Rodrigo da Silva Sousa

MEMÓRIAS QUÂNTICAS PROBABILÍSTICAS EM COMPUTADORES QUÂNTICOS RUIDOSOS DE PEQUENA-ESCALA

Dissertação de Mestrado

Universidade Federal Rural de Pernambuco
secretaria@preg.ufrpe.br
http://www.ufrpe.br/br/graduacao

RECIFE
2019

Universidade Federal Rural de Pernambuco
Departamento de Estatística e Informática
Programa de Pós Graduação em Informática Aplicada

Rodrigo da Silva Sousa

MEMÓRIAS QUÂNTICAS PROBABILÍSTICAS EM COMPUTADORES QUÂNTICOS RUIDOSOS DE PEQUENA-ESCALA

Dissertação de Mestrado apresentada ao Programa de Pós Graduação em Informática Aplicada do Departamento de Estatística e Informática da Universidade Federal Rural de Pernambuco como requisito parcial para obtenção do grau de Mestre em Informática Aplicada.

Orientador: *Adenilton José da Silva*

RECIFE
2019

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema Integrado de Bibliotecas da UFRPE
Biblioteca Central, Recife-PE, Brasil

S725m Sousa, Rodrigo da Silva
Memórias quânticas probabilísticas em computadores quânticos ruidosos de pequena-escala / Rodrigo da Silva Sousa. – 2019.
59 f. : il.

Orientador: Adenilton José da Silva.
Dissertação (Mestrado) – Universidade Federal Rural de Pernambuco, Programa de Pós-Graduação em Informática Aplicada, Recife, BR-PE, 2019.
Inclui referências.

1. Computação quântica 2. Aprendizado do computador 3. Computadores quânticos I. Silva, Adenilton José da, orient. II. Título

CDD 004

Dissertação de Mestrado apresentada por **Rodrigo da Silva Sousa** ao Programa de Pós Graduação em Informática Aplicada do Departamento de Estatística e Informática da Universidade Federal Rural de Pernambuco sob o título **Memórias quânticas probabilísticas em computadores quânticos ruidosos de pequena-escala**, orientada pelo **Prof. Adenilton José da Silva** e aprovada em 12/08/2019 pela banca examinadora formada pelos professores:

Prof. Adenilton José da Silva
Centro de Informática/UFPE

Prof. Wilson Rosa de Oliveira Junior
Departamento de Estatística e Informática/UFRPE

Prof. Fernando Maciano de Paula Neto
Centro de Informática/UFPE

RECIFE
2019

Agradecimentos

Agradeço aos meus pais, Maria e Arlindo, por todo o amor, apoio, e pela educação que conseguiram me proporcionar com muito esforço.

A minha querida namorada Priscila, por todos os momentos que passamos juntos e por sempre ser minha inspiração.

Agradeço em especial ao professor Adenilton Silva, que me orientou neste trabalho, por toda a dedicação, ensinamentos e paciência.

Agradeço também a todos os professores do Departamento de Estatística e Informática e Departamento de Computação da Universidade Federal Rural de Pernambuco e colegas do grupo de computação quântica, por todas as importantes colaborações para a minha formação acadêmica.

Por fim agradeço ao instituto Serrapilheila pelo suporte financeiro durante o desenvolvimento deste e outros trabalhos.

L'enfer, c'est les autres

—JEAN-PAUL SARTRE

Resumo

A capacidade de armazenar informação é essencial para qualquer dispositivo computacional. Uma memória quântica é a realização quântica de um dispositivo de armazenamento e recuperação de dados. A memória probabilística quântica é um modelo de memória associativa que armazena dados binários em uma superposição quântica e recupera as informações a partir do cálculo da distância de Hamming entre o padrão de entrada e os demais padrões binários armazenados na memória.

Neste trabalho, uma avaliação experimental da memória probabilística quântica é realizada em um dispositivo quântico ruidoso que dispõe de apenas 5 qubits. Computadores quânticos universais ainda não são uma realidade e os dispositivos quânticos disponíveis atualmente são ruidosos e possuem uma quantidade pequena de qubits, além de ter uma arquitetura limitada. Dessa forma, para realizar a execução de uma memória quântica em um desses dispositivos, uma implementação híbrida clássico-quântica foi proposta neste trabalho na forma de um protocolo otimizado que reduz o número de qubits e operações necessárias para a execução da memória quântica em dispositivos quânticos de pequena escala. Através de uma avaliação experimental foi verificado que a implementação proposta apresentou o funcionamento esperado da memória quântica.

Palavras-chave: computação quântica, memória probabilística quântica, computadores quânticos de escala intermediária, dispositivos quânticos

Abstract

The ability to store information is essential for any computing device. A quantum memory is the quantum realization of a data storage and retrieval device. A probabilistic quantum memory is an associative memory model which stores binary data on a quantum superposition and retrieves the information by calculating the Hamming distance between the input pattern and any other binary patterns stored on the memory.

In this work, an experimental evaluation of the probabilistic quantum memory is performed on a noisy quantum device that has only 5 qubits. Universal quantum computers are not yet a reality and the quantum devices that are available at present are noisy and have a small amount of qubits, in addition to having a limited architecture. In this way, in order to perform the implementation of a quantum memory in one of these devices, a hybrid classical-quantum implementation was proposed in this work in the form of a optimized protocol which reduces the number of qubits and quantum operations necessary for the execution of the quantum memory on small-scale quantum devices. Through an experimental evaluation it was verified that the proposed implementation presented the expected functioning of the quantum memory.

Keywords: quantum computing, probabilistic quantum memory, near intermediate-scale quantum computer, quantum devices

Lista de Figuras

Lista de Tabelas

Lista de Pseudocódigos

Lista de Acrônimos

MQP	Memória Quântica Probabilística
PQM	Probabilistic Quantum memory
CQ	Computação Quântica
EPR	Einstein-Podolsky-Rose
NISQ	Noisy Intermediate-Scale Quantum

Sumário

1

Introdução

Este trabalho apresenta a realização de uma memória associativa quântica em um computador quântico de pequena escala. A Computação Quântica (CQ) é uma área que incorpora à computação clássica as mecânicas e fenômenos que se manifestam em escala atômica (NIELSEN; CHUANG, 2002); tais fenômenos, como a superposição de estados e o paralelismo quântico, são muito importantes e muito utilizados no contexto da computação quântica; embora sejam de difícil compreensão, pois não são considerados intuitivos a partir da percepção clássica do mundo físico.

A utilização dos princípios da mecânica quântica como paradigma computacional foi impulsionada pelo seu potencial de ser aplicada em diversas áreas da computação e comunicações. O processamento e armazenamento de dados é parte essencial da computação clássica, da mesma maneira, soluções quânticas eficientes para o armazenamento e processamento de dados tem igual importância dentro do contexto computacional em geral.

Na computação quântica, dois princípios fundamentais e amplamente utilizados são a superposição de estados base e o emaranhamento quântico. A superposição de estados, ou superposição quântica, descreve a possibilidade de partículas assumirem mais de um estado quântico ao mesmo tempo. Para se determinar em que estado uma partícula se encontrará é necessário realizar uma medição. Após a medição, uma partícula em superposição assume apenas um dos estados. Os estados base possíveis para uma partícula são agrupados por amplitudes de probabilidades associadas aos mesmos. O emaranhamento quântico, por sua vez, consiste no fenômeno em que duas ou mais partículas podem ser descritas por um estado conjunto, onde uma não pode ser descrita separadamente das outras que fazem parte do emaranhamento. O emaranhamento quântico tem aplicações em diversos algoritmos. Um exemplo famoso é o algoritmo de teleporte quântico proposto por BENNETT et al. (1993).

Os primeiros estudos que tratam de computação quântica foram publicados na década de 80 por BENIOFF (1980) e FEYNMAN (1982). Na computação quântica o qubit é usado como o equivalente quântico do bit utilizado na computação clássica. Um bit clássico pode assumir apenas um valor em dado instante, 0 ou 1; enquanto um qubit, ainda sendo a unidade básica de representação de informação no contexto quântico, além de assumir 0 ou 1, também

pode ser formado como uma combinação de ambos os valores ao mesmo tempo, graças ao fenômeno conhecido como superposição quântica. Dessa forma, o qubit pode ser descrito como um sistema linear de amplitudes associadas aos diferentes possíveis estados base, onde o módulo ao quadrado das amplitudes determina a probabilidade em que o sistema pode ser medido em cada estado possível.

Uma Memória Quântica Probabilística (MQP) (TRUGENBERGER, 2001), Probabilistic Quantum memory (PQM) em inglês, é um modelo de memória associativa no contexto da computação quântica. A PQM é uma memória associativa quântica capaz de armazenar cadeias binárias em qubits e acessar o seu conteúdo a partir de uma dada entrada, ainda que esta entrada seja apresentada de forma corrompida ou incompleta.

Diferente das memórias tradicionais que usam tabelas para endereçar e acessar o seu conteúdo, uma memória associativa é capaz de retornar os dados armazenados por endereçamento pelo conteúdo. Essas memórias realizam associações da entrada fornecida com os dados armazenados. Dessa maneira, é possível recuperar informação sem ser necessário conhecimento prévio do conteúdo da memória, podendo-se usar até mesmo entradas corrompidas ou incompletas.

As memórias associativas quânticas fornecem algumas vantagens em comparação com as memórias associativas clássicas. Em uma memória associativa quântica é possível armazenar a informação em um estado quântico formado pela superposição uniforme dos dados na memória, o que permite efetivamente armazenar uma quantidade exponencial de dados. Assim, a capacidade de armazenamento da memória quântica é uma grande vantagem sobre as memórias associativas clássicas. Seguindo a mesma ideia, as operações necessárias de busca na memória podem ser realizadas em todos os estados da superposição ao mesmo tempo.

A computação quântica avançou rapidamente desde a sua concepção, no entanto, apenas recentemente começaram a aparecer os primeiros computadores quânticos. Os computadores quânticos atuais ainda são bastante limitados na quantidade e qualidade dos qubits que os compõe, além de apresentarem ruído considerável. Os computadores quânticos existentes atualmente possuem de 5 a 20 qubits e taxas de erro na ordem de 1.89×10^{-3} na aplicação de operadores quânticos e 14.2×10^{-2} na medição dos qubits. O ruído e a descoerência, o processo de perda de informação do sistema quântico para o ambiente, são dois aspectos inerentes aos dispositivos atuais e que constituem uma barreira importante que ainda precisa ser transposta para permitir a utilização dos computadores quânticos em escala prática (PRESKILL, 2012).

Dessa forma, este trabalho utilizou um dos dispositivos quânticos disponibilizados pela plataforma Q Experience[1] da IBM. O computador onde foram executados os experimentos possui 5 qubits e é referenciado pelo codinome "Tenerife" (TEAM, 2018). Os dispositivos disponíveis na plataforma podem ser acessados por meio do kit de desenvolvimento Qiskit[2], o qual contém API própria para a implementação e execução de circuitos quânticos.

[1] https://quantumexperience.ng.bluemix.net/qx/experience
[2] https://qiskit.org/

1.1 Motivação

A computação quântica é um campo de pesquisa relativamente recente, o qual foi primeiramente conjecturado por BENIOFF (1980) e FEYNMAN (1982). Desde então, diversas pesquisas foram conduzidas na área, alcançando descobertas e avanços como a teoria quântica formalizada em DEUTSCH (1989) e os algoritmos de Shor (SHOR, 1994, 1999) e Grover (GROVER, 1996). Atualmente, estão sendo realizados grandes investimentos e pesquisas em áreas relacionadas com a computação quântica, principalmente em campos que envolvem o desenvolvimento e utilização de computadores quânticos.

As maiores potências econômicas mundiais estão montando esforços com o objetivo de se adiantarem na corrida quântica. Algumas empresas, como a IBM e Rigetti, já conseguiram resultados relevantes na construção de dispositivos quânticos. Nos últimos anos a IBM construiu uma variedade de computadores com 5 (TEAM, 2018) a 20 qubits; enquanto a Rigetti desenvolveu computadores com 8 (REAGOR et al., 2018) e 20 qubits (OTTERBACH et al., 2017). Outros dispositivos, com uma quantidade ainda maior de qubits, estão sendo desenvolvidos pela IBM (KNIGHT, 2017), Intel e Google (KELLY, 2018). Apesar dos resultados ainda se mostrarem modestos em um primeiro momento, principalmente devido ao número muito limitado de qubits, as perspectivas seguem otimistas e os avanços obtidos até o momento confirmam o grande potencial da área (BRAVYI; GOSSET; KOENIG, 2018).

Algoritmos quânticos existem desde o início das pesquisas em computação quântica, como os algoritmos descritos em SHOR (1994) e GROVER (1996). Por outro lado, o computador quântico universal ainda não se tornou uma realidade. A tecnologia se encontra no patamar dos computadores quânticos ruidosos de escala intermediária, conhecidos pela sigla NISQ de Noisy Intermediate-Scale Quantum (NISQ) (PRESKILL, 2018). Um dispositivo NISQ é caracterizado pela pequena quantidade de qubits, entre 50 e 100, bem como pelo ruído considerável na execução de medições e operações quânticas. Essas limitações impossibilitam a execução prática da maioria dos algoritmos quânticos existentes.

No entanto, a pesquisa em tecnologia quântica não deixa de avançar. Com a promessa de dispositivos quânticos com um maior número de qubits à disposição, muito espera-se conseguir demonstrações da chamada supremacia-quântica. O termo supremacia-quântica é usado para denominar a capacidade de um computador quântico resolver eficientemente problemas que não podem ser resolvidos pela computação clássica. Espera-se que, para os próximos anos, os computadores quânticos ultrapassem a barreira dos 50 qubits, podendo chegar até a 100 qubits.

Enquanto computadores quânticos com maior capacidade ainda não são uma realidade, os computadores de pequena escala já podem ser utilizados para realizar experimentos com algoritmos quânticos, apesar dos desafios (SILVA; OLIVEIRA; LUDERMIR, 2010; SCHULD; SINAYSKIY; PETRUCCIONE, 2014a; SANTOS et al., 2018a,b). O que se observa no geral é a necessidade de modificações e otimizações nos algoritmos com o intuito de contornar as limitações dos computadores quânticos atuais. Nesse contexto, este trabalho se motiva em

investigar e avaliar a possibilidade da realização direta de uma memória probabilística em um computador quântico de pequena escala com 5 qubits.

1.2 Objetivos

1.2.1 Objetivo geral

Desenvolver um método híbrido para execução de uma memória probabilística quântica em um computador quântico de pequena escala

1.2.2 Objetivos específicos

- Analisar a viabilidade de realização dos algoritmos da PQM em um computador quântico real de pequena escala;
- Desenvolver um protocolo para otimização da PQM com o objetivo de diminuir os recursos utilizados e operações necessárias;
- Verificar o desempenho do modelo a partir de testes comparativos no dispositivo quântico;

1.3 Contribuições obtidas

Nesta seção serão sintetizadas as principais contribuições obtidas neste trabalho. Dado o fato que os computadores quânticos disponíveis atualmente possuem uma quantidade limitada de qubits, é muito importante para o contexto atual da área que sejam construídas aplicações capazes de serem executadas nestes computadores observando suas limitações (PRESKILL, 2018). Dessa forma, se torna possível investigar e avaliar o funcionamento dos algoritmos quânticos diretamente no contexto em que se propõe e também avaliar o próprio funcionamento dos computadores quânticos de pequena escala. Neste trabalho, uma Memória Probabilística Quântica é implementada e avaliada em um computador quântico real de 5-qubits.

