica, fazendo o papel de bit flip. Esse operador tem esse nome pois realiza uma rotação de  $180^\circ$  em torno do eixo x, e pode ser representado matricialmente por:

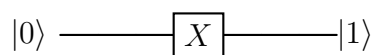
$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Aplicando esse operador aos estados  $|0\rangle$  e  $|1\rangle$ , temos:

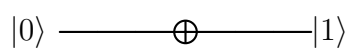
$$X|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0+0 \\ 1+0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$$

$$X|1\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0+1 \\ 0+0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle$$

Em um diagrama de circuitos quânticos, podemos representar o operador X de duas diferentes maneiras, como mostram o exemplo abaixo:



ou

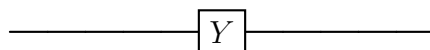


### 3.3.1.2 OPERADOR Y

O operador Y realiza uma rotação de  $180^\circ$  em torno do eixo y na esfera de bloch e pode ser definido matricialmente como:

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

Além de poder ser definido em diagramas de circuitos quânticos como:

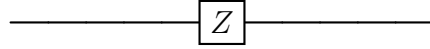


### 3.3.1.3 OPERADOR Z

O operador Z realiza uma rotação de  $180^\circ$  em torno do eixo z na esfera de bloch e pode ser definido matricialmente como:

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

Além de poder ser definido em diagramas de circuitos quânticos como:



Aplicando o operador Z a um estado computacional qualquer  $j$ , temos:

$$Z |j\rangle = (-1)^j |j\rangle$$

Esse operador é muito importante pois ele aplica uma mudança de fase (phase flip, em inglês) no estado  $|1\rangle$  e não no estado  $|0\rangle$

$$Z|0\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1+0 \\ 0+0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = (-1)^0 |0\rangle = |0\rangle$$

$$Z|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0+0 \\ 0-1 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix} = (-1)^1 |1\rangle = -|1\rangle$$

### 3.3.1.4 OPERADOR $RZ_\varphi$

O operador  $RZ_\varphi$  realiza uma rotação de  $\varphi$  radianos em torno do eixo z na esfera de bloch e pode ser definido matricialmente como:

$$RZ_\varphi = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{pmatrix}$$

Podemos notar que o operador Z nada mais é do que um caso especial do operador  $RZ_\varphi$ , onde  $\varphi = \pi$ . Pela identidade de Euler temos que  $e^{i\varphi} = -1$ , então

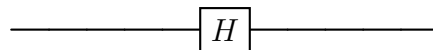
podemos substituir  $e^{i\pi}$  por -1 na matrix de Z. Esse operador é especialmente importante em computadores quânticos reais, pois com ele, conseguimos fazer diversas rotações diferentes em torno do eixo z.

### 3.3.1.5 OPERADOR DE HADAMARD

O operador de Hadamard é responsável por colocar um qubit que esteja em um estado da base computacional em uma superposição quântica de dois estados. Por isso, ele é um dos principais operadores quânticos, pois nos permite aproveitar essa propriedade quântica, sendo normalmente aplicado a todos os qubits no início do circuito. Podemos representá-lo matricialmente por:

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

Além de poder ser definido em diagramas de circuitos quânticos como:



Aplicando esse operador às bases computacionais temos:

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

$$H|1\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Podemos perceber que a aplicação do operador de Hadamard projeta os operadores da base em estados diferentes na esfera de bloch, mas ambos são estados de superposição. Esses estados são chamados de estados "mais" ( $|+\rangle$ ) e "menos" ( $|-\rangle$ ).

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

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

A porção  $\sqrt{2}$  nos lembra da Regra de Born, que diz que o quadrado do módulo da amplitude de determinado estado é a probabilidade de obter esse estado ao medir o sistema. Para  $|\Psi\rangle = \alpha |0\rangle + \beta |1\rangle$ , temos:

$$|\alpha|^2 + |\beta|^2 = 1$$

Para  $|+\rangle$ , temos:

$$\left| \frac{1}{\sqrt{2}} \right|^2 + \left| \frac{1}{\sqrt{2}} \right|^2 = \frac{1}{2} + \frac{1}{2} = 1$$

### 3.3.2 OPERADORES BINÁRIOS

Os operadores binários são operadores aplicados a dois qubits e são representados por uma matrizes 4 x 4.



### 3.3.2.1 OPERADOR SWAP

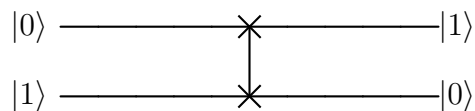
O operador Swap é um dos operadores binários mais importantes, principalmente por sua necessidade ao analisar a topologia de um computador quântico, como veremos mais a frente. Esse operador recebe dois qubits de entrada e troca o valor entre eles. Por exemplo, recebe o estado  $|01\rangle$  e devolve o estado  $|10\rangle$ , ou recebe o estado  $|10\rangle$  e devolve o estado  $|01\rangle$ . Podemos representá-lo matricialmente como:

$$\text{SWAP} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Aplicando o operador swap ao estado  $|01\rangle$ , temos:

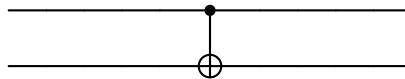
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = |10\rangle$$

Além disso, podemos representá-lo em diagramas quânticos como:



### 3.3.2.2 OPERADOR CNOT

O operador CNOT ou NOT controlado ("*Controlled NOT*", em inglês) também é outro operador de extrema importância para a computação quântica. Esse operador funciona de forma condicional, um dos qubits é o qubit de controle e o outro é o qubit alvo. Se o qubit de controle tiver valor  $|0\rangle$ , nada acontece com o qubit alvo. Porém, se o qubit de controle tiver valor  $|1\rangle$ , o qubit alvo sofrerá a ação da porta NOT (ou X). O qubit de controle é representado por um círculo pequeno, totalmente preenchido com a cor preta, enquanto que o qubit alvo é representado por um círculo com uma cruz no centro, conforme o exemplo a seguir:



A porta CNOT pode ser representada matricialmente por:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Aplicando a porta CNOT aos estados  $|01\rangle$  e  $|11\rangle$ , podemos observar seu comportamento:

$$\text{CNOT} |01\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = |01\rangle$$

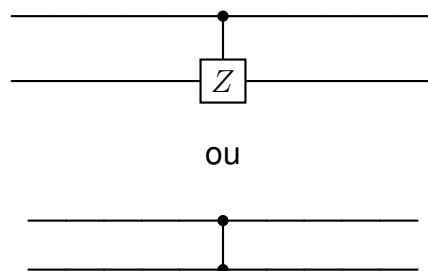
$$\text{CNOT } |11\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = |10\rangle$$

### 3.3.2.3 OPERADOR CZ

Outro operador muito importante, e muito parecido com o CNOT, é o operador CZ ou Z controlado ("*Controlled Z*", em inglês). Esse operador também funciona de forma condicional, assim como o operador CNOT, mas ao invés de aplicar um operador NOT quando o qubits de controle tiver valor igual a  $|1\rangle$ , aplicamos o operador Z, que realiza uma rotação de  $180^\circ$  em torno do eixo z na esfera de bloch. Esse operador pode ser representado matricialmente como:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Além disso, podemos representar esse operador em diagramas como a seguir:



### 3.4 AMBIENTE DE PROGRAMAÇÃO

Ao longo do projeto, utilizamos a biblioteca Qiskit (Quantum Information Science Kit), que é uma biblioteca de Python criada pela IBM [1], que nos possibilitou criar circuitos quânticos, portas quânticas, além de nos permitir acessar simuladores de computadores quânticos com até 5000 qubits e computadores quânticos reais, com até 127 qubits. Além disso, utilizamos a plataforma Google Colaboratory, para escrever e executar código Python pelo navegador [16].

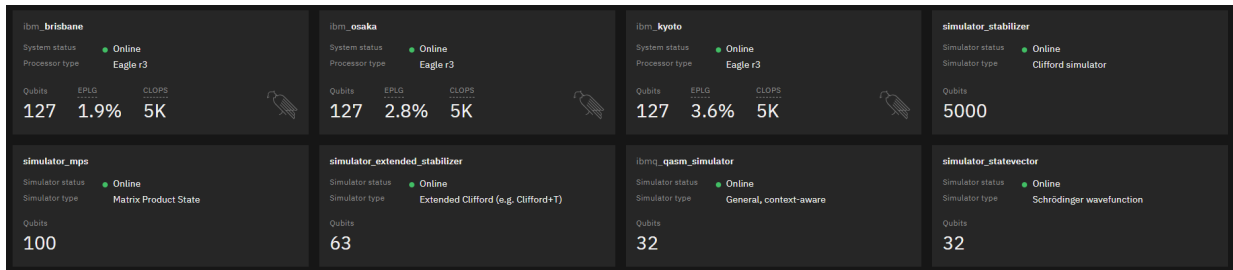


Figura 2: Computadores Quânticos e Simuladores de Computadores Quânticos da IBM disponíveis

```
✓ 2s ▶ from qiskit import QuantumCircuit

def init(qc, qubits):
    for q in qubits:
        qc.h(q)
    return qc

grover_circuit = QuantumCircuit(2)
grover_circuit = init(grover_circuit, [0, 1])
grover_circuit.draw(output="mpl", style="iqp")

grover_circuit.cz(0,1) # oraculo
grover_circuit.draw(output="mpl", style="iqp")

#difusor
grover_circuit.h([0,1])
grover_circuit.z([0,1])
grover_circuit.cz(0,1)
grover_circuit.h([0,1])

grover_circuit.draw(output="mpl", style="iqp")
```

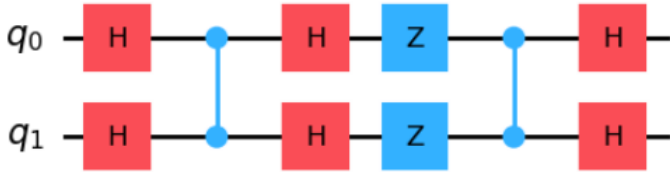


Figura 3: Exemplo de Circuito Quântico criado com a biblioteca Qiskit, usando o Google Colaboratory

## 4 ALGORITMO DE GROVER

Em 1996, Lov Grover propôs um algoritmo quântico de busca que obtinha um ganho quadrático em comparação com sua implementação clássica. Conhecido como Algoritmo de Grover, esse algoritmo é especialmente interessante como uma alternativa pra solucionar problemas de busca não estruturada, ou busca não ordenada, em que é muito difícil de calcular a solução da busca, mas ao tomar um candidato de solução, é fácil verificar se esse candidato é ou não solução do problema. Podemos pensar no caso de uma sequência não ordenada de números, como ilustra a figura 4, em que não há uma maneira de calcular qual o número de cada casa, onde estamos procurando, por exemplo, pelo número 10. Para achar o número 10 eu devo conferir a primeira casa, e verificar se esse número é o número 10.

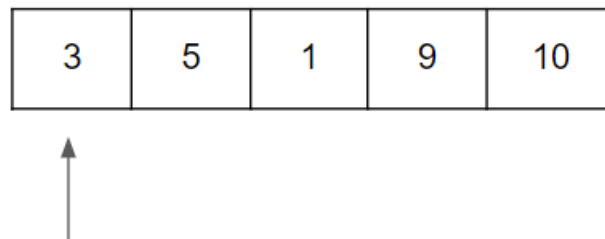


Figura 4: Busca não ordenada

Se não for, confiro a segunda casa e assim por diante até encontra-lo. Podemos notar que é fácil conferir se nosso palpite é ou não solução, mas não é possível calcular essa solução. Logo, temos uma busca por força bruta, com complexidade computacional  $O(n) = n$ , em que para uma lista de tamanho  $n$ ,

teremos que rodar o algoritmo  $n$  vezes, no pior dos casos. Com o Algoritmo de Grover temos um ganho quadrático, com  $O(n) = \sqrt{n}$  [5].

Nesse capítulo explicaremos com riqueza de detalhes o funcionamento do algoritmo, desde sua formulação matemática, até sua implementação em circuitos quânticos, e estudaremos uma possível aplicação do Algoritmo de Grover.

#### 4.1 FUNCIONAMENTO DO ALGORITMO DE GROVER

O Algoritmo de Grover utiliza a superposição dos qubits e a interferência quântica para solucionar um problema de busca não estruturada. Ele faz isso através de um mecanismo de amplificação de amplitudes, amplificando a amplitude da solução do problema e diminuindo a amplitude das demais possibilidades.

Tomemos como exemplo um problema em que queremos encontrar o ket  $|11\rangle$  e temos um vetor  $|\psi\rangle$  em superposição uniforme, dado por:

$$|\psi\rangle = \frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle + \frac{1}{2}|11\rangle$$

Através do Algoritmo de Grover iremos amplificar a amplitude associada ao ket  $|11\rangle$  e reduzir as demais amplitudes, obtendo o vetor abaixo:

$$|\psi\rangle = \frac{1}{\sqrt{12}}|00\rangle + \frac{1}{\sqrt{12}}|01\rangle + \frac{1}{\sqrt{12}}|10\rangle + \frac{\sqrt{3}}{2}|11\rangle$$

Esse procedimento de amplificação de amplitude ocorre em duas etapas principais, uma inversão de fase da amplitude associada a solução (*phase quick-back trick* ou *phase flip*, em inglês), realizada por um operador  $U_w$  (figura 5), e

posteriormente, uma inversão de todas as amplitudes ao redor da média das amplitudes, realizada por um operador  $U_s$  (figura 6). Esse conjunto de operações são chamados de iterada de Grover ou operador de Grover e serão aplicados diversas vezes para aumentar a probabilidade de se medir o estado solução do problema [1].