A primeira contribuição deste trabalho é a análise da viabilidade de realização da memória quântica em um computador quântico de pequena escala. São levadas em consideração as limitações na quantidade de qubits, bem como a arquitetura e topologia do dispositivo utilizado. Os algoritmos que descrevem o funcionamento da memória são analisados e é discutido em detalhes a possibilidade de implementação direta da memória no dispositivo quântico.

Diante disso, outra contribuição deste trabalho é a otimização da memória de forma a permitir sua execução no dispositivo quântico de pequena escala. Para tal, foi proposto um procedimento híbrido clássico-quântico, o qual adapta e otimiza os algoritmos para fazer melhor uso da estrutura clássico-quântica fornecida pela plataforma quântica considerada neste trabalho.

Com isso foi mostrado ser possível executar a memória quântica em computadores quânticos ruidosos de pequena escala. Esta foi a primeira execução da PQM já realizada em um computador quântico real com experimentos para avaliar seu funcionamento.

1.4 Organização do trabalho

Este trabalho de dissertação está organizado da seguinte forma:

No Capítulo 2 são introduzidos os principais conceitos da computação quântica, necessários para o entendimento do estudo aqui apresentado. É apresentado o conceito do qubit, a menor unidade de informação quântica. São apresentadas as principais portas quânticas usadas para construir os circuitos quânticos. Além disso, é visto o conceito de paralelismo quântico e a definição do algoritmo de Grover.

O Capítulo 3 é dedicado as memórias associativas quânticas, em especial à PQM, o principal objeto de estudo deste trabalho. Neste capítulo é primeiro exposto o conceito de memória quântica e as propostas existentes na literatura. A PQM é bastante detalhada em várias seções onde são discutidas as ideias, aplicações, vantagens e desvantagens e principalmente o seu funcionamento.

O Capítulo 4 apresenta a plataforma usada para realizar a execução dos experimentos e exemplos de implementação de circuitos quânticos dentro da plataforma. Além disso, são discutidas as principais características dos dispositivos quânticos de pequena escala e as suas implicações para a execução dos circuitos.

No Capítulo 5 são expostas as contribuições centrais do trabalho: a análise de viabilidade e o método proposto para a adaptação e otimização da PQM. No mesmo capítulo são apresentados os experimentos que foram realizados e a discussão dos resultados obtidos. Por fim, o Capítulo 6 contém as conclusões finais e trabalhos futuros.

2

Computação Quântica

Este trabalho versa sobre a realização de uma memória associativa quântica em um computador quântico real, deste modo se faz necessário a apresentação de alguns conceitos fundamentais de computação quântica para proporcionar um melhor entendimento do trabalho.

2.1 Computação Quântica

A computação quântica é uma forma de fazer computação baseada em mecânicas e princípios da teoria quântica. Dessa forma, em oposição aos computadores clássicos, ou computadores tradicionais, um computador quântico se utiliza de efeitos de nível microscópico para realizar tarefas computacionais. A versão quântica do bit, ou seja, a menor unidade de informação quântica, é denominada qubit (*quantum bit* ou bit quântico). O qubit não assume apenas um de dois valores, 0 ou 1, em um dado instante, como acontece com o bit clássico. Um qubit pode assumir os dois valores ao mesmo tempo, em uma superposição de 0 e 1. Esse estado é descrito como uma combinação linear dos valores, da seguinte forma:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,,$$

onde $|\alpha|^2 + |\beta|^2 = 1$, sendo α e β os valores das amplitudes de probabilidade associadas aos estados $|0\rangle$ e $|1\rangle$, respectivamente.

As amplitudes α e β são representadas por números complexos, assim, um qubit é um vetor em um espaço vetorial complexo bidimensional. Os estados $|0\rangle$ e $|1\rangle$ são chamados de base computacional, eles constituem uma base ortonormal para esse espaço vetorial. Um qubit pode ser descrito na forma matricial:

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

O qubit também pode ser descrito através de uma representação geométrica:

$$|\psi\rangle = cos\frac{\theta}{2}\,|0\rangle + e^{i\phi} sen\frac{\theta}{2}\,|1\rangle$$

Os valores θ e ϕ indicam um ponto na esfera tridimensional que é chamada de esfera de Bloch, representada na Figura 2.1.

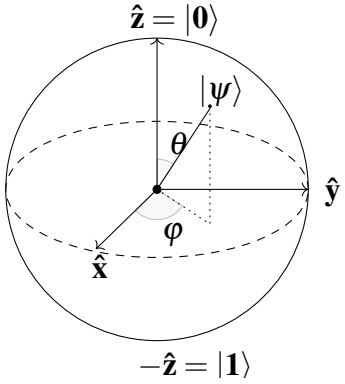


Figura 2.1: Esfera de Bloch

Em síntese, $|\psi\rangle$ pode estar em uma superposição de ambos os estados 0 e 1 simultaneamente e as amplitudes indicam a probabilidade do qubit assumir um dos estados após ser realizada uma medição. A medição causa o colapso do qubit para apenas um dos valores possíveis. Uma vez que o qubit foi colapsado não é mais possível determinar o seu estado quântico antes da medição.

Os estados da base computacional para dois qubits formam $|00\rangle, |01\rangle, |10\rangle, |11\rangle$, análogos aos quatro estados que podem ser formados com dois bits clássicos. Porém, o estado composto por dois bits quânticos pode estar em uma combinação linear dos estados base. Seja $|\psi_2\rangle$ um estado quântico com dois qubits, ele pode ser descrito como:

$$|\psi_2\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle.$$

Da mesma forma como ocorreu com o estado base formado por apenas um qubit, aqui o resultado da medição dos dois qubits de $|\psi_2\rangle$ resulta em um dos quatro estados base com a probabilidade indicada pelo módulo ao quadrado das amplitudes, que somadas resultam em 1:

$$\sum_{x\in\{0,1\}^2} |\alpha_x|^2 = 1$$

É possível também medir apenas parte dos qubits que compõe um sistema quântico de múltiplos qubits. Realizando a medição apenas no primeiro qubit de $|\psi_2\rangle$, a probabilidade de obter 0 é $|\alpha_{00}|^2 + |\alpha_{01}|^2$ e a de se obter 1 é $|\alpha_{10}|^2 + |\alpha_{11}|^2$. O estado após a medição colapsa para:

$$|\psi_2'\rangle = \frac{\alpha_{00}|00\rangle + \alpha_{01}|01\rangle}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}$$

O estado obtido após a medição constitui também um estado quântico normalizado, cujo módulo ao quadrado das amplitudes somadas é igual a 1. É possível observar que a normalização

se dá pelo fator $\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}$.

Os estados quânticos formados por mais de um qubit podem ser compostos pelo produto tensorial $\otimes$ de qubits. Dessa forma, tem-se:

$$|0\rangle \otimes |0\rangle = |00\rangle$$

$$|0\rangle \otimes |1\rangle = |01\rangle$$

$$|1\rangle \otimes |0\rangle = |10\rangle$$

$$|1\rangle \otimes |0\rangle = |11\rangle$$

Na forma matricial:

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}; |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}; |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}; |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Da mesma forma, considere os qubits descritos por $|\psi_0\rangle = \alpha_0 |0\rangle + \beta_0 |1\rangle$ e $|\psi_1\rangle = \alpha_1 |0\rangle + \beta_1 |1\rangle$, o sistema quântico formado por eles é representado por $|\psi_0 \psi_1\rangle = \alpha_0\alpha_1 |00\rangle + \alpha_0\beta_1 |01\rangle + \beta_0\alpha_1 |10\rangle + \beta_0\beta_1 |11\rangle$.

Existem, contudo, estados quânticos que não podem ser decompostos em produtos tensoriais, estes estados recebem o nome de estados emaranhados. Um importante exemplo de um estado emaranhado é o par Einstein-Podolsky-Rose (EPR), também chamado de estado de Bell:

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

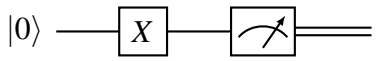
Esse estado é importante pois a sua característica de não-separabilidade EPR é a base do conceito de teletransporte quântico (BENNETT et al., 1993) e da codificação superdensa (BENNETT; WIESNER, 1992).

2.2 Portas Quânticas

A manipulação de informação na computação clássica é representada através de portas lógicas que implementam uma lógica booleana que possibilita operações sobre os bits. Na computação quântica, as operações realizadas sobre qubits ou sistemas quânticos se dão através de circuitos quânticos compostos por portas quânticas. Tais portas alteram o sistema quântico a partir de transformações unitárias.

Nesta seção as principais portas quânticas serão apresentadas tanto nas suas respectivas representações matriciais quanto em forma de circuito. A notação de circuito é comumente usada

para representar um sistema quântico composto por uma ou mais operações unitárias. Cada linha que cruza o circuito horizontalmente representa um qubit que compõe o sistema quântico enquanto caixas portando algum simbolo são usadas para indicar as operações realizadas no circuito. Além das operações unitárias também é possível representar operações de medição em circuitos quânticos, no exemplo a seguir temos um circuito que demonstra um qubit inicializado no estado $|0\rangle$ e a aplicação de uma porta X seguida pela medição do qubit:



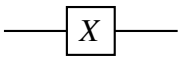
A porta X, utilizada no exemplo acima, bem como outras portas unitárias são descritas nas subseções seguintes.

2.2.1 Portas de Pauli

As portas de Pauli são portas quânticas que agem sobre estados quânticos de 1 qubit. As matrizes descrevendo cada uma das portas são apresentadas a seguir:

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

A Porta X, também conhecida como NOT é o equivalente quântico da porta clássica de mesmo nome. Essa porta realiza uma rotação de 180 graus no eixo x da esfera de Bloch. A porta NOT é representada por:

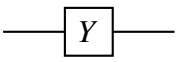


Essa porta inverte os estados da seguinte forma:

$$X|0\rangle = |1\rangle$$

$$X|1\rangle = |0\rangle$$

A porta Y equivale a uma rotação de 180 graus no eixo y da esfera de Bloch. É representada por:



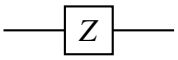
Sua atuação é representada por:

$$Y|0\rangle = i|1\rangle$$

$$Y|1\rangle = -i|0\rangle.$$

A porta Z, ou porta de mudança de fase, equivale a uma rotação de 180 graus no eixo z

da esfera de Bloch. É representada por:



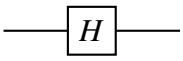
Sua atuação é representada por:

$$Z|0\rangle = |0\rangle$$

$$Z|1\rangle = -|1\rangle.$$

2.2.2 Hadamard

Uma porta de extrema importância é a porta chamada Hadamard. É representada por



Essa porta age sobre um qubit da seguinte forma:

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

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

Dessa maneira, a porta Hadamard é capaz de criar uma superposição de estados. Um qubit que estava no estado $|0\rangle$ é colocado em superposição com 50% de chances de estar no estado $|0\rangle$ ou $|1\rangle$.

2.2.3 Portas Controladas

As portas controladas realizam uma operação condicional em um qubit alvo dependendo do estado de um ou mais qubits de controle. Essas portas atuam em estados quânticos compostos por ao menos 2 qubits. A porta *CNOT* (Não-Controlado) é a versão controlada da porta *NOT* (Pauli X). Ela atua rotacionando o estado do qubit alvo apenas se o qubit de controle se encontra no estado $|1\rangle$. O circuito da porta *CNOT* pode ser visto na Figura 2.2

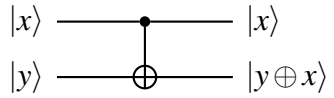


Figura 2.2: Operador CNOT

Assim, a porta X será aplicada sobre o qubit alvo $|y\rangle$ caso o qubit de controle $|x\rangle = 1$. Esse operador é representado pela matriz:

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

Dessa mesma maneira é possível fazer operações controladas usando outras portas quânticas, como pode ser observado na Figura 2.3.

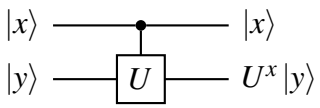


Figura 2.3: Operação U-Controlada

Onde U é uma porta quântica unitária qualquer que atua em um qubit e é representada pelos valores x_{00}, x_{10}, x_{01} e x_{11}. A notação $U^x|y\rangle$ no circuito da Figura 2.3 descreve o estado resultante de $|y\rangle$ considerando que a operação CU aplicará U em $|y\rangle$ apenas quando $|x\rangle = 1$ e, portanto, o estado resultante será $U^1|y\rangle$. Quando $|x\rangle = 0$ temos $U^0|y\rangle$ ou $I|y\rangle$ e $|y\rangle$ não é alterado.

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

A representação matricial de CU será, portanto:

$$CU = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & x_{00} & x_{10} \\ 0 & 0 & x_{01} & x_{11} \end{bmatrix}$$

2.2.4 Porta Toffoli

As operaçãos controladas também podem ser formadas a partir de múltiplos qubits de controle. A porta Toffoli é um exemplo, ela atua em três qubits sendo dois deles qubits de controle. Como pode ser visto na Figura 2.4, o estado do qubit alvo $|z\rangle$ é alterado com a porta X apenas se ambos os qubits de controle $|x\rangle$ e $|y\rangle$ forem iguais a $|1\rangle$. A partir da porta Toffoli é possível simular qualquer circuito clássico (TOFFOLI, 1980), mostrando que a computação quântica possui poder computacional no minimo equiparável ao dos computadores clássicos determinísticos.

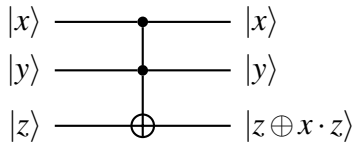


Figura 2.4: Porta Toffoli

2.3 Paralelismo Quântico

O paralelismo quântico é uma característica fundamental da computação quântica, possibilitando o cálculo de uma função $f(x)$ para diferentes valores de x em superposição com apenas uma chamada da função. Seja $f(x) : \{0,1\} \rightarrow \{0,1\}$. O paralelismo quântico pode ser criado a partir do estado com dois qubits $|x,y\rangle$. Para tanto, é utilizado um operador unitário U_f que opera a transformação no estado:

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

onde $\oplus$ representa a adição módulo 2.

O qubit y é inicializado em $|0\rangle$ e x é inicializado com uma superposição uniforme de $|0\rangle$ e $|1\rangle$, esse estado pode ser obtido aplicando a porta Hadamard no qubit $|0\rangle$. Uma vez aplicado o operador U_f, é obtido um estado contendo ambas as configurações de $f(x)$:

$$\begin{aligned} U_f\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|0\rangle\right) &= \frac{1}{\sqrt{2}}(U_f\,|00\rangle + U_f\,|10\rangle) \\ &= \frac{|0, f(0)\rangle + |1, f(1)\rangle}{\sqrt{2}} \end{aligned}$$

Dessa forma, a função $f(x)$ foi avaliada para os dois valores de x, $f(0)$ e $f(1)$, simultaneamente; demonstrando o paralelismo quântico. Para criar um estado de paralelismo quântico de forma geral, para um número qualquer de bits, é utilizada a transformação de Hadamard, que consiste em n portas de Hadamard aplicadas em n qubits.

Para calcular em paralelo uma função de n bits de entrada, são necessários $n+1$ bits. A partir do estado $|0\rangle^{\otimes n}\,|0\rangle$ é aplicada a transformação de Hadamard nos n primeiros qubits. Por fim, aplicando o operador U_f é atingido o estado:

$$\frac{1}{\sqrt{2^n}} \sum_x |x\rangle\,|f(x)\rangle$$

2.4 Algoritmo de Grover

Proposto em GROVER (1997), o algoritmo de Grover é uma abordagem quântica para o problema de encontrar um determinado elemento em uma base de dados não estruturada. O

algoritmo de Grover apresenta vantagem polinomial sobre o melhor algoritmo clássico conhecido para essa tarefa.