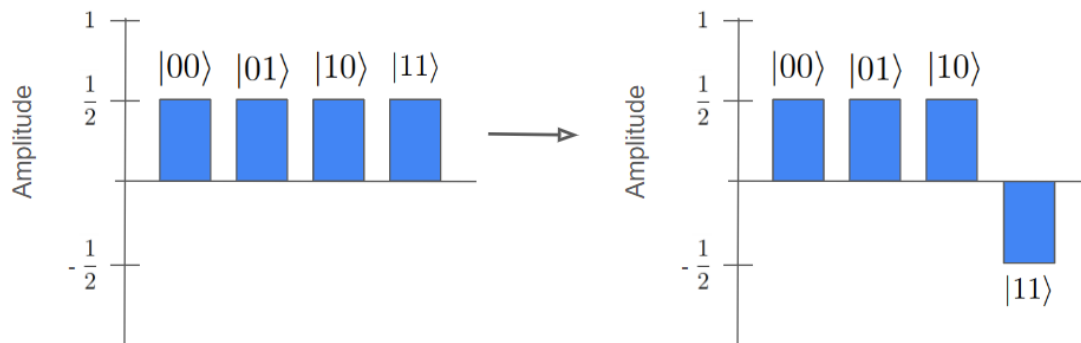


Figura 5: Inversão de fase da amplitude associada a solução

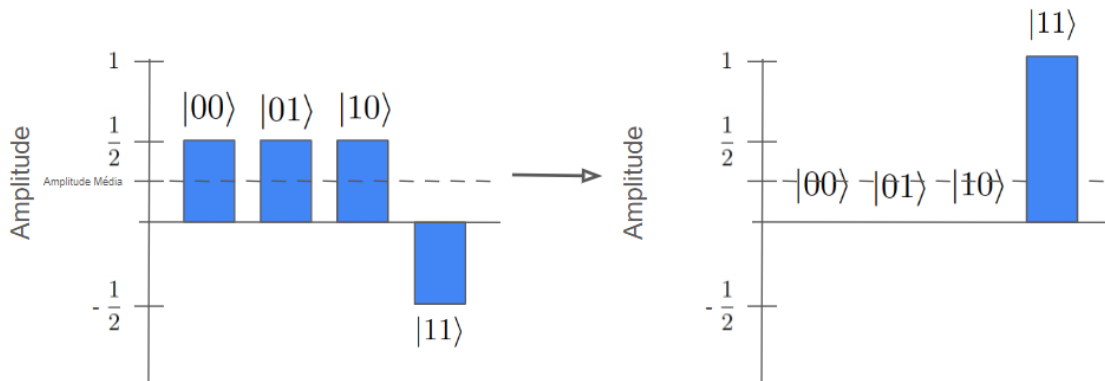


Figura 6: Inversão de todas as amplitudes ao redor da média das amplitudes



A inversão de fase da amplitude associada a solução é realizada por uma função oráculo. Essa função  $f(x)$  é responsável por identificar a solução do problema de modo que se  $x$  é solução,  $f(x)$  é igual a 1, e se  $x$  não é solução,  $f(x) = 0$ , conforme veremos abaixo. Já a etapa de inversão de todas as amplitudes ao redor da média das amplitudes é chamada de etapa de difusão e é realizada por um operador de difusão de Grover.

Uma maneira muito útil de entender o Algoritmo de Grover é através de sua representação geométrica em duas dimensões. Cada um dos vetores é a representação de estados quânticos que estão presentes em um espaço de Hilbert. Tomando  $|\omega\rangle$  como o vetor que representa o estado solução do nosso problema, representado na vertical,  $|\omega_{\perp}\rangle$  como um vetor perpendicular (ortogonal) ao vetor  $|\omega\rangle$ , e  $|s\rangle$  como o vetor de superposição uniforme (Figura 7), podemos aplicar um operador  $U_{\omega}$  no vetor  $|s\rangle$ , realizando uma rotação ao redor do vetor  $|\omega_{\perp}\rangle$  (Figura 8). Posteriormente, podemos aplicar um operador  $U_s$  que faz uma reflexão do vetor  $U_{\omega}|s\rangle$  ao redor do vetor  $|s\rangle$  resultando no vetor  $U_s U_{\omega}|s\rangle$  (Figura 9). Podemos notar que se fizermos esse procedimento iterativamente  $r$  vezes, estaremos nos aproximando do vetor  $|\omega\rangle$ . Essa é uma intuição geométrica do que acontece com os vetores de estado no algoritmo de Grover [5].

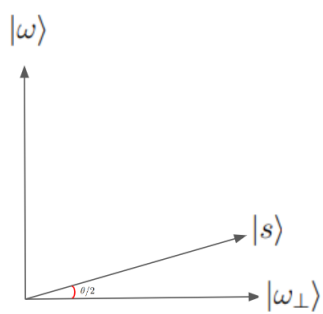


Figura 7: Algoritmo de Grover - representação vetorial

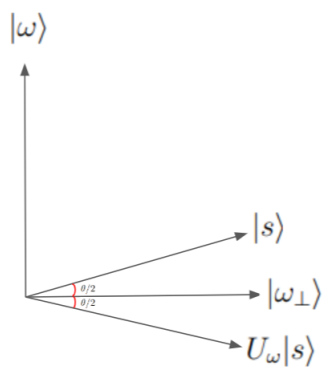


Figura 8: Algoritmo de Grover - representação vetorial (Inversão de fase da amplitude associada a solução)

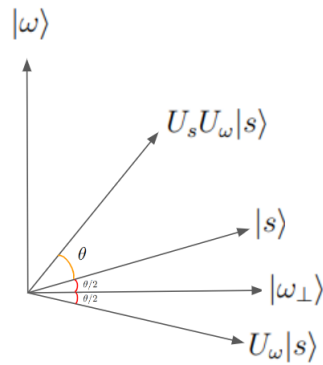


Figura 9: Algoritmo de Grover - representação vetorial (Inversão de todas as amplitudes ao redor da média das amplitudes)

Podemos representar essas transformações através de um circuito com portas lógicas quânticas como mostrado na figura 11. Onde  $U_\omega$  é o operador que implementa a função oráculo e  $U_s$  é o operador de difusão de Grover. Esses dois operadores são aplicados  $r$  vezes para aumentar a probabilidade de se obter a solução do problema.

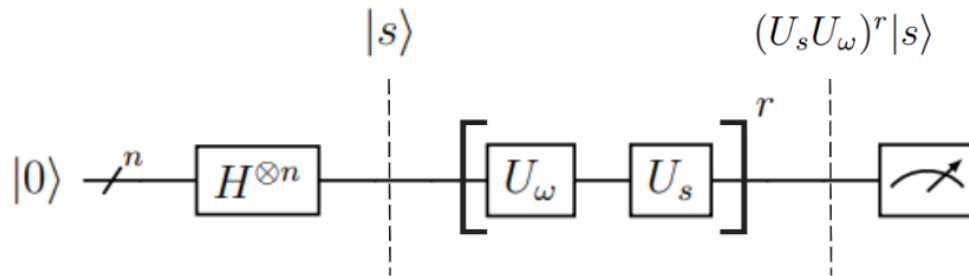


Figura 10: Circuito geral do Algoritmo de Grover

O operador  $U_s$  é igual a  $2|s\rangle\langle s| - I$ , ou ainda,  $H^{\otimes n}(2|0^n\rangle\langle 0^n| - I)H^{\otimes n}$ , onde  $|s\rangle\langle s|$  é o produto externo e  $I$  é a identidade. Esse operador é responsável

por rotacionar o ket que ele atua ao redor de  $|s\rangle$ , o que equivale a fazer uma intervenção ao redor da média no Algoritmo de Grover.

$$U_s = 2|s\rangle\langle s| - I = H^{\otimes n}(2|0^n\rangle\langle 0^n| - I)H^{\otimes n} \quad (4)$$

O operador  $U_\omega$  é igual a  $I - 2|\omega\rangle\langle\omega|$  e pode ser representado como uma matriz diagonal, com a diagonal principal igual a  $(-1)^{f(x)}$ , com  $x$  indo de 0 a  $N - 1$ , e  $N$  como a dimensão do espaço de Hilbert, ou ainda,  $N = 2^n$ , sendo  $n$  o número de qubits. Como mencionado anteriormente, esse operador é responsável por realizar rotação ao redor do vetor  $|\omega_\perp\rangle$ , o que equivale a implementar a função oráculo  $f(x)$  que identifica a solução do problema e inverte a fase da solução.

$$U_\omega = I - 2|\omega\rangle\langle\omega| \quad (5)$$

$$U_\omega = \begin{pmatrix} (-1)^{f(0)} & 0 & \dots & 0 \\ 0 & (-1)^{f(1)} & \dots & 0 \\ \dots & 0 & \dots & \dots \\ 0 & 0 & \dots & (-1)^{f(N-1)} \end{pmatrix}$$

$$U_\omega |x\rangle = (-1)^{f(x)} |x\rangle = \begin{cases} -|x\rangle, & \text{para } x = w, \text{ logo } f(x) = 1 \\ |x\rangle, & \text{para } x \neq w, \text{ logo } f(x) = 0 \end{cases}$$

Para o exemplo mencionado anteriormente nesse capítulo, onde nosso estado solução é o estado  $|11\rangle$ , podemos implementar o circuito da figura, utili-

zando a linguagem Qiskit. Nesse caso, o operador CZ faz o papel do oráculo, enquanto que o operador de difusão de Grover é composto por um operador de Hadamard em cada qubits, um operador Z de Pauli em cada qubit, um operador CZ entre os qubits e finalmente um operador de Hadamard em cada qubit.