Como entrada para o algoritmo tem-se $n+1$ qubits inicializados no estado $|0\rangle$ e um oráculo $O|x\rangle = (-1)^{f(x)}|x\rangle$, onde $f(x) = 0$ para todos os elementos na base de dados exceto para um elemento x_0 para o qual $f(x) = 1$. Para representar o espaço de busca de tamanho 2^n são necessários n qubits. O algoritmo é iniciado no estado:

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

O primeiro passo do circuito ilustrado pela Figura 2.5 é aplicar a porta Hadamard em todos os n primeiros $|0\rangle$, criando uma superposição de estados, e aplicar H no último qubit. O estado resultante será:

$$|\psi_1\rangle = \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

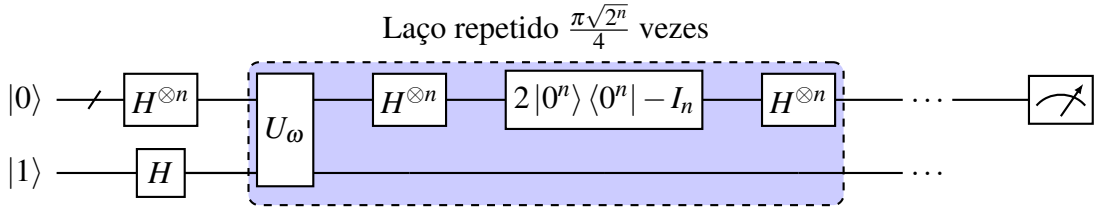


Figura 2.5: Circuito do algoritmo de Grover

A partir de então é iniciado o laço do algoritmo, destacado na Figura 2.5, a ser repetido $\frac{\pi\sqrt{2^n}}{4}$ vezes. Dois operadores são utilizados: rotação de fase e inversão sobre a média. A rotação de fase utiliza o oráculo para modificar o estado da configuração que está sendo buscada. A amplitude do estado é rotacionada em π radianos. Esse oráculo atua como uma caixa preta, realizando modificações no sistema quântico sem causar o colapso do sistema. O efeito do oráculo pode ser descrito como $|x\rangle \xrightarrow{O} (-1)^{f(x)}|x\rangle$. Onde $f(x) = 0$ para todas as configurações exceto aquela que está sendo buscada, para a qual $f(x) = 1$.

A operação de inversão sobre a média realiza a alteração das amplitudes de acordo com a média do sistema, aumentando as que estiverem abaixo ou diminuindo as que se encontram acima. Esse operador consiste em um $H^{\otimes n}$, uma mudança de fase condicional que altera os estados por -1, exceto $|0\rangle$; e ao final é aplicada outra operação $H^{\otimes n}$:

$$H^{\otimes n}[2|0\rangle\langle 0| - I]H^{\otimes n} = 2|\psi\rangle\langle\psi| - I$$

Após a repetição dos operadores no laço do algoritmo, a medição é realizada nos primeiros n qubits para obter o resultado procurado, x_0.

3

Memórias Associativas Quânticas

Neste capítulo será apresentada a memória probabilística quântica (PQM), cuja implementação em um computador quântico é o objetivo de estudo deste trabalho. A Seção 3.1 discute o conceito de memória quântica, apresentando as diferentes propostas e conceitos relacionados. Seção 3.2 versa sobre a PQM, realizando uma revisão da literatura sobre a memória. O funcionamento da memória é descrito na Seção 3.3, onde é explicado em detalhes os procedimentos de armazenamento e recuperação. Por fim, a Seção 3.4 discute algumas críticas que foram levantadas sobre o modelo.

3.1 Memórias Quânticas

As memórias são essenciais no contexto de armazenamento e processamento de informação. Um modelo de memória quântica faz uso da computação quântica para guardar e recuperar informação. As propostas de memórias quânticas na literatura são bastante diversas (SIMON et al., 2010). Para este trabalho, o foco são as propostas de memórias quânticas associativas, a considerar que o objetivo a que se propõe este trabalho é a realização de uma memória deste tipo em computadores quânticos atuais de pequena escala.

Memórias associativas são memórias endereçáveis pelo conteúdo. Diferentemente de memórias RAM, que retornam os dados guardados mediante algum endereço de memória fornecido, as memórias associativas recebem um determinado dado de busca como entrada e retornam algum dado correspondente guardado na memória.

Em VENTURA; MARTINEZ (2000) é proposta a ideia básica para a construção de um modelo de memória associativa quântica, a partir da qual a maioria das propostas posteriores se inspiraram (SCHULD; SINAYSKIY; PETRUCCIONE, 2014b). A ideia central é criar a nemória como uma superposição de padrões e utilizar um algoritmo de busca para recuperar informações armazenadas na memória. A inicialização da memória proposta originalmente segue o algoritmo quântico polinomial visto em VENTURA; MARTINEZ (1999), porém outros esquemas de inicialização também foram considerados, como o apresentado em LONG; SUN (2001). Para recuperar padrões dos estados em superposição, a proposta inicial faz uso da busca em bases de

dados não estruturadas. Uma versão modificada do algoritmo de Grover foi usada para buscar na superposição de estados que representam a memória e recuperar todos os padrões que contém uma determinada sequência. Outros trabalhos propuseram modificações para a memória de Ventura: Em ZHOU et al. (2012) é descrito um operador não-linear para substituir o uso do algoritmo de Grover na recuperação de padrões da memória; e em ANDRECUT; ALI (2003) é proposta uma modificação em um dos operadores do algoritmo de Grover para evitar que a memória encontre padrões que não existem na mesma.

A memória quântica que este trabalho se propõe a implementar em um computar quântico real foi proposta em TRUGENBERGER (2001). Ela segue a ideia base das memórias associativas quânticas tal qual discutido acima mas é capaz de recuperar padrões específicos, ao invés de padrões contendo uma determinada sequência, funcionando de fato como um memória associativa. Essa memória bem como seu funcionamento serão descritos e discutidos em detalhes na próxima seção.

Outras pesquisas relevantes para este trabalho são as propostas de implementações nos computadores quânticos atuais. Em FIGGATT et al. (2017) é realizada uma busca completa usando o algoritmo de Grover em um computador quântico programável de pequena escala (DEBNATH et al., 2016). A busca é realizada em 3 qubits, utilizando um computador quântico dispondo de 5 qubits. Para tanto, os passos do algoritmo são montados em circuitos que são posteriormente simplificados com o intuito de diminuir a quantidade de operações que atuam em mais de um qubit. Feito isso, os circuitos são transpostos para a linguagem do computador e novamente simplificados. Outros algoritmos podem vir a ser implementados seguindo a mesma ideia geral de utilizar a estrutura do computador para auxiliar possíveis simplificações.

Em SCHULD; FINGERHUTH; PETRUCCIONE (2017) é demonstrado um classificador baseado em distância implementado a partir de uma abordagem simples, utilizando apenas um circuito de interferência e duas medições em qubits únicos. A principal característica desse classificador é o armazenamento dos dados nas amplitudes dos qubits. O classificador foi implementado e avaliado no computador quântico de 5 qubits disponível na plataforma Q Experience (TEAM, 2018).

3.2 Memória Quântica Probabilística (PQM)

A chamada Memória Quântica Probabilística (MQP), ou PQM na sigla em inglês, tratada neste trabalho, é um tipo de memória associativa quântica. A principal característica de uma memória associativa é a capacidade que ela fornece em retornar dados mesmo que a entrada seja incompleta ou esteja corrompida.

Em um contexto quântico, uma memória associativa como a PQM permite um aproveitamento exponencial de recursos de memória, podendo efetivamente guardar até 2^n bits em uma memória associativa quântica de n qubits. O ganho exponencial em espaço aliado ao processamento dos padrões na memória em superposição, contribuem para o grande potencial da

PQM em aplicações de aprendizagem de máquina (SANTOS et al., 2018a).

3.2.1 Aplicações

Esta seção apresenta trabalhos que aplicaram diretamente ou fizeram uso de algum conceito ou ideia da PQM.

A PQM foi usada como base para a construção de um novo modelo de neurônio quântico para redes neurais sem peso, proposto em SILVA; OLIVEIRA; LUDERMIR (2010). Nesse modelo a PQM é utilizada como os nós de uma rede neural sem peso de maneira similar a nós RAM. No trabalho de SCHULD; SINAYSKIY; PETRUCCIONE (2014a) é proposto um algoritmo quântico para a classificação de padrões baseado no calculo da distância de Hamming usado na PQM. O classificador é aplicado para o reconhecimento de caracteres.

Em SANTOS et al. (2018a) a PQM é aplicada para um problema de seleção de arquiteturas de redes neurais. É demonstrado que a PQM é capaz de efetivamente encontrar arquiteturas quase-ótimas usando apenas uma entrada fixa para o algoritmo de recuperação. Em outro trabalho, SANTOS et al. (2018b) descreve um algoritmo quântico para validação cruzada, o método utiliza a PQM para obter as performances de um dado modelo durante as validações cruzadas.

3.3 Funcionamento

Nesta seção explicamos o modo como a PQM é construída como descrito em TRUGENBERGER (2001). A PQM em si é composta por um conjunto de registradores quânticos que exercem diferentes funções. A construção da PQM requer a implementação de dois procedimentos básicos: Um procedimento para guardar os padrões dentro da memória, o qual chamaremos de algoritmo de armazenamento; e um procedimento para recuperar padrões da memória considerando uma entrada fornecida, o qual vamos chamar de algoritmo de recuperação. Ambos algoritmos são descritos nas subseções seguintes.

3.3.1 Algoritmo de armazenamento

Seja p o número de padrões que se deseja guardar em uma PQM. O algoritmo de armazenamento tem como objetivo construir um estado quântico onde todos os p padrões se encontram em uma superposição uniforme. Tal estado quântico é representado da seguinte forma:

$$|M\rangle = \frac{1}{\sqrt{p}} \sum_{i=1}^{p} |p^i\rangle$$

onde $|M\rangle$ é o estado final da memória, p é o número de padrões armazenados, e p^i é o $i$ésimo padrão que reside na memória.

Para construir $|M\rangle$, o algoritmo de armazenamento utiliza três registradores quânticos, cada um com uma função específica. Mais uma vez, são considerados p padrões com n bits cada um. Antes que um dado padrão p^i seja efetivamente gravado na memória, ele deve ser inicializado no registrador de entrada $|p_n\rangle$ para ser processado e adicionado na memória. A memória em si é separada dos padrões de entrada que ainda não foram processados e reside no registrador $|m_n\rangle$. É importante ressaltar que m representa o registrador da memória durante a execução do algoritmo de armazenamento, enquanto M representa o estado final de m, quando todos os padrões estiverem armazenados e se encontrarem em uma superposição uniforme. Por último, o registrador $|u\rangle_2$ é usado para registrar e controlar as operações do algoritmo. O estado formado pelos três registradores descritos acima pode ser representado da seguinte forma:

$$\left|\psi_0^1\right\rangle = |p_1 p_2 \cdots p_n; u_1 u_2; m_1 m_2 \cdots m_n\rangle$$

A preparação inicial do estado $\left|\psi_0^1\right\rangle$ envolve inicializar o registrador m com $|0\rangle_n$, enquanto o registrador u é inicializado com o estado $|01\rangle$. O registrador u, como já dito, tem uma função utilitária e serve para distinguir entre padrões que já foram armazenados e padrões que ainda não foram processados pelo algoritmo. Quando o estado do segundo qubit de u é $|0\rangle$, significa que se trata de uma parte já gravada; enquanto $|1\rangle$ indica a parte que deve ser processada. O algoritmo de armazenamento controla a separação dessas duas partes usando u.

Pseudocódigo 1 *Algoritmo de Armazenamento da PQM*

1: **Procedimento** ARMAZENAR-PADRÕES($patterns_k$)
2: Preparar o estado inicial $\left|\psi_0^i\right\rangle = \left|p_1^i, \cdots, p_n^i; 01; 0_1, \cdots, 0_n\right\rangle$
3: **Para** $p^i \in patterns_k$ **Faça**
4: $\left|\psi_1^i\right\rangle = \prod_{j=1}^{n} 2XOR_{p_j^i, u_2, m_j} \left|\psi_0^i\right\rangle$
5: $\left|\psi_2^i\right\rangle = \prod_{j=1}^{n} NOT_{m_j} XOR_{p_j^i, m_j} \left|\psi_1^i\right\rangle$
6: $\left|\psi_3^i\right\rangle = nXOR_{m_1 \cdots m_n, u_1} \left|\psi_2^i\right\rangle$
7: $\left|\psi_4^i\right\rangle = CS_{u1,u2}^{p+1-i} \left|\psi_3^i\right\rangle$
8: $\left|\psi_5^i\right\rangle = nXOR_{m_1 \cdots m_n, u_1} \left|\psi_4^i\right\rangle$
9: $\left|\psi_6^i\right\rangle = \prod_{j=1}^{n} XOR_{p_j^i, m_j} NOT_{m_j} \left|\psi_5^i\right\rangle$
10: $\left|\psi_7^i\right\rangle = \prod_{j=1}^{n} 2XOR_{p_j^i, u_2, m_j} \left|\psi_6^i\right\rangle$
11: **Fim**
12: **Fim**

O Pseudocódigo 1 descreve todos os passos executados pelo algoritmo de armazenamento. O procedimento recebe k padrões, cada um com tamanho n; para cada padrão p_i são realizados 7 passos. O primeiro passo do laço, passo 4, aplica a operação $2XOR$ para realizar a cópia dos n bits do padrão de entrada para o respectivo registrador $|m\rangle$ indicado por $|u_2\rangle = 1$. Esta operação é feita com a aplicação da porta Toffoli. No passo 5 são aplicadas a operação XOR, nos registradores $|p\rangle$ e $|m\rangle$; e a operação NOT em $|m\rangle$. Este passo preenche com 1s os bits da memória que são iguais aos bits de $|p\rangle$, o que é verdade apenas para a parte que está sendo

processada no momento. No passo 6 é usada a operação *nXOR*, uma generalização da porta *CNOT* para n bits. No passo em questão, a porta é controlada por todos os bits de $|m\rangle$ e opera no primeiro bit de $|u\rangle$. Portanto, caso $|m\rangle = |1\rangle_n$, X é aplicado a $|u_1\rangle$. O passo 7 adiciona o termo em processamento na superposição $|M\rangle$, com as amplitudes já normalizadas. Para tanto é utilizada a porta CS^j, descrita pela matriz:

$$CS^j = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \sqrt{\frac{j-1}{j}} & \frac{1}{\sqrt{j}} \\ 0 & 0 & \frac{-1}{\sqrt{j}} & \sqrt{\frac{j-1}{j}} \end{bmatrix}$$

Os passos restantes tem como objetivo retornar a memória ao estado original, pronta para receber o próximo padrão. Para isso são aplicadas as operações inversas que retornam os registradores $|u\rangle$ e $|m\rangle$ aos estados iniciais, $|01\rangle$ e $|0\rangle_n$, respectivamente.