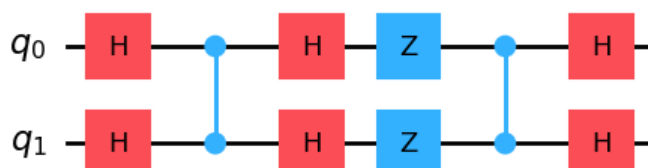


Figura 11: Circuito do Algoritmo Grover em Qiskit, para encontrar o  $|11\rangle$

A grande dificuldade quando se trata de implementar o Algoritmo de Grover é implementar uma função oráculo que nos permita identificar as soluções do nosso problema, e que faça isso de forma eficiente, de maneira que seja mais vantajoso ao comparar com um computador clássico. Ao longo do capítulo 6 iremos explorar uma possível aplicação do Algoritmo de Grover.

## 5 MÁQUINA DE VETOR DE SUPORTE QUÂNTICA

Ao longo desse capítulo iremos explorar uma promissora aplicação da computação quântica com o aprendizado de máquina. Utilizando as propriedades quânticas de superposição e emaranhamento, podemos buscar novas soluções para resolver problemas que aparentam ser de difícil resolução utilizando técnicas em computadores clássicos. Existem diferentes algoritmos de aprendizado de máquina quântico, "*Quantum Machine Learning (QML)*", em inglês, que vão desde métodos simples de classificação e regressão até redes neurais mais complexas. Neste trabalho escolhemos nos aprofundar na técnica de Máquina de vetor de suporte quântica, "*Quantum Support Vector Machine (QSVM)*", em inglês, que é um algoritmo de classificação binária que combina o algoritmo de máquina de vetor de suporte (SVM) desenvolvido para computadores clássicos, com alguns passos sendo executados em computadores quânticos, afim de aproveitar suas propriedades para executar tarefas que não são executados de forma simples em computadores clássicos. Dessa maneira, o QSVM é um algoritmo híbrido que combina computação clássica e quântica.

### 5.1 IMPLEMENTAÇÃO DA MÁQUINA DE VETOR DE SUPORTE (SVM)

A máquina de vetor de suporte é um modelo muito efetivo e muito bem estudado, consistindo em um modelo supervisionado de classificação binária e linear. Isto é, um modelo que ao receber um conjunto de dados  $C_d$  com  $n$  amostras  $C_d = \{\{x_1, y_1\}, \dots, \{x_n, y_n\}\}$ , sendo  $x$  o vetor de características, e  $y \in \{\pm 1\}$  a variável de resposta com duas possíveis classes, é capaz de "aprender" o comportamento

desse conjunto de dados, e gerar um hiperplano que separa com a maior margem possível essas duas classes, como mostra a figura 12. Desta maneira, o método é capaz de gerar previsões ao receber um conjunto de dados desconhecido.

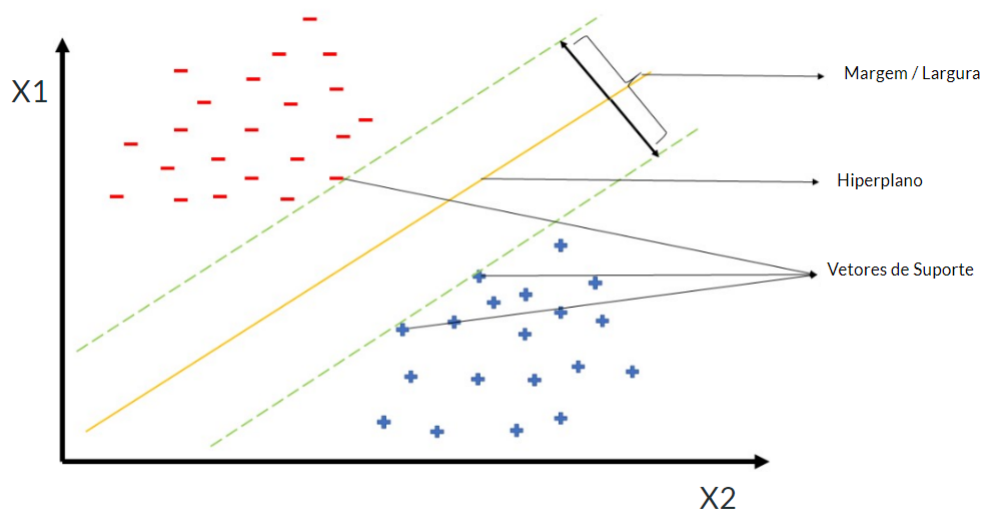


Figura 12: Máquina de vetor de Suporte

A predição entre essas duas classes ocorre através do resultado de um  $f^*(x) = \text{sign}(w \cdot x + b)$ , onde  $w \cdot x + b$  define o hiperplano que separa as classes,  $w \cdot x$  é o produto escalar entre  $x$  e um vetor de pesos  $w$ , e  $b$  é um valor de *offset* de controle. Se  $f^*(x) \geq 0$ ,  $x$  pertence a uma das classes e se  $f^*(x) < 0$ ,  $x$  pertence a outra classe [9].

Para construir o hiperplano, precisamos dos valores de  $w$  e  $b$ , e podemos encontrar esses valores através de uma função Lagrangiana  $L(\alpha)$ .

$$L(\alpha) = \frac{1}{2} - \sum_{i=1}^n \alpha_i [y_i(\vec{w} \cdot \vec{x}_i + b) - 1] \quad (6)$$

Sendo  $\alpha_i$  os coeficientes que minimizam essa função Lagrangiana. Fazendo algumas manipulações algébricas chegamos a expressão a seguir:

$$L(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n [\alpha_i \alpha_j y_i y_j \vec{x}_i \cdot \vec{x}_j] \quad (7)$$

Podemos notar que agora essa expressão só depende de  $\alpha$ ,  $\vec{x}_j$  e  $y_j$ .

Porém, existem casos em que não é possível traçar um hiperplano que separe os dois grupos. Nesse caso utilizamos um mapa de características  $M(x)$ , como ilustra a figura 13, que mapeia os pontos de  $x$  em um espaço vetorial de dimensão maior, onde é possível traçar esse hiperplano que separa os dois grupos.