3.3.2 Algoritmo de recuperação

Pseudocódigo 2 *Algoritmo de Recuperação da PQM*

1: **Procedimento** RECUPERAR-PADRÃO($|p\rangle, |M\rangle$)
2: Carregar a entrada $|p\rangle$ no registrador quântico $|i\rangle$
3: $|\psi_1\rangle = \prod_{j=1}^{n} X_{m_j} XOR_{i_j,m_j} |\psi_0\rangle$
4: $|\psi_2\rangle = \prod_{i=1}^{n} \left(CU^{-2}\right)_{c,m_i} \prod_{j=1}^{n} U_{m_j} |\psi_1\rangle$
5: $|\psi_3\rangle = H_c \prod_{j=n}^{1} XOR_{i_j,m_j} X_{m_j} |\psi_2\rangle$
6: Medir o qubit $|c\rangle$
7: **Se** $|c\rangle == |0\rangle$ **Então**
8: Medir $|M\rangle$ para obter o estado desejado.
9: **Fim**
10: **Fim**

O algoritmo de recuperação recebe um padrão de entrada e retorna a probabilidade do mesmo se encontrar na memória. Este procedimento também faz uso de três registradores. Seja i um padrão de entrada com n bits. O registrador $|i_n\rangle$ é inicializado com o padrão de entrada. O registrador $\left|p_n^k\right\rangle$ é o registrador da memória, contendo k padrões com n bits cada. E por fim, o registrador $|c\rangle$ é um qubit de controle inicializado com H, o qual é usado para indicar a probabilidade do padrão fornecido como entrada se encontrar na memória.

O estado inicial do algoritmo é descrito por:

$$\frac{1}{\sqrt{2p}} \sum_{k=1}^{p} \left| i_1 \cdots i_n; p_1^k \cdots p_n^k; 0 \right\rangle + \\ \frac{1}{\sqrt{2p}} \sum_{k=1}^{p} \left| i_1 \cdots i_n; p_1^k \cdots p_n^k; 1 \right\rangle$$

Após a execução do algoritmo de recuperação, as amplitudes do estado são descritas por:

$$\frac{1}{\sqrt{p}}\sum_{k=1}^{p} cos\frac{\pi}{2n}d_H(i,p^k)\left|i_1\cdots i_n;p_1^k\cdots p_n^k;0\right\rangle +$$
$$\frac{1}{\sqrt{p}}\sum_{k=1}^{p} sen\frac{\pi}{2n}d_H(i,p^k)\left|i_1\cdots i_n;p_1^k\cdots p_n^k;1\right\rangle$$

onde d_H é a distância de Hamming.

Assim, os padrões mais próximos do padrão fornecido como entrada irão retornar as amplitudes mais altas e, se medirmos o qubit de controle $|c\rangle$, teremos como retorno o estado $|0\rangle$ com grande probabilidade. Da mesma forma, caso os padrões sejam distantes, teremos como resultado $|1\rangle$ com grande probabilidade quando medirmos $|c\rangle$.

3.3.3 Exemplificando a PQM

Nesta seção é exemplificado o funcionamento da memória probabilística a partir do armazenamento e recuperação de padrões de dois bits.

3.3.3.1 Armazenamento

Serão armazenados os padrões $p = \{01, 10, 11\}$. Assim, $n = 2$ é a quantidade de bits e $p = 3$ é a quantidade de padrões. Na primeira iteração do algoritmo de armazenamento, $i = 1$ será armazenado o padrão 01:

- $i = 1, p^1 = 01$

O estado inicial do algoritmo tem no primeiro registrador o padrão a ser armazenado, $|01\rangle$. O registrador auxiliar é inicializado em $|01\rangle$ e a memória em $|00\rangle$.

$$\left|\psi_0^1\right\rangle = \left|p_1^1\cdots p_n^1;01;0_1\cdots 0_n\right\rangle \rightarrow |\psi_0\rangle^1 = |01;01;00\rangle$$

O primeiro passo aplicará os operadores necessários para modificar o registrador da memória com o padrão a ser armazenado caso ainda não haja algum padrão salvo, condição que é especificada pelo segundo bit do registrador auxiliar.

$$\left|\psi_1^i\right\rangle = \prod_{j=1}^{n} 2XOR_{p_j^i,u_2,m_j}\left|\psi_0^i\right\rangle \rightarrow \left|\psi_1^1\right\rangle = |01;01;01\rangle$$

O próximo passo modifica o estado da memória para $|1\rangle_n$ quando o padrão ainda está para ser salvo, ou seja, $|u_2 = 1\rangle$.

$$\left|\psi_2^i\right\rangle = \prod_{j=1}^{n} NOT_{m_j}XOR_{p_j,m_j}\left|\psi_1^i\right\rangle \rightarrow \left|\psi_2^1\right\rangle = |01;01;11\rangle$$

Então o primeiro bit do registrador auxiliar é atualizado para 1 quando o padrão está para ser salvo.

$$\left|\psi_3^i\right\rangle = nXOR_{m_1,\cdots,m_n,u_1}\left|\psi_2^i\right\rangle \rightarrow \left|\psi_3^1\right\rangle = |01;11;11\rangle$$

Para o próximo passo temos: $p+1-i = 3+1-1$. Portanto o operador CS^j é aplicado com $j = 3$.

$$\left|\psi_4^i\right\rangle = CS_{u_1,u_2}^{p+1-i}\left|\psi_3^i\right\rangle$$

$$CS_{u_1,u_2}^3\left|\psi_3^i\right\rangle = |01\rangle|1\rangle\left(\frac{1}{\sqrt{3}}|0\rangle + \frac{\sqrt{2}}{3}|1\rangle\right)|11\rangle$$

Dessa forma, o padrão se encontra armazenado quando $|u_2\rangle = 0$.

$$\left|\psi_4^1\right\rangle = \frac{1}{\sqrt{3}}|01;10;11\rangle + \frac{\sqrt{2}}{3}|01;11;11\rangle$$

Os próximos passos revertem as alterações realizadas. Primeiro, o estado de $|u_1\rangle$:

$$\left|\psi_5^i\right\rangle = nXOR_{m_1,\cdots,m_n,u_1}\left|\psi_4^i\right\rangle \rightarrow \left|\psi_5^1\right\rangle = \frac{1}{\sqrt{3}}|01;00;11\rangle + \frac{\sqrt{2}}{3}|01;11;11\rangle$$

Em seguida, o estado da memória é revertido:

$$\left|\psi_6^i\right\rangle = \prod_{j=1}^{n} XOR_{p_j^i,m_j} NOT_{m_j}\left|\psi_5^i\right\rangle \rightarrow \left|\psi_6^1\right\rangle = \frac{1}{\sqrt{3}}|01;00;01\rangle + \frac{\sqrt{2}}{3}|01;01;01\rangle$$

Por fim, o estado da memória voltará a $|0\rangle_n$, quando não há padrão salvo. Ou seja, quando $|u_2 = 1\rangle$.

$$\left|\psi_7^i\right\rangle = \prod_{j=1}^{n} 2XOR_{p_j^i,u_2,m_j}\left|\psi_6^i\right\rangle \rightarrow \left|\psi_7^1\right\rangle = \frac{1}{\sqrt{3}}|01;00;01\rangle + \frac{\sqrt{2}}{3}|01;01;00\rangle$$

Na próxima iteração o segundo padrão será armazenado.

- $i = 2, p^2 = 11$

Os passos se repetem. Primeiramente o padrão é inicializado no primeiro registrador:

$$\frac{1}{\sqrt{3}}|11;00;01\rangle + \frac{2}{\sqrt{3}}|11;01;00\rangle$$

O padrão é copiado para a memória quando ela ainda não contém nenhum padrão:

$$\frac{1}{\sqrt{3}}|11;00;01\rangle + \frac{2}{\sqrt{3}}|11;01;11\rangle$$

O estado é modificado para $|1\rangle_n$ quando o padrão é igual a parcela da memória:

$$\frac{1}{\sqrt{3}}|11;00;01\rangle + \frac{2}{\sqrt{3}}|11;01;11\rangle$$

O estado de $|u_1\rangle$ é modificado para 1 quando a parcela da memória se encontra no estado $|1\rangle_n$:

$$\frac{1}{\sqrt{3}}|11;00;01\rangle + \frac{2}{\sqrt{3}}|11;11;11\rangle$$

Nessa iteração temos $p+1-i=2$, então obtêm-se:

$$\frac{1}{\sqrt{3}}|11;00;01\rangle + \frac{\sqrt{2}}{3}|11\rangle|1\rangle(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle)|11\rangle$$

$$\frac{1}{\sqrt{3}}|11;00;01\rangle + \frac{1}{\sqrt{3}}|11;10;11\rangle + \frac{1}{\sqrt{3}}|11;11;11\rangle$$

Os próximos passos revertem as modificações realizadas. O estado final após essa iteração será:

$$\frac{1}{\sqrt{3}}|11;00;01\rangle + \frac{1}{\sqrt{3}}|11;00;11\rangle + \frac{1}{\sqrt{3}}|11;01;00\rangle$$

Seguindo os mesmos passos, a próxima iteração irá armazenar o último padrão, resultando em:

$$|M\rangle = \frac{1}{\sqrt{3}}|00;00;01\rangle + \frac{1}{\sqrt{3}}|00;00;11\rangle + \frac{1}{\sqrt{3}}|00;00;10\rangle$$

3.3.3.2 Recuperação

Aqui será considerada uma memória com dois padrões armazenados: $\{00,11\}$. O algoritmo de recuperação será ilustrado ao tentar recuperar o padrão $i=00$ dessa memória.

O registrador $|c\rangle$ é inicializado com $H|0\rangle$. Assim, a configuração inicial será:

$$\frac{1}{\sqrt{2}}\frac{1}{\sqrt{2}}(|00;00;0\rangle + |00;11;0\rangle + |00;00;1\rangle + |00;11;1\rangle)$$

O primeiro passo marca no registrador da memória os bits iguais entre o padrão a ser recuperado e os padrões da memória com 1 caso forem iguais, ou com 0 caso contrário.

$$|\psi_1^1\rangle = \prod_{j=1}^{n} X_{m_j} XOR_{i_j,m_j}$$

$$|\psi_1\rangle = \frac{1}{2}(|00;11;0\rangle + |00;00;0\rangle + |00;11;1\rangle + |00;00;1\rangle)$$

O próximo passo mede a diferença de bits contadas no passo anterior (dada pelo número de 0s) com um sinal positivo quando $|c\rangle = 0$ ou sinal negativo quando $|c\rangle = 1$.

$$e(\frac{i\pi}{2n}H)|\psi_1\rangle = \prod_{p=1}^{n}(CU^{-2})C_m : \prod_{p=1}^{n} U_m j |\psi_1\rangle$$

$$U = \begin{bmatrix} e^{i\pi/2n} & 0 \\ 0 & 1 \end{bmatrix}$$

$$CU^{-2}|0\rangle\langle 0| \otimes 1 + |1\rangle\langle 1| \otimes U^{-2}$$

Quando aplicamos $e^{i\pi/2n}H$ em $|\psi_1\rangle$:

$$|\psi_2\rangle = \frac{1}{2}(|00;11;0\rangle + e^{2i\pi/2n}|00;00;0\rangle + |00;11;1\rangle + e^{-2i\pi/2n}|00;00;1\rangle)$$

Por fim, o estado da memória é revertido e o operador Hadamard é aplicado ao registrador $|c\rangle$.

$$|\psi_3\rangle = H_c \prod_{k=n}^{i} XOR_{ik,mk} NOT_m k |\psi_2\rangle$$

$$|\psi_3\rangle = \frac{1}{2\sqrt{2}}(2|00;00;0\rangle + 2cos(\frac{\pi}{2n})|00;11;0\rangle + 2sen(\frac{\pi}{2n})|00;11;0\rangle)$$

Agora podemos medir o qubit de controle $|c\rangle$:

$$P(|c\rangle = |0\rangle) = \left|\frac{1}{\sqrt{2}}\right|^2 + \left|\frac{1}{\sqrt{2}}cos(\frac{\pi}{2})\right|^2 = \frac{1}{2}$$

$$P(|c\rangle = |1\rangle = \left|\frac{1}{\sqrt{2}}sen(\frac{\pi}{2})\right|^2 = \frac{1}{2}$$

Dessa forma, com 50% de probabilidade o registrador $|c\rangle$ seria medido no estado 0, indicando que o padrão se encontra na memória.

3.4 Críticas

Nesta seção são mostrados trabalhos da literatura que teceram críticas e apontaram possíveis problemas relacionados com a proposta da PQM.

Em BRUN et al. (2003) é argumentado que uma memória associativa quântica como

a PQM não teria vantagens se comparada a procedimentos clássicos simples. Além disso, o processo de recuperação de padrões da PQM implicaria no colapso da memória, um problema que é reconhecido pelo próprio autor da PQM. A possível solução primeiramente proposta, a clonagem probabilística do registrador $|M\rangle$, é descreditada por não oferecer uma vantagem considerável em comparação a uma simples re-preparação do estado.

O autor da PQM responde a esses comentários em TRUGENBERGER (2003). Trugenberger afirma que não há limitações no número de padrões armazenados na memória e que o sistema pode ser implementado eficientemente para um número de padrões polinomial no número de bits, o que é considerado uma vantagem sobre os modelos clássicos, os quais são limitados linearmente no número de bits.

A PQM é citada em DUNJKO; BRIEGEL (2018) como um dos primeiros exemplos da codificação de dados clássicos nas amplitudes dos estados; um método que é bastante utilizado pelas abordagens modernas. O autor avalia que esses trabalhos pioneiros, embora sugerissem ideias criativas para a recuperação de padrões em uma memória de capacidade exponencial, ainda sofreriam de problemas de escalabilidade e outros problemas fundamentais, os quais ele não expõe e apenas se refere à discussões anteriores em BRUN et al. (2003); TRUGENBERGER (2003) e SCHULD; SINAYSKIY; PETRUCCIONE (2014a,b).

4

Plataforma Q Experience

Este capítulo apresenta a plataforma que fornece acesso aos dispositivos quânticos desenvolvidos pela IBM. A Seção 4.2 é uma introdução da plataforma Q Experience, onde foram realizados os experimentos deste trabalho. O framework Qiskit é introduzido na Seção 4.3 junto com um exemplo de implementação de um circuito quântico. A Seção 4.4 lista e descreve os dispositivos quânticos disponíveis na plataforma.

4.1 Introdução

Um bom número de computadores ou processadores quânticos já existe e alguns outros se encontram em desenvolvimento no momento. As contribuições mais antigas no campo partiram de empresas como D-Wave, IBM e Rigetti, enquanto os processadores recentemente anunciados pela Google e Intel ainda se encontram em processo de testes.

Desconsiderando computadores quânticos como o desenvolvido pela D-Wave, os quais funcionam através do modelo adiabático de computação quântica e, portanto, não podem ser considerados dispositivos capazes de realizar computação quântica universal, a maioria dos computadores quânticos disponíveis atualmente se enquadram na categoria dos dispositivos quânticos de pequena escala. Esses dispositivos tem como características principais a quantidade limitada de qubits e a alta susceptibilidade à ruídos durante a execução de operações e medições.

A IBM possui dispositivos quânticos funcionais com 5 a 20 qubits, alguns deles disponíveis publicamente para acesso. O presente trabalho se propõe a testar os primeiros computadores de pequena escala voltados para o uso comercial. A escolha dos dispositivos quânticos fornecidos pela IBM é justificada pela facilidade de acesso e uso da plataforma. As próximas seções tratam das características da plataforma Q Experience e dos dispositivos disponibilizados no sistema.

4.2 Q Experience

Q Experience é uma plataforma na internet mantida pela IBM onde o público em geral pode aprender e pesquisar sobre computação quântica. A plataforma disponibiliza uma grande

quantidade de material sobre conceitos básicos de computação quântica, algoritmos e tutoriais.

O grande diferencial da plataforma se dá pela possibilidade de acesso gratuito a computadores quânticos reais para a execução de experimentos. O acesso é facilitado pela disponibilização de um conjunto de ferramentas e simuladores para a implementação de experimentos e circuitos quânticos: um kit de desenvolvimento implementado na linguagem de programação Python (ROSSUM, 1995), o Qiskit, permite a implementação de algoritmos quânticos/clássicos e acesso direto à API da plataforma Q Experience para realizar o envio de circuitos para execução nos dispositivos quânticos na nuvem. Outra ferramenta, chamada *composer*, é um ambiente gráfico que permite a construção de circuitos quânticos usando operações básicas diretamente a partir da página do projeto, podendo ser facilmente operado por pessoas sem conhecimentos de programação.

O uso do material e ferramentas é livre para o público em geral. No entanto, o uso dos dispositivos quânticos é controlado por um sistema de créditos que limita diariamente o número de execuções possíveis por usuário. Os usuários comuns possuem poucos créditos diários enquanto pesquisadores cadastrados e parceiros da plataforma possuem limites maiores. A maioria dos computadores quânticos disponíveis na plataforma da IBM são acessíveis gratuitamente, porém os dispositivos maiores tem acesso limitado à instituições e grupos parceiros.

Os computadores quânticos da plataforma Q Experience foram utilizados em diversas aplicações e trabalhos, experimentos com simulações de vida artificial quântica (ALVAREZ-RODRIGUEZ et al., 2018), criptografia e comunicação segura (CHEN et al., 2018; BEHERA; BANERJEE; PANIGRAHI, 2017; JOY et al., 2018), *benchmarking* (WOOTTON, 2018; LINKE et al., 2017), avaliação de algoritmos (COLES et al., 2018; MANDVIWALLA; OHSHIRO; JI, 2018; BALU; CASTILLO; SIOPSIS, 2018; ZHAO et al., 2018; LEE; JOO; LEE, 2018), etc.

4.3 Framework Qiskit

Qiskit é um framework de código aberto desenvolvido pela IBM na linguagem Python e usado para o desenvolvimento e implementação de circuitos e algoritmos quânticos. Nesta seção serão apresentados alguns dos principais elementos e funcionalidades do framework e exemplos de implementação. É importante destacar que para os exemplos e experimentos realizados neste trabalho foi usada a versão 0.5.7 do Qiskit, o framework está em contínuo desenvolvimento e no momento é distribuído na versão 0.12. O código fonte do Qiskit e informações básicas de instalação e uso se encontram disponíveis no repositório dedicado ao framework no GitHub[1].

4.3.1 Implementação de circuitos quânticos

A implementação de circuitos quânticos no framework se dá de maneira bastante intuitiva utilizando um conjunto de componentes básicos representando registradores, portas e circuitos.

[1] https://github.com/Qiskit/qiskit

Vamos tomar como exemplo a implementação de um circuito simples para gerar um par EPR como na Figura. 4.1.

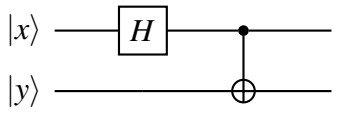


Figura 4.1: Circuito quântico para criar um par EPR

O circuito da Figura. 4.1 trabalha com dois qubits x e y e gera pares emaranhados de acordo com o estado inicial de $|xy\rangle$ como na Tabela 4.1.

Tabela 4.1: Pares emaranhados de acordo com o estado incial de $|xy\rangle$

Entrada	Saída
$\|00\rangle$	$\frac{1}{\sqrt{2}}(\|00\rangle + \|11\rangle)$
$\|01\rangle$	$\frac{1}{\sqrt{2}}(\|01\rangle + \|10\rangle)$
$\|10\rangle$	$\frac{1}{\sqrt{2}}(\|00\rangle - \|11\rangle)$
$\|11\rangle$	$\frac{1}{\sqrt{2}}(\|01\rangle - \|10\rangle)$

Para implementar este exemplo no framework Qiskit precisamos inicialmente declarar os qubits usados no circuito. Temos apenas dois qubits, x e y, e podemos declará-los como um único registrador quântico de tamanho 2:

```
q_register = qiskit.QuantumRegister(2)
```

Como queremos fazer a medição dos dois qubits para obter o resultado do circuito precisamos também declarar um registrador clássico de mesmo tamanho que o registrador quântico. O framework requer a criação de registradores clássicos apenas para armazenar o resultado de medições realizadas nos qubits do circuito:

```
c_register = qiskit.ClassicalRegister(2)
```

Uma vez que todos os registradores necessários foram declarados podemos criar um circuito quântico no framework instanciando a classe *QuantumCircuit* e passando como parâmetro os registradores quânticos e clássicos que fazem parte do circuito:

```
q_circuit = qiskit.QuantumCircuit(q_register, c_register)
```

Por fim aplicamos as portas quânticas no circuito criado. As portas quânticas no Qiskit são implementadas como metódos aplicáveis à classe *QuantumCircuit* e recebem como parâmetro os qubits em que serão aplicadas. Como pode ser visto no código abaixo, primeiramente aplicamos a porta H no primeiro qubit do registrador quântico que criamos anteriormente, em seguida aplicamos a porta *CNOT* passando o primeiro qubit como controle e o segundo como alvo. A última operação é a medição do registrador quântico, para isso usamos o método *measure*

passando os qubits que queremos medir (no caso todos os qubits do registrador quântico) e o registrador clássico que receberá o resultado da medição:

```
q_circuit.h(q_register[0])
q_circuit.cx(q_register[0], q_register[1])
q_circuit.measure(q_register, c_register)
```

Com isso implementamos o circuito da Figura 4.1 no objeto *q_circuit*. A execução do circuito quântico é feita através da função *execute* do Qiskit:

```
qiskit.execute(circuits, backend, shots, initial_layout)
```

Os principais parâmetros recebidos pela função são:

circuits Uma lista de circuitos a serem executados.

backend Onde os circuitos devem ser executados, algum dispositivo físico na nuvem ou o simulador local.

shots O número de vezes que o circuito deve ser executado.

initial_layout Um dicionário que mapeia os qubits do circuito para os qubits físicos do dispositivo.

A função execute é responsável por compilar o circuito de acordo com o *backend* e *initial_layout* escolhidos. A definição do *initial_layout* é importante quando se deseja otimizar o posicionamento dos qubits na topologia do dispositivo físico, evitando a adição de operações de *SWAP* desnecessárias durante a compilação do circuito, por exemplo. Para esse exemplo não precisamos definir um mapeamento para os qubits pois usaremos o simulador local onde os qubits simulados não possuem restrição de conectividade. Chamamos a função *execute* passando apenas o circuito que implementamos, o *backend* escolhido e o número de *shots* (execuções) desejado.

```
job = qiskit.execute([q_circuit],
                     backend='local_qasm_simulator',
                     shots=1000)
counts = job.result().get_counts()
```

Dessa forma executamos 1000 vezes o circuito *q_circuit* no simulador local do framework e recebemos um objeto com informações sobre a execução e resultados. Os resultados são fornecidos em formato de dicionário contendo as saídas observadas e o número de vezes que cada uma foi medida dentro das execuções realizadas. Para o circuito EPR do exemplo obtemos o resultado já esperado, de 1000 execuções o estado $|00\rangle$ foi observado 498 vezes e o estado $|11\rangle$ 502 vezes.

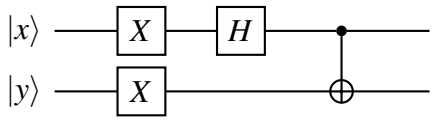


Figura 4.2: Circuito quântico para criar um par $\frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$

Para obter diferentes pares EPR só precisamos mudar o estado inicial dos qubits, aplicando a porta X antes de aplicar as operações do circuito. Se quisermos gerar um par $\frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$ por exemplo, só precisamos montar o circuito como na Figura 4.2.

O código do circuito anterior é alterado adicionando uma porta X aplicada ao dois qubits antes de qualquer operação no circuito:

```
q_circuit.x(q_register)
q_circuit.h(q_register[0])
q_circuit.cx(q_register[0], q_register[1])
q_circuit.measure(q_register, c_register)
```

E então obtemos o resultado que esperamos: são medidos os estados $|01\rangle$ e $|10\rangle$, 503 e 497 vezes, respectivamente.

4.4 Dispositivos quânticos

Atualmente a plataforma Q Experience dispõe de dois computadores quânticos reais com acesso livre: *Tenerife* com 5 qubits e *Melbourne* com 14 qubits. Além destes, existem mais dois outros dispositivos: Tokyo com 20 qubits, de uso particular de parceiros e clientes da IBM; e Yorktown com 5 qubits, que não se encontra disponível no momento mas ainda é usado internamente. Os dispositivos desenvolvidos pela IBM utilizam a tecnologia de qubits supercondutores baseada em junções de Josephson (MAKHLIN; SCHÖN; SHNIRMAN, 2001).

Cada um destes dispositivos possui uma topologia própria de acordo com a arquitetura e *layout* do processador quântico. As topologias dos dispositivos Tenerife e Melbourne podem ser vistas nas Figuras 4.3 e 4.4, respectivamente.

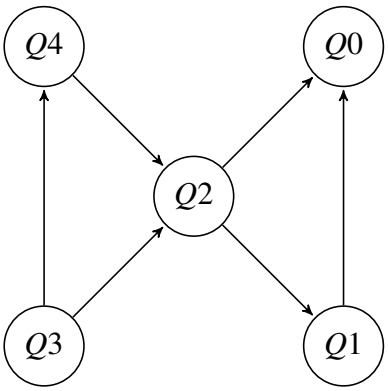


Figura 4.3: Topologia do computador de 5 qubits “Tenerife”

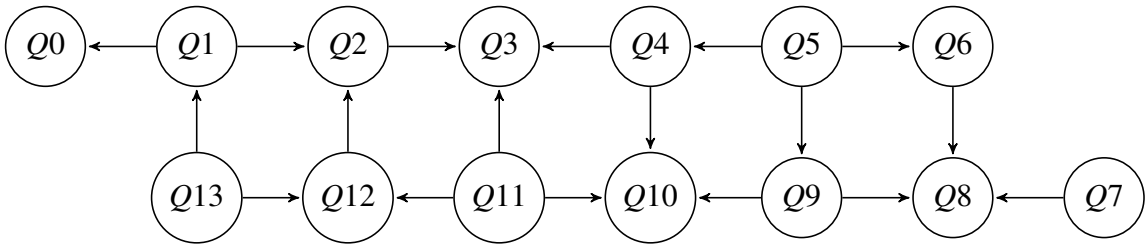


Figura 4.4: Topologia do computador de 14 qubits "Melbourne"

A topologia é representada por um grafo direcionado que descreve os qubits e as conexões possíveis entre eles, sendo portanto um aspecto muito importante a ser considerado durante a implementação dos circuitos quânticos que se deseja executar nos dispositivos físicos.

Os dispositivos da IBM implementam diretamente 3 portas básicas: $U_1(\lambda)$, $R_X(\pi/2)$ e a porta $CNOT$. As portas $U_1(\lambda)$ e $R_X(\pi/2)$ são definidas matricialmente da seguinte forma:

$$U_1(\lambda) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{bmatrix}$$

$$R_X(\pi/2) = \begin{bmatrix} 1/\sqrt{2} & -i/\sqrt{2} \\ -i/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

O conjunto completo de portas disponíveis inclui todas as portas unitárias e outras portas complexas, as quais podem ser instanciadas a partir do software e são construídas utilizando as portas básicas. O conjunto completo de portas disponíveis para os dispositivos é formado pelas portas: I, X, Y, Z, H, S, $S^\dagger$, T, $T^\dagger$, $R_X(\pi/2)$, $U_1(\lambda)$, $U_2(\lambda,\phi)$, $U_3(\lambda,\phi,\theta)$ e $CNOT$. As portas $U_2(\lambda,\phi)$ e $U_3(\lambda,\phi,\theta)$ são definidas da seguinte maneira:

$$U_2(\lambda,\phi) = \begin{bmatrix} 1/\sqrt{2} & -e^{i\lambda}/\sqrt{2} \\ e^{i\phi}/\sqrt{2} & e^{i(\lambda+\phi)}/\sqrt{2} \end{bmatrix}$$

$$U_3(\lambda,\phi,\theta) = \begin{bmatrix} \cos(\theta/2) & -e^{i\lambda}\sin(\theta/2) \\ e^{i\phi}\sin(\theta/2) & e^{i(\lambda+\phi)}\cos(\theta/2) \end{bmatrix}$$

Os dispositivos construídos pela IBM são passíveis de erros oriundos principalmente da aplicação das portas quânticas, que não são perfeitas; e da descoerência do estado dos qubits que não conseguem manter o seu estado por muito tempo. Cada dispositivo passa por ciclos de manutenção nos quais os qubits são recalibrados e as taxas de erro são mensuradas. Dessa forma, é preferível construir circuitos com o menor número possível de operações para diminuir a chance de erros na aplicação das portas quânticas, bem como diminuir o tempo necessário para a execução, consequentemente diminuindo as chances de ruído nos qubits.

4.4.1 Conectividade e a operação SWAP

A conectividade como representada pelo grafo direcionado da topologia do dispositivo quântico é responsável por ditar as conexões possíveis entre os qubits. Dessa forma, a topologia traz implicações diretas na aplicação de portas quânticas que usam 2 qubits: Apenas qubits adjacentes podem ser escolhidos.

Quando uma situação de conectividade incompatível acontece, o dispositivo precisa executar uma ou mais operações de *SWAP* para rearranjar as posições dos qubits envolvidos na operação, de forma que eles terminem em posições adjacentes. A operação de *SWAP* é definida pelo circuito da Figura 4.5.

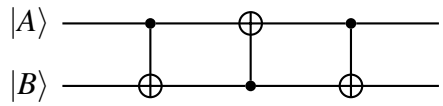


Figura 4.5: Circuito quântico para realizar a operação de *SWAP* entre dois qubits

A operação de *SWAP* é composta por 3 operações *CNOT* alternadas entre os dois qubits, $|A\rangle$ e $|B\rangle$. A operação atua fazendo a troca de estados entre os qubits, de forma que $|A\rangle$ termina com o estado de $|B\rangle$ e $|B\rangle$ termina com o estado de $|A\rangle$. Exemplificando a aplicação das portas *CNOT* do circuito *SWAP* podemos observar a troca de estados entre os qubits:

$$\begin{aligned}
|A,B\rangle &\Rightarrow |A,A\oplus B\rangle \\
&\Rightarrow |A\oplus(A\oplus B),A\oplus B\rangle \\
&\Rightarrow |A\oplus(A\oplus B),A\oplus B\rangle = |B,A\oplus B\rangle \\
&\Rightarrow |B,(A\oplus B)\oplus B\rangle \\
&\Rightarrow |B,A\rangle
\end{aligned}$$

Dessa forma, cada operação de *SWAP* acaba adicionando mais 3 portas *CNOT* ao circuito final, consequentemente aumentando o ruído.

5

Proposta híbrida clássica-quântica para a Memória Probabilística Quântica

Em seções anteriores foram expostas as principais limitações dos computadores quânticos de pequena e intermediária escalas. O número de qubits se mostra como o recurso mais crítico dentro de muitas aplicações, seguido do ruído devido à baixa qualidade dos qubits atuais.