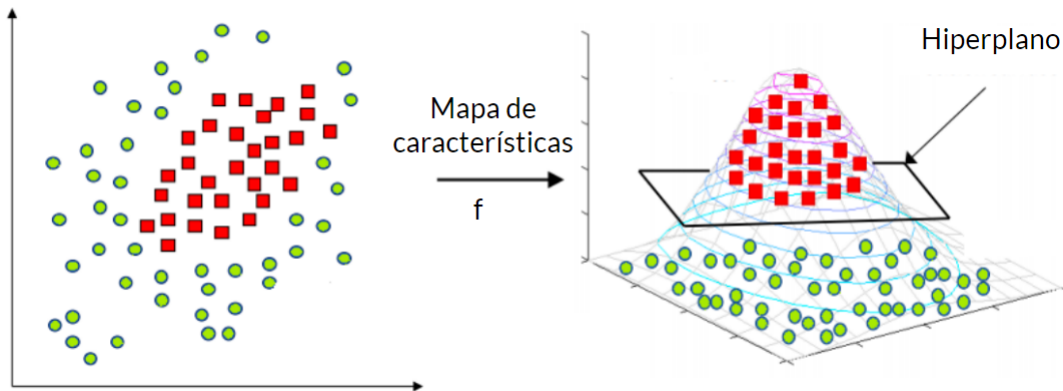


Figura 13: Mapa de Características

Utilizando o mapa de características, a função Lagrangiana fica:



$$L(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n [\alpha_i \alpha_j y_i y_j M(\vec{x}_i) M(\vec{x}_j)] \quad (8)$$

A expressão acima pode ser escrita em função do produto interno  $M(\vec{x}_i) \cdot M(\vec{x}_j)$ . Esse produto interno entre dos vetores resultantes da aplicação do mapa de características é chamado de função Kernel  $K(u, v) = M(\vec{x}_i) \cdot M(\vec{x}_j)$  e define todos os pares de produtos internos para um mapa de características. A utilização de funções Kernel nos permite modelar de forma complexa e não linear nosso conjunto de dados. Com a função Kernel, a equação do Lagrangiano fica:

$$L(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n [\alpha_i \alpha_j y_i y_j K(\vec{x}_i, \vec{x}_j)] \quad (9)$$

## 5.2 IMPLEMENTAÇÃO QUÂNTICA DA MÁQUINA DE VETOR DE SUPORTE (QSVM)

Podemos alinhar a computação quântica e a máquina de vetor de suporte para classificar conjuntos de dados que são difíceis de classificar usando funções kernel clássicas. Dessa maneira, podemos utilizar o espaço de Hilbert, em que os qubits estão para criar o mapa de características e representar nosso conjunto de dados em um espaço de dimensão maior do que o inicial. Note que ainda precisamos enviar esses dados do computador quântico para o computador clássico, para que o computador clássico possa realizar o problema de otimização que é minimizar o Lagrangiano, encontrando assim  $w$  e  $b$  que nos permitem classificar novas amostras [9].

Dessa maneira, não podemos enviar para o computador clássico os estados  $|\Psi(\vec{x}_i)\rangle$  e  $|\Psi(\vec{x}_j)\rangle$ , pois para isso precisaríamos ler o estado dos qubits e isso quebraria a superposição, perdendo assim o vetor representado no espaço de Hilbert. Logo, só precisamos enviar para o computador clássico o kernel  $K_{ij} = \langle \Psi(\vec{x}_i) | \Psi(\vec{x}_j) \rangle$  que representa o produto interno para um determinado mapa de características.

A escolha do mapa de característica é provavelmente uma das escolhas mais importantes ao projetar uma máquina de vetor de suporte quântica. Estudaremos na próxima seção diferentes mapas de características formados por operadores unários de rotação de Pauli (X, Y, Z, XX, YY, ZZ).

## 6 ESTUDOS DE CASO

Nessa seção exploramos duas aplicações práticas da computação quântica, uma associada ao Algoritmo de Grover e uma associada a Máquina de Vetor de Suporte Quântica.

### 6.1 COLORAÇÃO DE GRAFOS (GROVER)

O problema da coloração de grafos é um clássico problema da teoria de grafos. Ele consiste em atribuir cores para os vértices de um grafo, de modo que dois vértices adjacentes (conectados), não tenham a mesma cor. A idéia é descobrir quais são as maneiras de colorir um grafo de modo que ele satisfaça a esse requisito. Esse problema surgiu de um outro problema, o problema da coloração de mapas, onde temos um mapa, de um país, por exemplo, e queremos colorir os estados de modo que dois estados que fazem fronteira não tenham a mesma cor. Esse problema pode ser reduzido a um problema de coloração de grafo, com os vértices representando os estados, e as arestas representam as fronteiras.

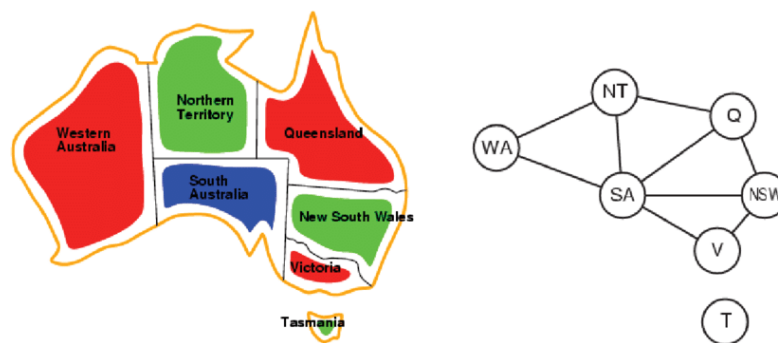


Figura 14: Problema da coloração de mapas expresso em um grafo

Podemos utilizar o algoritmo de Grover para tentar solucionar esse problema. Para isso, vamos pegar uma versão simplificado com apenas três vértices, conforme a figura 15.

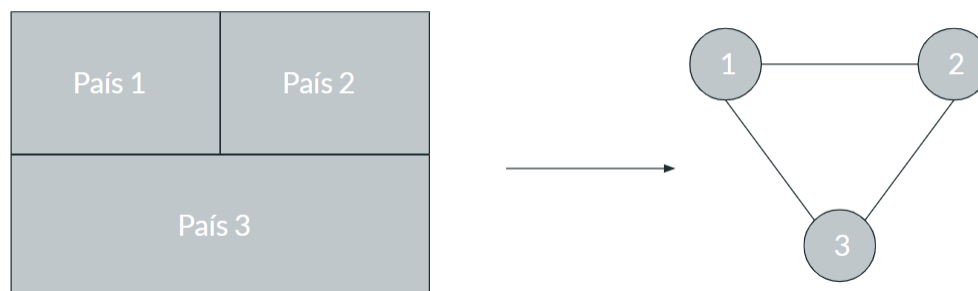


Figura 15: Grafo com 3 vértices

Todo problema de coloração de grafos pode ser resolvido com quatro cores [15], então, tomaremos quatro cores e iremos representá-las codificando seu valor em dois qubits, da seguinte maneira: 00 = branco, 01 = vermelho, 10, azul e 11 = preto. Desse modo, cada vértice será representado por dois qubits, que podem assumir um desses quatro valores. Usaremos mais 3 qubits de comparação, pois temos três comparações a serem feitas: vértice 1 - vértice 2, vértice 1 - vértice 3 e vértice 2 - vértice 3. Finalmente, teremos um qubit que armazenará se o sistema é solução, e fará o *phase-flip*. Portanto, temos 10 qubits, mas apenas 6 serão medidos, já que esses representam os valores dos vértices. Inicialmente, colocaremos esses 6 qubits em superposição uniforme, aplicando uma porta de Hadamard em cada um deles. Posteriormente, iremos aplicar nossa função oráculo, que compara os valores dos pares de qubits que representam cada um dos vértices, e verifica se esses valores são iguais entre si, e salva esse valor nos *check\_qubits*. Finalmente, usamos portas CNOT, CCNOT e CCCNOT para pas-

sar essa informação para o qubit de saída, como ilustra a figura 16. Podemos ver o código em Qiskit que implementa esse circuito na figura 17.