O número de qubits está ainda atrelado à topologia do dispositivo, a qual também exerce influência no ruído final do sistema. Os poucos qubits disponíveis não se encontram completamente conectados, o que dificulta o uso livre de portas de controle como *CNOT*. Os qubits devem ser dispostos e configurados de modo a aproveitar ao máximo as conexões disponíveis na topologia e, por serem limitadas, se torna um desafio para grandes circuitos que utilizam muitas operações. Uma configuração não otimizada precisa adicionar *SWAP* gates que alteram a disposição dos qubits durante a execução do circuito de forma a posicioná-los próximos uns dos outros na topologia e assim permitir as conexões que são necessárias.

Este capítulo apresenta o método híbrido desenvolvido para a realização e execução da PQM em computadores quânticos de pequena escala. Uma análise dos recursos necessários para a implementação direta da memória, assim como descrita em TRUGENBERGER (2001), é feita na Seção 5.1. O método proposto neste trabalho é detalhado e discutido na Seção 5.2. Na Seção 5.3 são descritos os experimentos realizados no dispositivo quântico. Por fim, os resultados obtidos nos experimentos são apresentados e discutidos na Seção 5.4.

5.1 Análise de recursos

O algoritmo de armazenamento da PQM, descrito pelo Pseudocódigo 1, é composto por sete passos que são repetidos para cada um dos padrões a serem gravados na memória. Considerando uma memória com capacidade para guardar padrões de até n bits, são necessários dois registradores com n qubits cada, o registrador auxiliar p, inicializado com o padrão a ser inserido na memória; e o registrador m, o qual funciona como a memória propriamente dita. Além disso, um registrador auxiliar u com 2 qubits também é utilizado para controle do

procedimento de gravação.

Seguindo o algoritmo original, muitas operações de controle são usadas durante o processo de armazenamento dos padrões. Para cada padrão, as operações XOR, $2XOR$ e $nXOR$ são aplicadas duas vezes cada uma. A execução dessas operações é limitada pela conectividade entre qubits e, portanto, demanda o uso extensivo de operações de *SWAP* para alterar as ligações entre qubits desconectados na topologia.

O algoritmo de recuperação, descrito pelo Pseudocódigo 2, tem requisitos equivalentes quanto ao número de qubits usados. Da mesma forma, são necessários dois registradores com n qubits. O registrador M contendo a superposição final da memória; e um registrador i, cujo conteúdo se resume ao padrão de entrada que se deseja recuperar. Por fim, um registrador auxilar c é usado para indicar a probabilidade da entrada fazer parte da superposição M.

Na execução do algoritmo de recuperação são utilizadas duas operações XOR e nenhuma operação envolvendo diretamente mais do que dois qubits. As operações U e CU^{-2} não fazem parte do conjunto de portas quânticas padrão do dispositivo quântico; no entanto, ambas podem ser implementadas como uma composição de três portas quânticas básicas. Dessa forma, os dois principais fatores limitantes para uma implementação exclusivamente quântica do algoritmo de recuperação são a escalabilidade da memória, a qual depende do número de qubits disponíveis; e a grande quantidade de operações necessárias, que implica no aumento de ruído no sistema.

Considerando o que foi exposto nesta seção, para uma memória quântica com capacidade de armazenar padrões de até n bits são necessários $2n+2$ qubits para o algoritmo de armazenamento e, $2n+1$ qubits para executar o algoritmo de recuperação. Dessa forma, o número de qubits disponível pelo dispositivo quântico usado neste trabalho possibilitaria a execução de uma PQM de não mais do que 1 bit de capacidade. O número de qubits disponíveis em computadores quânticos atuais de pequena escala ainda é um recurso muito limitado, o que faz deste um problema proibitivo para a execução direta da PQM nos dispositivos quânticos atuais.

Além disso, a topologia e o número de conexões disponíveis demanda a adição de operações desnecessárias no circuito original, as quais contribuem para o aumento de ruído durante a execução e também na saída dos algoritmos.

5.2 Protocolo clássico-quântico

Dados os problemas identificados na Seção 5.1, os dois objetivos principais do método proposto neste trabalho são: diminuir o número de qubits necessários para os algoritmos de armazenamento e de recuperação e, diminuir o número de operações necessárias em ambos os algoritmos.

A arquitetura clássico-quântica dos dispositivos quânticos atuais permite o uso de variáveis e procedimentos clássicos para compilar os circuitos quânticos utilizados por um algoritmo quântico. O método proposto nesta seção faz uso desse princípio para diminuir o número de registradores e simplificar operações necessárias em ambos os algoritmos. A desvantagem

principal desta solução é a necessidade de recompilar o circuito sempre que o sistema recebe uma nova entrada, porém como pode ser visto em SANTOS et al. (2018a), a PQM possui aplicações para o problema de seleção de arquiteturas de redes neurais que utilizam apenas entradas fixas para o algoritmo de recuperação.

O algoritmo de armazenamento utiliza uma grande quantidade de operações para a construção da superposição balanceada da memória. Este é um processo que pode ser bastante simplificado se o estado final $|M\rangle$ for inicializado diretamente de maneira manual através de um procedimento clássico. Tal procedimento consiste em criar o circuito da superposição $|M\rangle$ a partir de uma determinada sequência de operações que resulte na superposição de padrões desejada, como pode ser visto no exemplo da Figura 5.1 que inicializa o registrador $|M\rangle = \frac{1}{\sqrt{2}}(|110\rangle + |111\rangle)$ usando portas X e H.

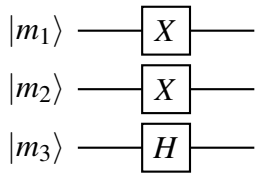


Figura 5.1: Circuito quântico usado para inicializar a superposição dos padrões 110 e 111 no registrador quântico $|M\rangle$

Dessa forma, é possível remover tanto o registrador auxiliar $|p\rangle$ quanto o registrador utilitário $|u\rangle$. O número de operações para chegar a $|M\rangle$ também é bastante reduzido, visto que não é mais necessário repetir operações para cada padrão que se deseja adicionar, nem utilizar portas de controle aplicadas a múltiplos qubits.

Além do algoritmo de armazenamento devemos considerar também o algoritmo de recuperação como visto no Pseudocódigo 2, com $|\psi_0\rangle = |i,m,c\rangle$ como estado inicial. No primeiro passo desejamos inicializar o registrador da entrada $|i\rangle$. A inicialização é feita manualmente utilizando portas X, como pode ser visto na Figura 5.2.

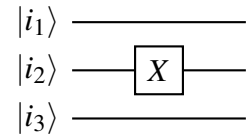


Figura 5.2: Circuito quântico usado para carregar 010 no registrador quântico $|i\rangle$

O segundo passo realiza uma comparação entre a entrada e o registrador $|M\rangle$ que contém os padrões da memória. Caso os valores comparados sejam iguais, os valores respectivos em $|M\rangle$ se tornam 1. O circuito equivalente pode ser visto na Figura 5.3.

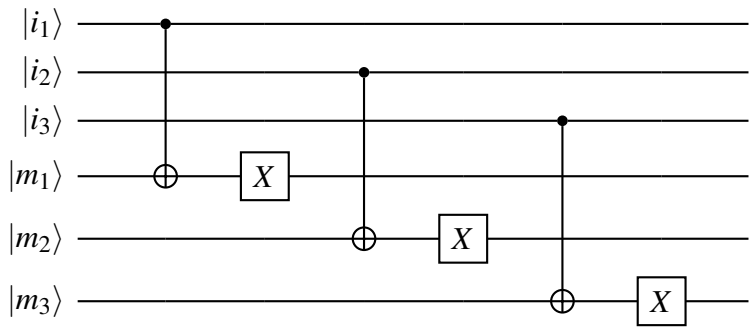


Figura 5.3: Circuito quântico usado para comparar os registradores quânticos $|i\rangle$ e $|M\rangle$

O passo 3 é de suma importância, nele é feito o cálculo da distância de Hamming, aplicando as operações U e CU^-2. Embora essas portas não tenham representantes diretas no conjunto de portas nativas dos computadores da IBM, elas podem ser descritas usando as portas u_1 e cu_1 da forma: $U = X \cdot u_1(\pi/2n) \cdot X$ e $CU^{-2} = (I \otimes X) \cdot C_{u_1}(\pi/2n) \cdot (I \otimes X)$. O circuito deste passo pode ser visto na Figura 5.4. Por fim, no passo 4 as operações realizadas são revertidas e a aplicação de H no registrador $|c\rangle$ prepara o qubit de controle para ser medido.

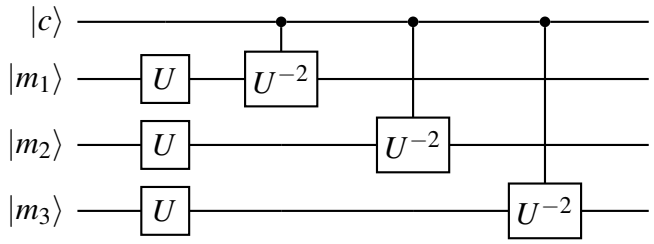


Figura 5.4: Circuito quântico usado para calcular a distância de Hamming entre a entrada e os padrões armazenados na memória

O Pseudocódigo 3 demonstra o protocolo proposto com os passos necessários para criar o circuito otimizado da PQM e considera as funcionalidades de implementação do framework Qiskit, apresentadas na Seção 4.3. O procedimento *Criar-Circuito-PQM* abrange os procedimentos de armazenamento e recuperação, portanto recebe como parâmetros os padrões da memória $patterns_k$ e a entrada i_c que será recuperada da memória.

O número de qubits necessários para o algoritmo de recuperação é diminuído guardando a entrada do algoritmo em uma variável clássica i_c com c bits e removendo o registrador de entrada $|i\rangle$. No passo 2 os padrões $patterns_k$ são guardados manualmente em superposição no registrador $|M\rangle$. Assim é possível remover todas as operações de controle que eram aplicadas do registrador de entrada para o registrador da memória no algoritmo de recuperação, substituindo pela aplicação de X no i-ésimo qubit de $|M\rangle$ apenas quando o i-ésimo bit da entrada for igual a 1, como pode ser visto no passo 4 do procedimento.

A aplicação das portas $X \cdot u_1(\pi/2n) \cdot X$ e $(I \otimes X) \cdot C_{u_1}(\pi/2n) \cdot (I \otimes X)$ equivalentes às portas U e CU^{-2}, responsáveis por fazer o cálculo da distância de Hamming, permite realizar

Pseudocódigo 3 *Protocolo Clássico-Quântico*

1: **Procedimento** CRIAR-CIRCUITO-PQM($patterns_k$, i_c)
2: Inicializar o registrador $|M\rangle$ manualmente com a superposição de $patterns_k$ da memória
3: Inicializar o registrador $|c\rangle$ com H
4: **Para** $i^i \in i_c$ **Faça**
5: **Se** $i^i == 1$ **Então**
6: Aplicar a porta X no i-ésimo qubit de $|M\rangle$
7: **Fim**
8: **Fim**
9: Aplicar a porta u_1 no registrador $|M\rangle$
10: Aplicar a porta C_{u_1} com o registrador $|M\rangle$ como alvo e o registrador $|c\rangle$ como controle
11: **Para** $i^i \in i_c$ **Faça**
12: **Se** $i^i == 1$ **Então**
13: Aplicar a porta X no i-ésimo qubit de $|M\rangle$
14: **Fim**
15: **Fim**
16: Aplicar a porta H no registrador $|c\rangle$
17: Realizar a medição do registrador $|c\rangle$
18: **Fim**

mais simplificações cancelando as operações X em sequência no algoritmo de recuperação, assim podemos aplicar apenas as portas u_1 e C_{u_1} nos passos 9 e 10. Os passos seguintes revertem as operações dos passos 3 e 4 para finalmente medir a saída do registrador $|c\rangle$ no passo 17.

Assim os algoritmos de armazenamento e recuperação tem a sua quantidade de operações reduzida no protocolo clássico-quântico através da quebra e simplificação de operações complexas em procedimentos clássico-quânticos equivalentes, os quais são condicionados pelas informações da entrada clássica para chegar ao mesmo estado resultante dos algoritmos da PQM.

As otimizações do protocolo clássico-quântico permitem diminuir o número de registradores e operações necessárias em ambos os algoritmos da PQM. Como discutido na Seção 5.1, o algoritmo original da PQM demanda $2n+2$ qubits, considerando uma memória com capacidade de armazenar padrões de até n bits. Com a otimização do protocolo clássico-quântico a PQM pode ser executada com apenas $n+1$ qubits. O número de portas que atuam em 2 qubits também foi reduzido de $3n$ para n com a remoção das portas *CNOT* usadas para comparar os registradores $|i\rangle$ e $|M\rangle$.

5.3 Descrição dos experimentos

Esta seção é dedicada para a apresentação dos experimentos enquanto a seção seguinte, Seção 5.4, apresenta e discute os resultados finais. Os experimentos relatados nesta seção foram implementados no framework de desenvolvimento Qiskit disponibilizado pela plataforma Q Experience e estão disponíveis no Github [1]. Os circuitos são compilados para a linguagem

[1] https://github.com/beesuit/pqm-ibm-experiments

QASM pelo framework e enviados em lote para a nuvem da Q Experience, onde são colocados em fila esperando execução. Uma vez executados os resultados ficam à disposição para consulta, associados à conta do usuário.

A PQM implementada seguindo o procedimento proposto neste trabalho foi testada em 4 configurações diferentes, variando de 1 a 4 qubits reservados para a memória e 1 qubit como bit de controle, o qual é necessário em todas as variações. Em cada configuração foram testados um certo número de estados diferentes de memória, totalizando 21 experimentos executados no computador *Tenerife* com 8192 *shots*, ou repetições. As configurações e estados podem ser vistos na Tabela 5.1.

Tabela 5.1: Resultados para o algoritmo de recuperação da PQM executado no computador Tenerife para diferentes estados de memória com entrada $|0\rangle_n$

Tamanho dos padrões	Estado da memória	P($\|c\rangle$=$\|0\rangle$) Tenerife	P($\|c\rangle$=$\|0\rangle$) Simulador local	P($\|c\rangle$=$\|0\rangle$) Probabilidade esperada
1	$\|0\rangle$	0.9374	1.0000	1.0000
1	$\frac{1}{\sqrt{2}}(\|0\rangle + \|1\rangle)$	0.5095	0.4974	0.5000
1	$\|1\rangle$	0.0913	0.0000	0.0000
2	$\|00\rangle$	0.9033	1.0000	1.0000
2	$\frac{1}{\sqrt{2}}(\|00\rangle + \|01\rangle)$	0.6854	0.7438	0.7500
2	$\|11\rangle$	0.1224	0.0000	0.0000
3	$\|000\rangle$	0.7618	1.0000	1.0000
3	$\frac{1}{\sqrt{2}}(\|000\rangle + \|010\rangle)$	0.6758	0.8782	0.8750
3	$\frac{1}{\sqrt{2}}(\|000\rangle + \|100\rangle)$	0.6924	0.8735	0.8750
3	$\frac{1}{\sqrt{2}}(\|000\rangle + \|001\rangle)$	0.6246	0.8804	0.8750
3	$\frac{1}{\sqrt{2}}(\|110\rangle + \|111\rangle)$	0.2925	0.1257	0.125
3	$\|111\rangle$	0.2278	0.0000	0.0000
4	$\|0000\rangle$	0.7545	1.0000	1. 0000
4	$\frac{1}{\sqrt{2}}(\|0000\rangle + \|0100\rangle)$	0.7432	0.9271	0.9268
4	$\|1000\rangle$	0.7303	0.856	0.8535
4	$\frac{1}{\sqrt{2}}(\|0100\rangle + \|1100\rangle)$	0.6844	0.67	0.6768
4	$\|1010\rangle$	0.5441	0.4961	0.5
4	$\frac{1}{\sqrt{2}}(\|0110\rangle + \|1110\rangle)$	0.4830	0.3336	0.3232
4	$\|1110\rangle$	0.3827	0.1396	0.1465
4	$\frac{1}{\sqrt{2}}(\|0111\rangle + \|1111\rangle)$	0.2423	0.073	0.0732
4	$\|1111\rangle$	0.2203	0.0000	0.0000

A disposição dos qubits na topologia do computador quântico foi realizada manualmente a fim de garantir a conectividade essencial para cada circuito e evitar a inclusão de operações

desnecessárias durante a etapa de compilação.

Além dos experimentos no computador quântico real, para fins de comparação a PQM também foi testada no simulador local incluso no framework Qiskit. O simulador local realiza uma simulação limpa do circuito, sem a introdução de erros, calculando e aplicando a matriz unitária completa do circuito nos qubits. Na seção seguinte são apresentados os resultados dos dois experimentos realizados e os cálculos teóricos que servem como base.

5.4 Resultados

Os resultados são apresentados nesta seção por meio de tabelas e gráficos que mostram as probabilidades obtidas na medição do bit de controle $|c\rangle$ que faz parte do algoritmo de recuperação da PQM. A probabilidade da entrada fornecida ao algoritmo de recuperação se encontrar armazenada no estado final da memória $|M\rangle$ é, portanto, dada por $P(|c\rangle = |0\rangle)$.

A Tabela 5.1 foi montada com todos os estados de $|M\rangle$ avaliados neste trabalho, os quais foram dispostos na coluna *Estado da memória*. As três colunas seguintes, *Tenerife*, *Simulador local* e *Probabilidade esperada*, contém os respectivos valores de $P(|c\rangle = |0\rangle)$ obtidos no computador quântico Tenerife, no simulador implementado na biblioteca Qiskit e, por fim, o valor teórico esperado para a probabilidade, o qual foi calculado numericamente de acordo com a descrição do algoritmo de recuperação da PQM.

A comparação das probabilidades apresentadas pode ser melhor observada na Figura 5.7, contendo os resultados para a memória de 1 qubit com padrões de entrada 0 e 1; Figura 5.8, com as probabilidades obtidas para as entradas 00 e 11 na memória de 2 qubits; Figura 5.10 e Figura 5.11, com os valores para memória de 3 qubits com entradas 000 e 111, e memória de 4 qubits com entradas 0000 e 1111. Em todos os gráficos o eixo horizontal representa a probabilidade de $|c\rangle = |0\rangle$ enquanto o eixo vertical descreve o estado da memória em que foi feita a medição.

5.4.1 Discussão dos resultados

O algoritmo de Grover foi comparado em execuções no simulador e no computador quântico. O circuito executado pode ser visto na Figura 5.5. O circuito prepara uma superposição com 2 qubits e busca pelo padrão 11.

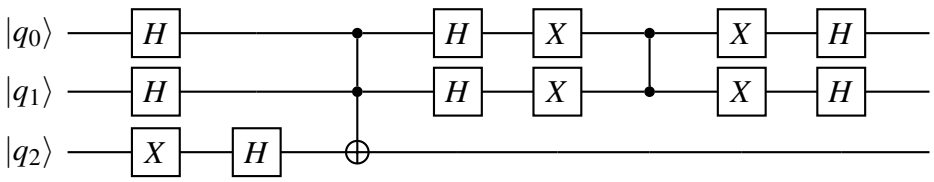
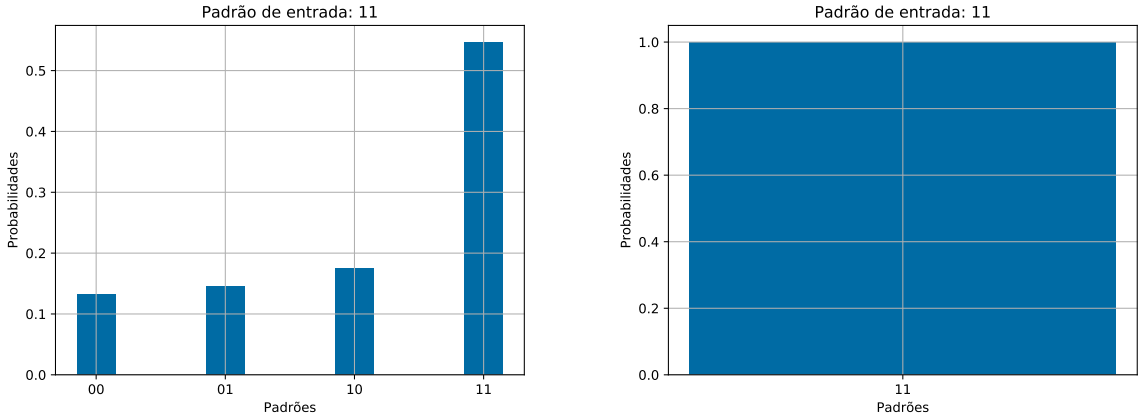


Figura 5.5: Circuito quântico do algoritmo de Grover

Os resultados dos experimentos realizados com o algoritmo de Grover podem ser vistos

na Figura 5.6. A grande quantidade de portas necessárias para o circuito de Grover adiciona um alto nível de ruído na medição da saída. É possível observar uma grande diferença da saída entre o algoritmo executado no simulador, visto na Figura 5.6b, comparado com a execução no computador quântico, visto na Figura 5.6a. Assim como na simulação, o resultado deveria retornar o padrão 11 com 100% de probabilidade, no entanto, a execução no dispositivo quântico retorna o padrão correto com apenas pouco mais de 50% de probabilidade junto com os outros padrões incorretos.

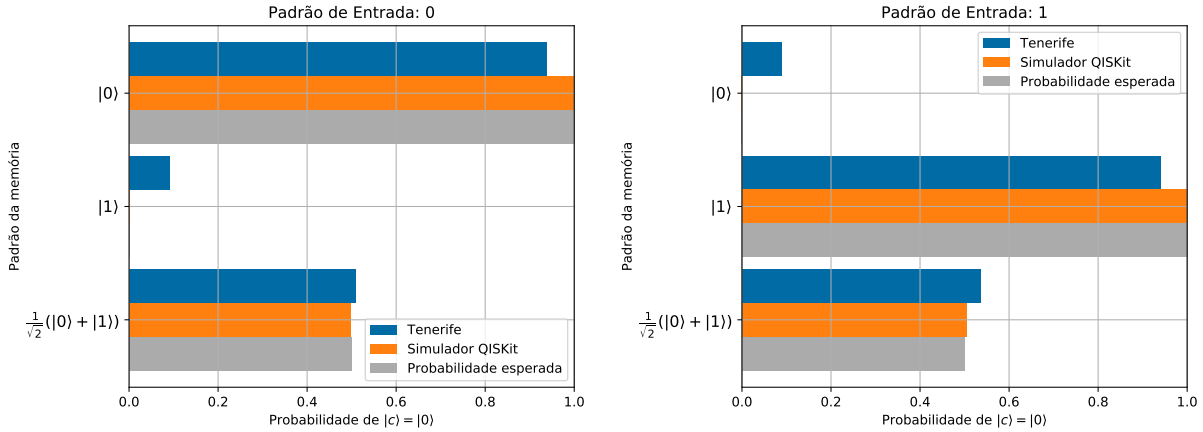


(a) Resultados obtidos no dispositivo quântico

(b) Resultados obtidos no simulador local

Figura 5.6: Resultados obtidos para o algoritmo de Grover executado no dispositivo quântico e no simulador local.

Nos resultados da PQM na Figura 5.7a podemos observar a discrepância entre os valores de probabilidades obtidos para $|c\rangle = |0\rangle$ dentre os diferentes contextos em que o algoritmo de recuperação foi executado considerando 1 qubit. Nessa configuração em específico é possível verificar que não existe uma grade diferença entre os valores obtidos pelo Tenerife e os valores teóricos para cada entrada e valores de memória, o mesmo pode ser observado na Figura 5.7b.

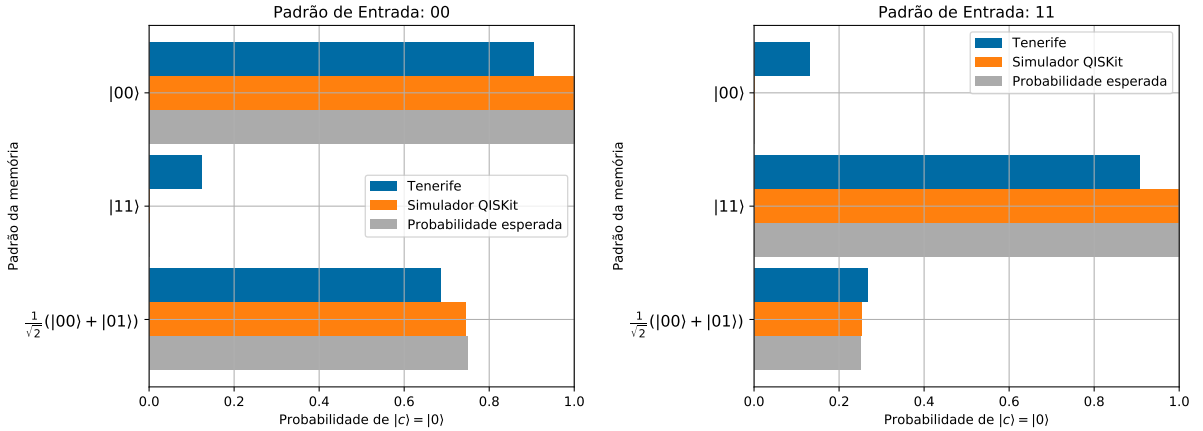


(a) Resultados para o padrão de entrada 0

(b) Resultados para o padrão de entrada 1

Figura 5.7: Resultados obtidos para uma PQM de 1 qubit executando o algoritmo de recuperação no dispositivo quântico e no simulador local. Juntamente com o valor esperado calculado numericamente.

Na Figura 5.8a são mostrados os resultados com 2 qubits. Existe um erro total médio mais elevado, principalmente quando se tem configurações de memória contendo apenas um padrão. Esse indício é melhor confirmado na Figura 5.8b onde, para a configuração de memória armazenando mais de um padrão temos os valores bem próximos, enquanto os demais apresentam uma diferença mais elevada.



(a) Resultados para o padrão de entrada 00

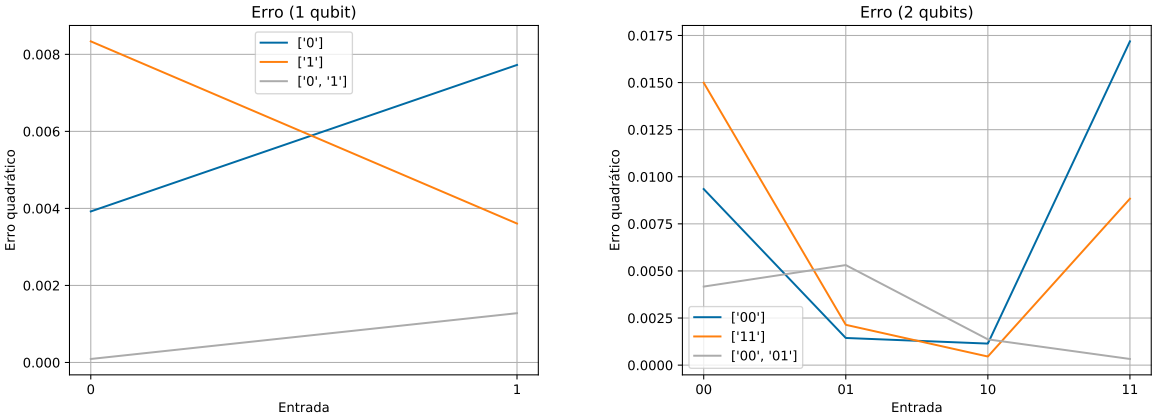
(b) Resultados para o padrão de entrada 11

Figura 5.8: Resultados obtidos para uma PQM de 2 qubits executando o algoritmo de recuperação no dispositivo quântico e no simulador local. Juntamente com o valor esperado calculado numericamente.

Os gráficos do erro quadrático da Figura 5.9 demonstram os picos de erros para as configurações de 1 e 2 qubits em função das entradas fornecidas ao algoritmo de recuperação. Na Figura 5.9a para a configuração de 1 qubit, o erro é menor quando a memória armazena dois

padrões, para ambas entradas consideradas. Para as outras configurações, o erro é maior quando a entrada é diferente do conteúdo da memória.

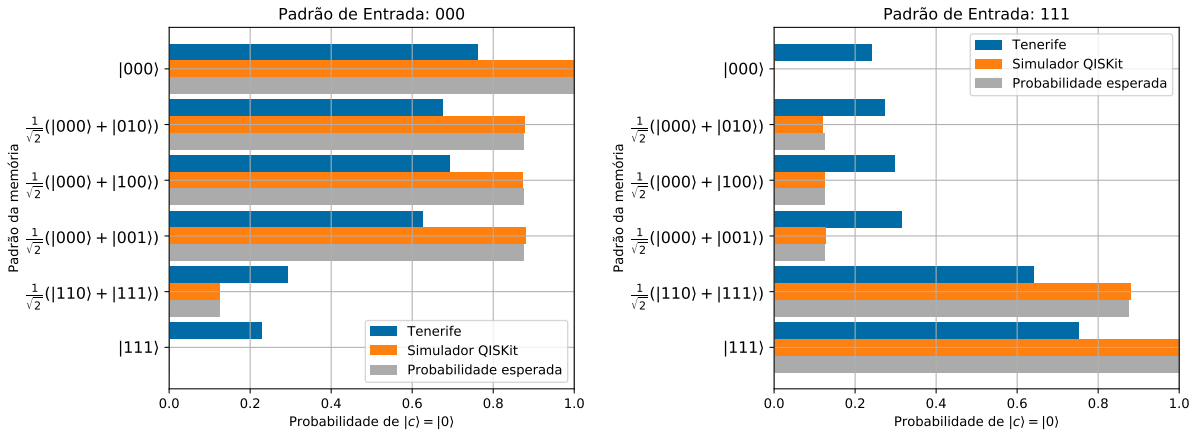
A Figura 5.9b mostra os valores de erro para a configuração de 2 qubits. A mesma tendência é reproduzida aqui, quando a memória armazena mais de um padrão tem-se um erro quadrático médio menor e o erro aumenta quando a entrada difere do conteúdo.



(a) Erro quadrático para 1 qubit **(b)** Erro quadrático para 2 qubits

Figura 5.9: Erro quadrático calculado para as configurações de memória de 1 e 2 qubits

Para a configuração de 3 qubits temos os resultados da Figura 5.10a. Essa configuração é a que apresentou maior distância para os valores teóricos, o erro se mostrou constante para todas as configurações de memória, em especial considerando a entrada $|111\rangle$, como pode ser visto na Figura 5.10b.