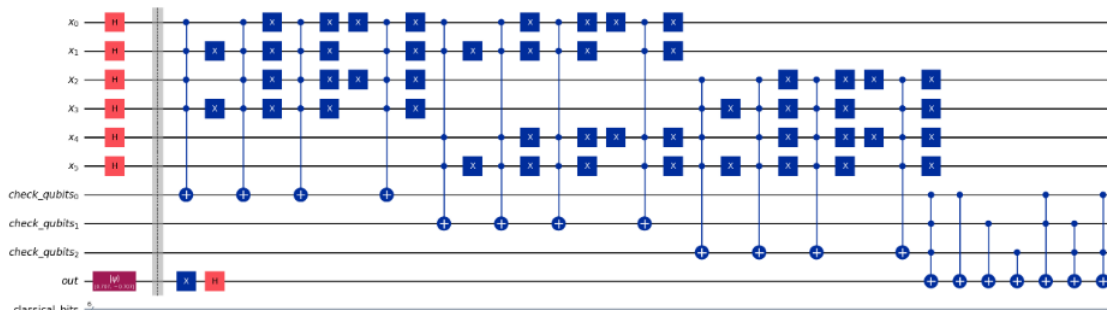


Figura 16: Circuito em Qiskit do Oráculo

```
[ ] def disagree_check(qc, qubits_a, qubits_b, check_qubit):

    # 1. Verifica se os qubits estão no estado 11 11.
    qc.mct(qubits_a + qubits_b, check_qubit)

    # 2. Verifica se os qubits estão no estado 01 01.
    qc.x(qubits_a[1])
    qc.x(qubits_b[1])
    qc.mct(qubits_a + qubits_b, check_qubit)
    qc.x(qubits_a[1])
    qc.x(qubits_b[1])

    # 3. Verifica se os qubits estão no estado 10 10.
    qc.x(qubits_a[0])
    qc.x(qubits_b[0])
    qc.mct(qubits_a + qubits_b, check_qubit)
    qc.x(qubits_a[0])
    qc.x(qubits_b[0])

    # 4. Verifica se os qubits estão no estado 00 00.
    qc.x(qubits_a)
    qc.x(qubits_b)
    qc.mct(qubits_a + qubits_b, check_qubit)
    qc.x(qubits_a)
    qc.x(qubits_b)
```

Figura 17: Código Qiskit do oráculo

Após o oráculo, precisamos aplicar o difusor de Grover para amplificar a

amplitude das soluções.

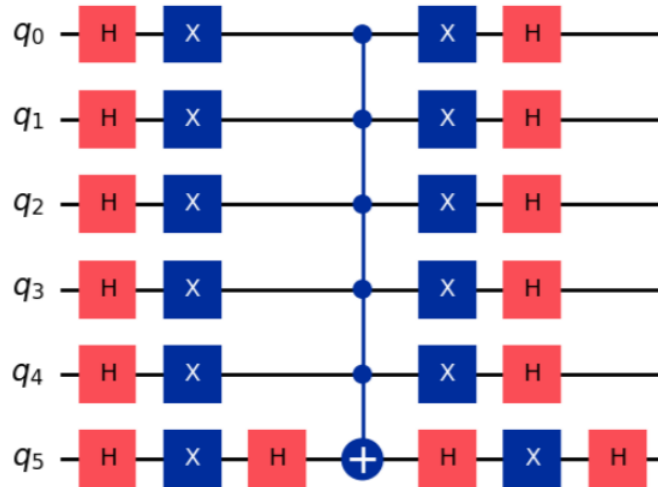


Figura 18: Circuito em Qiskit do Difusor de Grover

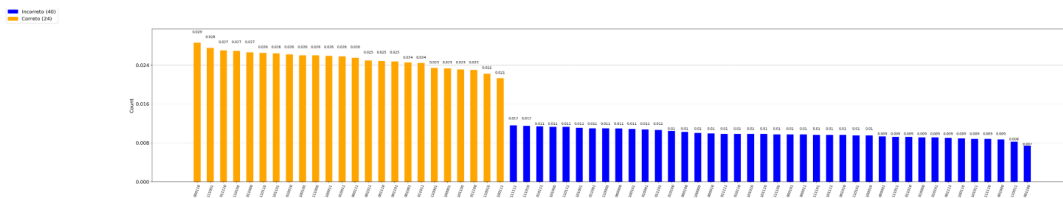


Figura 19: Amplitudes

Essa combinação de passos deve ser executada 8 vezes para aumentar a probabilidade das soluções. Essas amplitudes estão presentes na figura 19. Podemos concluir através desse exemplo que o Algoritmo de Grover consegue ser útil para resolver problemas de busca não estruturada. Entretanto, para o exemplo de coloração de grafos, precisamos adicionar dois qubits para cada novo vértice, e um qubit para cada nova aresta. Esse aumento no número de qubits é um

problema quando pensamos nos computadores quânticos atuais, onde existem limitações tanto no número de qubits disponíveis, quanto na conexão entre eles.

## 6.2 CLASSIFICAÇÃO (QSVM)

Como mencionado na seção 5, a máquina de vetor de suporte quântica se apresenta como uma aplicação promissora da computação quântica, combinando a representação implícita de um mapa de características no espaço de Hilbert através de uma função kernel, com a otimização de um Lagrangiano feita em um computador clássico. Nossa motivação nesse capítulo é fazer uma análise comparativa entre implementação clássica do SVM e a implementação quântica. Para a implementação clássica, utilizamos como mapa de características uma função kernel RBF. Já para a implementação quântica, optamos por variar a função kernel, afim de aprofundar um pouco no impacto dessa escolha nos resultados. Os mapas de características escolhidos foram mapas de operadores de rotação de Pauli com uma rotação (X, Y, Z) e operadores de rotação de Pauli com duas rotações (XX, YY, ZZ). Usando diferentes permutações de Pauli, conseguimos controlar os qubits ao redor da esfera de Bloch de maneiras diferentes, o que possibilita mapear de maneiras diferentes os conjuntos de dados.

Para fazer essa análise comparativa, optamos por utilizar apenas a precisão como métrica na comparação. Além disso, utilizamos 4 conjuntos de dados diferentes disponibilizados pela biblioteca de aprendizado de máquina Scikit-Learn <sup>1</sup>, comprimidos para duas dimensões utilizando a técnica de PCA (Algoritmo de Análise de componente principal) [17], que nos permite reduzir o número de dimensões de um conjunto de dados mantendo o maior número possível de informações. Além disso, separamos os conjuntos de dados em 70% como conjunto de treinamento e 30% como conjunto de testes.

---

<sup>1</sup><https://scikit-learn.org/stable/>



```

reps = 2
seed = 10240
feature_dim = 2

# Definindo instância do simulador
backend2 = QuantumInstance(Aer.get_backend('qasm_simulator'), shots=1024, seed_simulator=seed, seed_transpiler=seed)

# Definindo mapa de características Pauli X
feature_map = PauliFeatureMap(feature_dimension=feature_dim, reps=reps, paulis = ['X'])
# Criando Kernel com mapa de características Pauli X
kernel2 = QuantumKernel(feature_map=feature_map, quantum_instance=backend2)

qsvc = QSVC(quantum_kernel=kernel2)

# Treinando o modelo
qsvc.fit(X_train, y_train)

# Analisando a precisão para conjunto de teste
qsvc_score=qsvc.score(X_test, y_test)

```

Figura 20: Código em Qiskit da criação do mapa de características Pauli X, treinamento do modelo e obtenção da precisão para o conjunto de teste

## 6.2.1 CONJUNTOS DE DADOS UTILIZADOS

Os 4 conjuntos de dados da biblioteca de aprendizado de máquina Scikit-Learn utilizados foram o de câncer de mama, o de vinhos e o de íris.