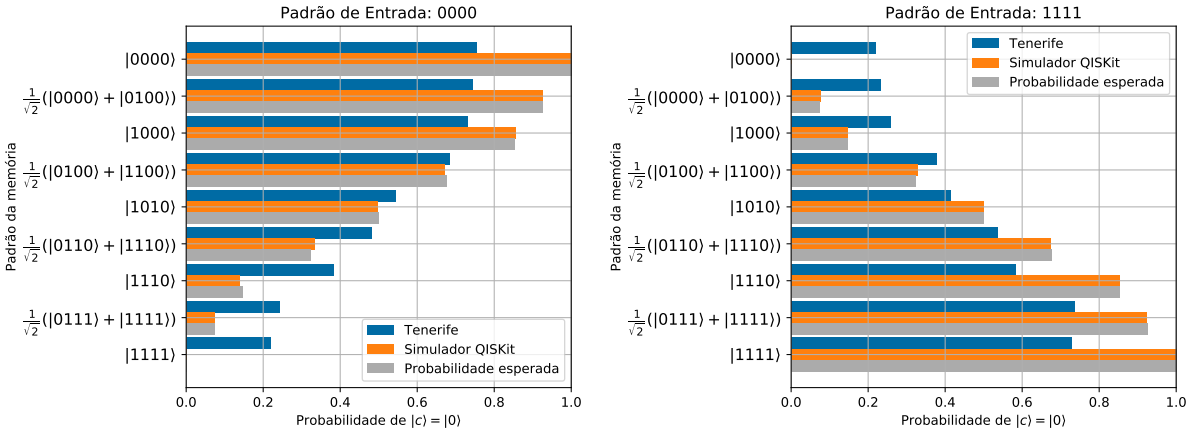


(a) Resultados para o padrão de entrada 000 **(b)** Resultados para o padrão de entrada 111

Figura 5.10: Resultados obtidos para uma PQM de 3 qubits executando o algoritmo de recuperação no dispositivo quântico e no simulador local. Juntamente com o valor esperado calculado numericamente.

Por último, a configuração de 4 qubits ainda exibe diferenças consideráveis entre os

contextos de execução. Embora consiga boa equiparação em algumas poucas configurações de memória, como por exemplo, na Figura 5.11a com as configurações $\frac{1}{\sqrt{2}}(|0100\rangle + |1100\rangle)$ e $|1010\rangle$. Para a entrada 1111, como pode ser visto na Figura 5.11b, os resultados foram na mesma escala. Até mesmo as duas configurações que apresentaram os melhores resultados foram as mesmas.



(a) Resultados para o padrão de entrada 0000

(b) Resultados para o padrão de entrada 1111

Figura 5.11: Resultados obtidos para uma PQM de 4 qubits executando o algoritmo de recuperação no dispositivo quântico e no simulador local. Juntamente com o valor esperado calculado numericamente.

6

Conclusões e Trabalhos Futuros

Neste trabalho, um modelo quântico de memória associativa foi experimentalmente avaliado em um dispositivo quântico real de pequena escala. Para tanto, foi necessário realizar modificações no algoritmo da memória, as quais foram justificadas pela avaliação de viabilidade da implementação quântica considerando a arquitetura de um computador de apenas 5 qubits. A avaliação em questão mostra que uma implementação direta da memória é impraticável dada as limitações da arquitetura do dispositivo quântico considerado neste trabalho. Com isso, foi proposto uma versão clássico-quântica da memória probabilística, utilizando melhor as capacidades dos dispositivos quânticos que se encontram à disposição atualmente, sem contudo alterar o funcionamento esperado da memória tal como proposta originalmente.

A proposta de adaptação da memória probabilística discutida neste trabalho, requer apenas $n+1$ qubits para armazenar 2^n padrões diferentes de até n bits. Para alcançar este resultado, foram feitas modificações no algoritmo de recuperação, transformando-o em um procedimento híbrido clássico-quântico. Desta forma, o número necessário de qubits foi reduzido de $2n+1$ para $n+1$. Além disso, a quantidade de operadores quânticos que agem em 2 qubits foi reduzida de $3n$ para n. A redução no número de qubits e portas que atuam em 2 qubits é de alta relevância para computadores quânticos de pequena e média escala. Com essa redução de dimensionalidade, foi possível executar uma memória quântica associativa de até 4 qubits em um dispositivo quântico de 5 qubits.

Como contribuições desse trabalho, podem ser destacadas inicialmente a análise de viabilidade da realização de uma memória associativa quântica em computadores quânticos reais. É preciso levar em consideração a quantidade de qubits disponíveis bem como suas conexões, indicadas pela arquitetura do dispositivo. Além disso, a proposta do algoritmo híbrido clássico-quântico constitui-se como a maior contribuição desse trabalho. A partir dele foi possível executar a memória em um computador quântico de pequena escala. A mesma ideia geral de simplificação proposta pode ser utilizada em outros algoritmos e dispositivos com arquiteturas similares. Por fim, a avaliação da memória a partir dos experimentos realizados mostrou que o funcionamento está dentro do esperado apesar da presença de erros comuns ao dispositivo.

6.1 Trabalhos Futuros

Por fim, esta seção conclui este trabalho levantando possíveis caminhos de pesquisa que podem ser pensados a partir do que foi desenvolvido aqui. Um possível trabalho futuro consiste em utilizar a abordagem proposta para avaliar outras memórias quânticas (ANDRECUT; ALI, 2003; SCHULD; FINGERHUTH; PETRUCCIONE, 2017). Assim, os processos de diferentes propostas de memórias quânticas poderiam ser avaliados e simplificados para permitir a execução em uma determinada arquitetura. Dessa forma, seria possível comparar a performance de diferentes memórias quânticas em computadores quânticos de pequena escala.

Investigar os erros da PQM no dispositivo quântico é outra linha interessante a ser seguida como trabalho futuro. Executando mais experimentos com um número razoável de repetições pode ser possível modelar os erros do dispositivo físico e desenvolver um método para corrigir as saídas do algoritmo de recuperação e melhorar o funcionamento da memória.

Outros possíveis trabalhos futuros envolvem os computadores quânticos de média-escala que estão por vir. Usando computadores quânticos com uma maior quantidade de qubits, as possibilidades de implementação são ainda mais amplas. Por exemplo, seria possível avaliar o desempenho da memória probabilística quântica utilizada em aplicações de aprendizagem de máquina. É preciso enfatizar que a arquitetura do dispositivo é tão relevante quanto a quantidade de qubits à disposição. Havendo computadores maiores com topologias que possuam ao menos um qubit conectado a todos os outros, pode-se avaliar o desempenho da PQM em tarefas de classificação de maior escala.

Referências

ALVAREZ-RODRIGUEZ, U. et al. Quantum artificial life in an IBM quantum computer. **Scientific reports**, [S.l.], v.8, 2018.

ANDRECUT, M.; ALI, M. Quantum associative memory. **International Journal of Modern Physics B**, [S.l.], v.17, n.12, p.2447–2472, 2003.

BALU, R.; CASTILLO, D.; SIOPSIS, G. Physical realization of topological quantum walks on IBM-Q and beyond. **Quantum Science and Technology**, [S.l.], v.3, n.3, p.035001, 2018.

BEHERA, B. K.; BANERJEE, A.; PANIGRAHI, P. K. Experimental realization of quantum cheque using a five-qubit quantum computer. **Quantum Information Processing**, [S.l.], v.16, n.12, p.312, 2017.

BENIOFF, P. The computer as a physical system: a microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. **Journal of statistical physics**, [S.l.], v.22, n.5, p.563–591, 1980.

BENNETT, C. H. et al. Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. **Physical review letters**, [S.l.], v.70, n.13, p.1895, 1993.

BENNETT, C. H.; WIESNER, S. J. Communication via one-and two-particle operators on Einstein-Podolsky-Rosen states. **Physical review letters**, [S.l.], v.69, n.20, p.2881, 1992.

BRAVYI, S.; GOSSET, D.; KOENIG, R. Quantum advantage with shallow circuits. **Science**, [S.l.], v.362, n.6412, p.308–311, 2018.

BRUN, T. et al. Comment on "Probabilistic Quantum Memories". **Phys. Rev. Lett.**, [S.l.], v.91, p.209801, Nov 2003.

CHEN, X. et al. Experimental Cryptographic Verification for Near-Term Quantum Cloud Computing. **arXiv preprint arXiv:1808.07375**, [S.l.], 2018.

COLES, P. J. et al. Quantum Algorithm Implementations for Beginners. **arXiv preprint arXiv:1804.03719**, [S.l.], 2018.

DEBNATH, S. et al. Demonstration of a small programmable quantum computer with atomic qubits. **Nature**, [S.l.], v.536, n.7614, p.63, 2016.

DEUTSCH, D. E. Quantum computational networks. **Proc. R. Soc. Lond. A**, [S.l.], v.425, n.1868, p.73–90, 1989.

DUNJKO, V.; BRIEGEL, H. J. Machine learning & artificial intelligence in the quantum domain: a review of recent progress. **Reports on Progress in Physics**, [S.l.], v.81, n.7, p.074001, 2018.

FEYNMAN, R. P. Simulating physics with computers. **International journal of theoretical physics**, [S.l.], v.21, n.6-7, p.467–488, 1982.

FIGGATT, C. et al. Complete 3-qubit Grover search on a programmable quantum computer. **Nature communications**, [S.l.], v.8, n.1, p.1918, 2017.

GROVER, L. K. A fast quantum mechanical algorithm for database search. In: ACM SYMPOSIUM ON THEORY OF COMPUTING. **Proceedings...** [S.l.: s.n.], 1996. p.212–219.

GROVER, L. K. Quantum mechanics helps in searching for a needle in a haystack. **Physical review letters**, [S.l.], v.79, n.2, p.325, 1997.

JOY, D. et al. In principle demonstration of quantum secret sharing in the IBM quantum computer. **arXiv preprint arXiv:1807.03219**, [S.l.], 2018.

KELLY, J. A preview of Bristlecone, Google's new quantum processor. **Google Research Blog**, [S.l.], v.5, 2018.

KNIGHT, W. IBM Raises the Bar with a 50-Qubit Quantum Computer. **Sighted at MIT Review Technology: https://www. technologyreview. com/s/609451/ibm-raises-the-bar-with-a-50-qubit-quantum-computer**, [S.l.], 2017.

LEE, Y.; JOO, J.; LEE, S. Hybrid quantum linear equation algorithm and its experimental test on IBM Quantum Experience. **arXiv preprint arXiv:1807.10651**, [S.l.], 2018.

LINKE, N. M. et al. Experimental comparison of two quantum computing architectures. **Proceedings of the National Academy of Sciences**, [S.l.], v.114, n.13, p.3305–3310, 2017.

LONG, G.-L.; SUN, Y. Efficient scheme for initializing a quantum register with an arbitrary superposed state. **Physical Review A**, [S.l.], v.64, n.1, p.014303, 2001.

MAKHLIN, Y.; SCHÖN, G.; SHNIRMAN, A. Quantum-state engineering with Josephson-junction devices. **Reviews of modern physics**, [S.l.], v.73, n.2, p.357, 2001.

MANDVIWALLA, A.; OHSHIRO, K.; JI, B. Implementing Grover's Algorithm on the IBM Quantum Computers. In: IEEE INTERNATIONAL CONFERENCE ON BIG DATA (BIG DATA), 2018. **Anais...** [S.l.: s.n.], 2018. p.2531–2537.

NIELSEN, M. A.; CHUANG, I. **Quantum computation and quantum information**. [S.l.]: AAPT, 2002.

OTTERBACH, J. et al. Unsupervised machine learning on a hybrid quantum computer. **arXiv preprint arXiv:1712.05771**, [S.l.], 2017.

PRESKILL, J. Quantum computing and the entanglement frontier. **arXiv preprint arXiv:1203.5813**, [S.l.], 2012.

PRESKILL, J. Quantum Computing in the NISQ era and beyond. **arXiv preprint arXiv:1801.00862**, [S.l.], 2018.

REAGOR, M. et al. Demonstration of universal parametric entangling gates on a multi-qubit lattice. **Science advances**, [S.l.], v.4, n.2, p.eaao3603, 2018.

ROSSUM, G. van. **Python tutorial**. Amsterdam: Centrum voor Wiskunde en Informatica (CWI), 1995. (CS-R9526).

SANTOS, P. dos et al. Quantum enhanced k-fold cross-validation. In: BRAZILIAN CONFERENCE ON INTELLIGENT SYSTEMS (BRACIS), 2018. **Anais...** [S.l.: s.n.], 2018b. p.194–199.

SANTOS, P. G. dos et al. Quantum enhanced cross-validation for near-optimal neural networks architecture selection. **International Journal of Quantum Information**, [S.l.], p.1840005, 2018a.

SCHULD, M.; FINGERHUTH, M.; PETRUCCIONE, F. Implementing a distance-based classifier with a quantum interference circuit. **EPL (Europhysics Letters)**, [S.l.], v.119, n.6, p.60002, 2017.

SCHULD, M.; SINAYSKIY, I.; PETRUCCIONE, F. Quantum computing for pattern classification. In: PACIFIC RIM INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE. **Anais...** [S.l.: s.n.], 2014. p.208–220.

SCHULD, M.; SINAYSKIY, I.; PETRUCCIONE, F. The quest for a quantum neural network. **Quantum Information Processing**, [S.l.], v.13, n.11, p.2567–2586, 2014.

SHOR, P. W. Algorithms for quantum computation: discrete logarithms and factoring. In: FOUNDATIONS OF COMPUTER SCIENCE, 1994 PROCEEDINGS., 35TH ANNUAL SYMPOSIUM ON. **Anais...** [S.l.: s.n.], 1994. p.124–134.

SHOR, P. W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. **SIAM review**, [S.l.], v.41, n.2, p.303–332, 1999.

SILVA, A.; OLIVEIRA, W. de; LUDERMIR, T. A weightless neural node based on a probabilistic quantum memory. In: ELEVENTH BRAZILIAN SYMPOSIUM ON NEURAL NETWORKS, 2010. **Anais...** [S.l.: s.n.], 2010. p.259–264.

SIMON, C. et al. Quantum memories. **The European Physical Journal D**, [S.l.], v.58, n.1, p.1–22, 2010.

TEAM, I. Q. **IBM Q 5 Tenerife backend specification V1.3.0**. Online; accessed 24-September-2018, `https://ibm.biz/qiskit-tenerife`.

TOFFOLI, T. Reversible computing. In: INTERNATIONAL COLLOQUIUM ON AUTOMATA, LANGUAGES, AND PROGRAMMING. **Anais...** [S.l.: s.n.], 1980. p.632–644.

TRUGENBERGER, C. A. Probabilistic quantum memories. **Physical Review Letters**, [S.l.], v.87, n.6, p.067901, 2001.

TRUGENBERGER, C. A. Trugenberger replies. **Physical Review Letters**, [S.l.], v.91, n.20, p.209802, 2003.

VENTURA, D.; MARTINEZ, T. Initializing the amplitude distribution of a quantum state. **Foundations of Physics Letters**, [S.l.], v.12, n.6, p.547–559, 1999.

VENTURA, D.; MARTINEZ, T. Quantum associative memory. **Information Sciences**, [S.l.], v.124, n.1-4, p.273–296, 2000.

WOOTTON, J. R. Benchmarking of quantum processors with random circuits. **arXiv preprint arXiv:1806.02736**, [S.l.], 2018.

ZHAO, Z. et al. Bayesian Deep Learning on a Quantum Computer. **arXiv preprint arXiv:1806.11463**, [S.l.], 2018.

ZHOU, R. et al. Quantum associative neural network with nonlinear search algorithm. **International Journal of Theoretical Physics**, [S.l.], v.51, n.3, p.705–723, 2012.