### 6.2.1.1 CONJUNTO DE DADOS 1 - CÂNCER DE MAMA

O conjunto de dados de câncer de mama pode ser encontrado em [11], e consiste em uma série de imagens computalizadas de exames de câncer de mama do tipo biópsia por aspiração agulha fina (PAAF), como mostrado na figura 21. Esse conjunto de dados apresenta 30 características que descrevem o núcleo celular presente na imagem, entre elas, perímetro, raio, concavidade, área, entre outras métricas que descrevem o núcleo celular. Além disso, o conjunto de dados tem

duas classes possíveis para a variável resposta, "B", para benigno, e "M", para maligno, e são essas duas classes que vamos classificar no nosso modelo. A figura 22 mostra o resultado do processamento quântico para a classificação de câncer de mama.

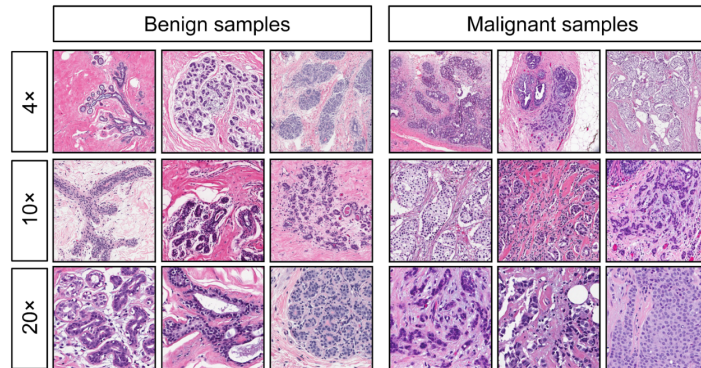


Figura 21: Exemplo de imagens de biópsia por aspiração agulha fina usadas na construção do conjunto de dados. Imagem retirada de [13]

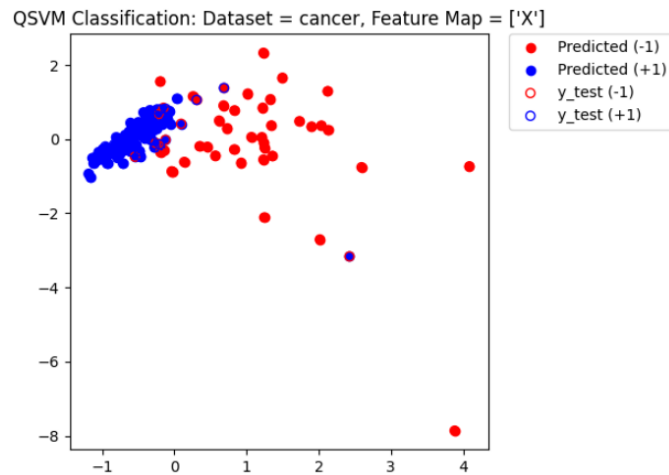


Figura 22: Diagrama de classificação - QSVM com conjunto de dados Scikit-Learn de reconhecimento de câncer de mama reduzido com PCA, com mapa de características Pauli X

### 6.2.1.2 CONJUNTO DE DADOS 2 - VINHOS

O conjunto de dados de identificação de vinhos pode ser encontrado em [12], e consiste em um registro de uma análise química de vinhos de uma mesma região da Itália mas de cultivares diferentes. Esse conjunto de dados apresenta 13 características que descrevem algumas propriedades do vinho, como quantidade de álcool, quantidade de magnésio, intensidade da cor, entre outras. Além disso, o conjunto de dados tem três classes possíveis para a variável resposta, 1, 2 ou 3, que corresponde ao índice de cada um dos tipos de vinhos. A figura 23 mostra o resultado do processamento quântico para a classificação de 3 tipos de vinhos.

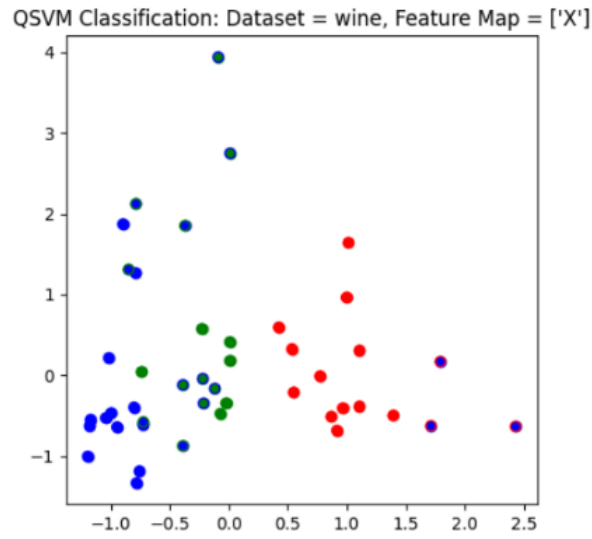


Figura 23: Diagrama de classificação - QSVM com conjunto de dados Scikit-Learn de reconhecimento de tipos de vinho reduzido com PCA, com mapa de características Pauli X

### 6.2.1.3 CONJUNTO DE DADOS 3 - ÍRIS

O conjunto de dados de identificação de flores do tipo Íris de Fisher pode ser encontrado em [14], e consiste em 50 amostras de cada uma das três espécies de Íris, Íris setosa, Íris virginica e Íris versicolor, mostradas na figura 24, e descritas por quatro características, comprimento das sépalas, largura das sépalas, comprimento das pétalas e largura das pétalas, em centímetros. A figura 25 mostra o resultado do processamento quântico para a classificação de 3 tipos de flores íris.



Figura 24: Três tipos de Íris: Setosa, Versicolor e Virginica

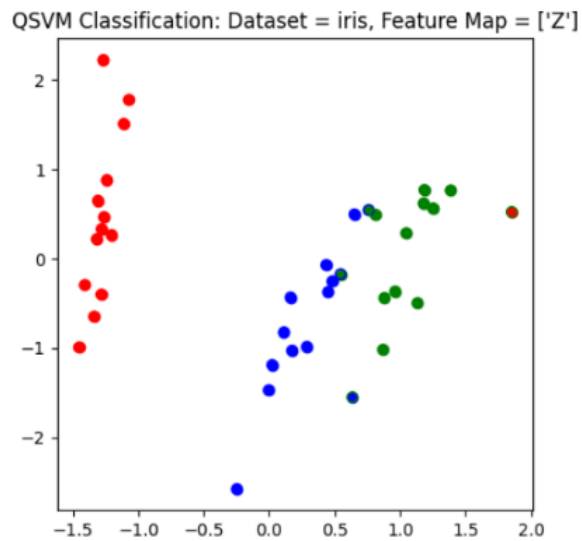


Figura 25: Diagrama de classificação - QSVM com conjunto de dados Scikit-Learn de reconhecimento de flores íris reduzido com PCA, com mapa de características Pauli Z.

## 6.2.2 COMPARAÇÃO DE RESULTADOS

Fazendo uma análise comparativa do desempenho dos algoritmos SVM e QSVM para os três conjuntos de dados, variando também o mapa de características da implementação do QSVM, obtivemos a tabela 1. Vale salientar que, por simplicidade, utilizamos a técnica PCA para diminuir a dimensionalidade dos vetores de características, fazendo essa análise com apenas duas características inicialmente.

Podemos perceber que para os três conjuntos de dados escolhidos, a implementação clássica SVM obteve uma precisão maior. Considerando apenas as implementações de QSVM, obtivemos melhor precisão, para os conjuntos de dados de câncer de mama e de vinho, com o mapa Pauli X, já para o conjunto de dados de íris, o mapa Pauli Z obteve uma precisão maior. Podemos notar também que apesar da maior precisão com SVM clássico, as implementações quânticas obtiveram um resultado muito próximo.

É possível demonstrar que, quando se trata de QSVM, devemos escolher o mapa de características certo para o conjunto de dados certo. Isso faz com que possamos mapear nossa função Kernel da melhor maneira para representar aqueles dados.

As figuras 22, 23 e 25 são diagramas que mostram a classificação feita para os três conjuntos de dados com o mapa de características mais adequado para aquele conjunto de dados, Pauli X para os conjuntos de dados de câncer de mama e de vinho, e Pauli Z para o conjunto de dados de íris.

2 Características				
Modelo	Mapa	Câncer	Vinho	Íris
SVM	RBF	0.993	0.733	0.921
QSVM	Pauli X	<b>0.958</b>	<b>0.688</b>	0.842
	Pauli Y	0.622	0.400	0.315
	Pauli Z	0.783	0.600	<b>0.894</b>
	Pauli XX	0.643	0.422	0.368
	Pauli YY	0.566	0.488	0.342
	Pauli ZZ	0.601	0.333	0.236

Tabela 1: Tabela com comparação da precisão das previsões dos algoritmos SVM e QSVM para diferentes conjuntos de dados e diferentes mapas de características

Fazendo uma segunda análise, mostrada na tabela 2, escolhemos o conjunto de dados de câncer de mama, e comparamos a precisão do SVM e do QSVM com Pauli X, variando o número de características. É interessante notar que, por se tratar de um conjunto de dados muito simples, não houve um aumento da precisão com o aumento do número de características. Tanto para o SVM, quanto para o QSVM, a maior precisão foi obtida com 2 características.

Câncer de mama		
Número de características/qubits	QSVM - Pauli X	SVM - RBF
2	0.958	0.993
4	0.888	0.958
8	0.853	0.979
10	0.895	0.930
12	0.860	0.937

Tabela 2: Tabela com comparação da precisão das previsões dos algoritmos SVM e QSVM Pauli X, variando o número de características

## 7 CONCLUSÃO

Ao longo desse projeto foi possível fazer um breve estudo da história da mecânica quântica e da computação quântica. Foi possível também, fazer uma descrição do espaço abstrato onde os fenômenos quânticos acontecem, além de fazer um estudo sobre o formalismo matemático que nos permite descrever os sistemas quânticos no espaço de Hilbert. Foi possível descrever os postulados que dão base a mecânica quântica e como esses postulados estão relacionados com a computação quântica. Além disso, conseguimos introduzir os principais conceitos e elementos da computação quântica, como o qubits, a esfera de bloch e as principais portas lógicas quânticas. Finalmente, foi possível descrever, com grande grau de detalhamento, dois importantes algoritmos quânticos, além de utilizá-los em problemas reais, implementando e executando esses algoritmos em simuladores de computadores quânticos, e fazendo uma análise de seu desempenho. As técnicas estudadas demonstraram que a computação quântica pode ser muito útil para resolver problemas do mundo real. Porém, ainda está em um estágio de desenvolvimento inicial e há muito a ser estudado na área, e uma supremacia da computação quântica sobre a computação clássica, só poderá ser obtida com o desenvolvimento e aperfeiçoamento dos algoritmos quânticos, além de melhorias na implementação dos hardwares. Os estudos desenvolvidos ao longo desse trabalho demonstram como essas técnicas podem ser empregadas na resolução de problemas reais. Ainda que esses problemas possam ser resolvidos de maneira mais eficiente usando computadores clássicos, esse estudo contribui para identificar algumas lacunas que ainda precisam ser melhor desenvolvidas na computação quântica.



## 8 REFERÊNCIAS

[1] JACK D HIDARY. **Quantum computing: an applied approach, volume 1**. Springer, 2021

[2] MICROSOFT. **Microsoft Quantum Concepts. Quantum computing history and background**. Disponível em <<https://learn.microsoft.com/en-us/azure/quantum/concepts-overview>>. Acesso em 17 de Maio de 2023.

[3] FERREIRA, RODRIGO. **Algoritmo de Shor**. Disponível em <<https://brazilquantum.medium.com/algoritmo-de-shor-bcf37e92924b>>. Acesso em 17 de Maio de 2023.

[4] AMAZON. **Amazon AWS. O que é computação quântica?**. Disponível em <<https://aws.amazon.com/pt/what-is/quantum-computing>>. Acesso em 21 de Maio de 2023.

[5] L. K. GROVER, **Um algoritmo mecânico quântico rápido para pesquisa em banco de dados**. Procedimentos do 28° Simpósio Anual em Teoria da Computação (STOC) 1996. <https://dl.acm.org/doi/10.1145/237814.237866>

[6] R. L. Jaffe, **SUPPLEMENTARY NOTES ON DIRAC NOTATION, QUANTUM STATES, ETC**. Massachusetts Institute of Technology 1996. Disponível em <<https://web.mit.edu/8.05/handouts/jaffe1.pdf>>. Acesso em 15 de Setembro de 2023.

[7] DAVID J. GRIFFITHS. **Mecânica Quântica, edição 2**. Pearson, 2011.

[8] J.J SAKURAI E J. NAPOLITANO. **Mecânica Quântica Moderna, segunda edição** . Bookman, 2013.

[9] PARK, JAE-EUN; QUANZ, BRIAN; WOOD, STEVE; HIGGINS, HEATHER; HARISHANKAR, RAY. **Practical application improvement to Quantum SVM: theory to practice**. IBM Global Business Services - Quantum Consulting Center of Competence, IBM Research 2020

[10] HAVLICEK, VOJTECH; D. CÓRCOLES, ANTONIO; TEMME, KRISTAN; W. HARROW, ARAM; KANDALA, ABHINAV; M. CHOW, JERRY; M. GAMBETTA, JAY. **Supervised learning with quantum enhanced feature spaces**. Centro de pesquisa IBM T.J. Watson e Centro de física teórica do Instituto de Tecnologia de Massachusetts.

[11] LICHMAN M. **UCI Machine Learning Repository: Breast Cancer Wisconsin (Diagnostic) Data Set**. 2014. Disponível em <<http://archive.ics.uci.edu/ml>>. Acesso 7 de Outubro de 2023.

[12] M. FORINA; AEBERHARD, STEFAN. **Machine Learning Repository: Wine Data Set**. 1991. Disponível em <<https://archive.ics.uci.edu/dataset/109/wine>>. Acesso 12 de Novembro de 2023.

[13] SUMBRIA, SHUBAM. **Breast Cancer Wisconsin [Diagnostic] - EDA**. 2021. Disponível em <<https://medium.com/analytics-vidhya/breast-cancer-diagnostic-dataset-eda-fa0de80f15bd>>. Acesso 7 de Dezembro de 2023.

[14] R. A. FISHER. **Machine Learning Repository: Iris Data Set**. 1988. Disponível em <<https://archive.ics.uci.edu/dataset/53/iris>>. Acesso 8 de Dezembro de 2023.

[15] K. APPEL, W. HAKEN, J. Koch. **Every planar map is four colorable. Part 2: reducibility**. 1977. Boletim da Sociedade Americana de Matemática.

[16] GOOGLE. **O que é o Colaboratory?**. Disponível em <<https://research.google.com/colaboratory/intl/pt-BR/faq.html>>. Acesso em 11 de Dezembro de 2023.

[17] JOLLIFFE, IAN T.; CADIMA, JORGE. **Principal component analysis: a review and recent developments**. 2016. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences.