Pós-Graduação em Ciência da Computação

Lucas dos Reis Silva

Caminhos Evolutivos para Circuitos Quânticos: Explorando Algoritmos Genéticos na Geração Automatizada de Designs

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

Recife
2024

Lucas dos Reis Silva

Caminhos Evolutivos para Circuitos Quânticos: Explorando Algoritmos Genéticos na Geração Automatizada de Designs

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

Área de Concentração: *Aprendizagem de Máquina Quântico, Algoritmos Genéticos*
Orientador: *Paulo Salgado Gomes de Mattos Neto, Fernando Maciano de Paula Neto*

Recife
2024

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Silva, Lucas dos Reis.
Caminhos Evolutivos para Circuitos Quânticos: Explorando Algoritmos Genéticos na Geração Automatizada de Designs / Lucas dos Reis Silva. - Recife, 2023.
65 p : il., tab.

Orientador(a): Paulo Salgado Gomes de Mattos Neto
Cooorientador(a): Fernando Maciano de Paula Neto
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Engenharia da Computação - Bacharelado, 2023.
Inclui referências, apêndices.

1. Aprendizagem de Máquina Quântico. 2. Algoritmos Genéticos. 3. Circuitos Quânticos Variacionais. I. Mattos Neto, Paulo Salgado Gomes de. (Orientação). II. Paula Neto, Fernando Maciano de. (Coorientação). IV. Título.

000 CDD (22.ed.)

LUCAS DOS REIS SILVA

Caminhos Evolutivos para Circuitos Quânticos: Explorando Algoritmos Genéticos na Geração Automatizada de Designs

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

.

Aprovado em: 17/10/2024.

BANCA EXAMINADORA

_____
Profª. Drª. Paulo Salgado Gomes de Mattos Neto (Orientador)
Universidade Federal de Pernambuco - UFPE

_____
Prof. Dr. Fernando Maciano de Paula Neto (Orientador)
Universidade Federal de Pernambuco - UFPE

_____
Prof. Dr. Stefan Michael Blawid (Examinador Interno)
Universidade Federal de Pernambuco - UFPE

Dedico esta dissertação à minha família e a todos que me apoiaram ao longo desta jornada.

AGRADECIMENTOS

Gostaria de expressar minha mais profunda gratidão aos meus pais, que, além de todo o amor e apoio incondicional, sempre estimularam minha curiosidade intelectual e me ofereceram as bases que carrego comigo até hoje.

Agradeço sinceramente ao professor Fernando Maciano, pela paciência e por me introduzir ao fascinante universo da computação quântica, onde pude unir minhas paixões por computação e física.

Registro também meus agradecimentos ao professor Paulo Salgado, por me apresentar ao campo dos algoritmos bioinspirados, demonstrando que a ciência da computação transcende os limites dos bits e se expande para áreas inovadoras e desafiadoras.

Sou grato a todos os professores que me acompanharam ao longo da minha trajetória acadêmica, pelos valiosos ensinamentos, pelas lições fundamentais e, sobretudo, pelas críticas construtivas, que foram essenciais para o meu crescimento pessoal e intelectual.

Às instituições de ensino que fizeram parte da minha formação, em especial ao Educandário Mendesrios, que me inspirou a sonhar com uma universidade pública, e à Universidade Federal de Pernambuco (UFPE), por proporcionar um ambiente acadêmico estimulante e os recursos necessários para meu desenvolvimento intelectual, expresso meu sincero agradecimento.

Aos meus colegas e amigos, cujo apoio foi indispensável para enfrentar os desafios da vida acadêmica e da experiência de morar sozinho, deixo meu reconhecimento e gratidão.

À minha namorada Carol, que sempre esteve ao meu lado, cobrando com carinho, motivando e me ajudando a superar os desafios ao longo desta jornada, minha eterna gratidão.

Por fim, estendo meus agradecimentos à banca examinadora, pela disponibilidade, pelas valiosas contribuições e por enriquecer este trabalho com suas análises e sugestões.

“Em algum lugar, algo incrível está esperando para ser descoberto."

–Carl Sagan

ABSTRACT

Quantum computing stands out for its promise of superior processing power and the development of innovative algorithms, with significant impacts on cryptographic security. However, in the NISQ (Noisy Intermediate-Scale Quantum) era, quantum computers face limitations such as noise and low decoherence time, which restrict their applicability. Hybrid strategies like variational quantum circuits (VQC) have gained relevance by combining aspects of classical and quantum computing to mitigate these limitations. This dissertation proposes a search algorithm based on genetic algorithms, capable of generating variational quantum circuits optimized for efficiency and performance in machine learning problems. The goal is to create minimal and effective architectures to solve both synthetic and real-world problems, without relying on prior information about data loading and processing. Various algorithm configurations were tested, generating efficient quantum architectures, with performance comparable to classical algorithms in some cases. The developed solution proves to be a promising method for creating quantum architectures, offering a viable approach to solving classification problems within the limitations of current quantum computers.

Keywords: Quantum Computing, Variational Quantum Circuits, Genetic Algorithms.

RESUMO

A computação quântica destaca-se pela promessa de uma capacidade de processamento superior e pelo desenvolvimento de algoritmos inovadores, com impactos significativos na segurança criptográfica. Contudo, na era NISQ (Noisy Intermediate-Scale Quantum), os computadores quânticos enfrentam limitações como ruído e baixo tempo de decoerência, o que restringe sua aplicabilidade. Para mitigar essas limitações, estratégias híbridas, como os circuitos quânticos variacionais, têm ganhado relevância ao combinar aspectos da computação clássica e quântica. Esta dissertação propõe um algoritmo de busca baseado em algoritmos genéticos, capaz de gerar circuitos quânticos variacionais otimizados para eficiência e desempenho em problemas de aprendizagem de máquina. O objetivo é criar arquiteturas mínimas e eficazes para resolver problemas sintéticos e reais, sem depender de informações prévias sobre o carregamento e processamento dos dados. Diversas configurações do algoritmo foram testadas, gerando arquiteturas quânticas eficientes, com desempenho comparável ao de algoritmos clássicos em alguns casos. A solução desenvolvida mostra-se promissora para a criação de arquiteturas quânticas, representando uma abordagem viável para a resolução de problemas de classificação dentro das limitações dos computadores quânticos atuais.

Palavras-chave: Computação Quântica, Circuitos Quânticos Variacionais, Algoritmos Genéticos.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ACRÔNIMOS

AG	*Algoritmo Genético*
CMA-ES	*Estratégia de Evolução por Covariância Matricial*
CNOT	*Controlled-NOT*
EQAS	*Busca de Arquitetura Quântica Baseada em Evolução*
EVQE	*Solucionador Quântico Variacional Evolutivo*
MoG-VQE	*Solucionador Quântico Variacional Multiobjetivo*
MQNE	*Neuroevolução Quântica Markoviana*
MSE	*Erro Quadrático Médio*
NISQ	*Noisy Intermediate-Scale Quantum*
NSGA-II	*Algoritmo Genético de Classificação Não-dominada II*
PQC	*Circuitos Quânticos Parametrizados*
QAOA	*Algoritmo Quântico Aproximado de Otimização*
QAS	*Quantum Architecture Search*
QCL	*Aprendizado de Circuitos Quânticos*
QSVM	*Máquinas de Vetor de Suporte Quântico*
RL	*Reforço de Aprendizado*
TCC	*Trabalho de Conclusão de Curso*
VQA	*Algoritmo Quântico Variacional*
VQC	*Circuito Quântico Variacional*
VQE	*Solucionador de Autovalores Quânticos Variacionais*

LISTA DE SÍMBOLOS

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

Neste capítulo, será apresentado o contexto em que este Trabalho de Conclusão de Curso (TCC) está inserido, bem como a motivação que justifica sua realização. A escolha do tema está fundamentada no crescente interesse global pelo campo da computação quântica, impulsionado por descobertas significativas, como os algoritmos de Shor Shor (1997) e Grover Grover (1996), que demonstraram a capacidade de resolver problemas com custos polinomiais em relação aos métodos tradicionais. A computação quântica, em particular, tem atraído a atenção de acadêmicos e profissionais devido ao seu potencial para revolucionar a maneira como o processamento de informações é realizado, oferecendo uma alternativa eficiente para a solução de problemas complexos.

Além disso, o aumento do interesse nas arquiteturas de circuitos quânticos variacionais, especialmente no contexto de algoritmos híbridos quântico-clássicos, revela a necessidade de explorar e otimizar essas abordagens para aplicações práticas. A aprendizagem de máquina quântica, por sua vez, desponta como uma área de estudo com grande potencial para a aplicação de novas tecnologias e metodologias, tornando essencial a investigação e o desenvolvimento de soluções inovadoras nesse domínio.

Em seguida, será delineado o problema de pesquisa que este estudo busca resolver, seguido pela apresentação dos objetivos gerais e específicos que orientam a condução deste trabalho. Por fim, serão descritas de maneira sucinta a metodologia adotada para a investigação e a estrutura do documento, detalhando a organização dos capítulos subsequentes.

1.1 CONTEXTO E MOTIVAÇÃO

A computação quântica emergiu das predições da mecânica quântica formuladas no início do século XX, por volta da década de 1920, quando cientistas buscavam compreender fenômenos microscópicos que não podiam ser explicados pela física clássica. Essa nova perspectiva introduziu a ideia de que o universo possui uma capacidade de processamento de informação significativamente maior do que aparenta Ladd *et al.* (2010).

Os computadores quânticos utilizam qubits, que, devido ao fenômeno da superposição, podem existir em múltiplos estados simultaneamente. Isso possibilita a execução de cálculos

paralelos e a consideração de diversas possibilidades ao mesmo tempo, aumentando consideravelmente o poder computacional desses dispositivos em comparação aos computadores clássicos, que utilizam bits tradicionais representando 0 ou 1 Das (2024). Outro princípio essencial da computação quântica é o emaranhamento, que permite que qubits emaranhados compartilhem estados de forma intrínseca, independentemente da distância entre eles, facilitando um processamento de informações mais interconectado e eficiente Das (2024).

O verdadeiro potencial da computação quântica reside na utilização desses princípios para realizar cálculos em uma escala que supera a capacidade dos computadores clássicos em problemas específicos. Essa capacidade permite resolver problemas matemáticos complexos, otimizar sistemas e simular fenômenos quânticos, possibilitando avanços significativos em áreas como descoberta de medicamentos, ciência dos materiais e inteligência artificial Das (2024).

As aplicações da computação quântica se estendem por diversas indústrias, desde segurança e criptografia, onde os computadores quânticos podem potencialmente quebrar algoritmos criptográficos atualmente em uso, até otimização logística e financeira. A tecnologia quântica promete otimizar cadeias de suprimentos, transporte e planejamento, melhorando a utilização de recursos e reduzindo custos Das (2024). Em ciência dos materiais e descoberta de medicamentos, a simulação precisa de moléculas e materiais pode acelerar o desenvolvimento de novos produtos. Além disso, a computação quântica tem o potencial de transformar a inteligência artificial e o aprendizado de máquina, permitindo a análise rápida de grandes volumes de dados e a execução de algoritmos complexos de otimização Das (2024).

Atualmente, a computação quântica se encontra na era NISQ, *Noisy Intermediate-Scale Quantum*, um termo introduzido por John Preskill para descrever o estágio intermediário em que a tecnologia se encontra Preskill (2018). Neste cenário, temos computadores quânticos com dezenas a algumas centenas de qubits, superando o que pode ser simulado pelos supercomputadores clássicos mais poderosos. No entanto, esses dispositivos ainda enfrentam grandes desafios para alcançar uma computação quântica verdadeiramente escalável e tolerante a falhas.

Uma das abordagens mais promissoras na era NISQ são os *Variational Quantum Algorithms* (VQAs), que se beneficiam de uma arquitetura híbrida quântico-clássica. Essa abordagem requer menos qubits e portas quânticas, permitindo que os algoritmos lidem com as limitações dos computadores quânticos atuais, que estão sujeitos a ruído e têm um número restrito de qubits disponíveis Schillo & Sturm (2024). Um exemplo relevante de VQAs é o *Quantum Circuit Learning* (QCL), que consiste em duas partes principais: a codificação dos dados e uma camada parametrizada que pode ser treinada. Este esquema implementa uma função de modelo quântico que pode ser ajustada ao problema específico em questão Schillo & Sturm (2024).

Estudos anteriores já demonstraram a efetividade desses circuitos em simulações clássicas Mitarai *et al.* (2018); Hatakeyama-Sato *et al.* (2023). Além disso, a aplicação desses circuitos para resolver equações diferenciais também foi analisada e simulada por métodos clássicos Paine *et al.* (2023); Kyriienko *et al.* (2021). Apesar dos avanços na computação quântica, sua aplicação prática ainda enfrenta desafios significativos, como limitações de equipamento e a exploração

relativamente limitada de algoritmos quânticos em comparação com o desenvolvimento da computação clássica. A elaboração manual de circuitos quânticos é uma tarefa complexa e demorada, exigindo um profundo conhecimento das interações quânticas, do emaranhamento e das propriedades dos qubits e portas quânticas disponíveis.

Em resposta a essas dificuldades, a busca automatizada de circuitos quânticos (*Quantum Architecture Search* - QAS) surge como uma solução promissora. O QAS concentra-se na geração automatizada de circuitos quânticos, facilitando a criação de PQCs, *Parametrized Quantum Circuits*, otimizados sem a necessidade de um projeto manual intensivo Martyniuk *et al.* (2024). Essa abordagem não apenas acelera o processo de design, mas também permite a exploração de configurações de circuitos que poderiam não ser consideradas por um projetista humano, contribuindo para o avanço da computação quântica na era NISQ.

1.2 OBJETIVOS

O objetivo principal deste projeto é desenvolver um Algoritmo Genético (AG) capaz de gerar arquiteturas quânticas variacionais ideais, independentemente do problema específico a ser resolvido. A solução proposta será guiada por uma função métrica que descreve os critérios necessários para a obtenção da melhor solução. Com base nessa função de qualidade, o algoritmo evoluirá as arquiteturas quânticas possíveis, visando encontrar a configuração mais adequada para cada problema. Para alcançar esse objetivo geral, os seguintes objetivos específicos devem ser atingidos:

1. Definir e formular o problema de otimização, que servirá como referência para validar experimentalmente a técnica proposta.
2. Modelar o Algoritmo Genético, desenvolvendo uma função de qualidade que permitirá avaliar as arquiteturas quânticas geradas.
3. Criar e implementar algoritmos auxiliares que garantam a execução eficiente do AG.
4. Automatizar a criação e simulação de circuitos quânticos variacionais, assegurando a eficácia e eficiência do processo.
5. Conduzir uma análise quantitativa dos resultados gerados pelo AG, permitindo a avaliação de seu desempenho e eficácia.

1.3 CONTRIBUIÇÕES DO TRABALHO

As principais contribuições deste trabalho, alinhadas aos objetivos definidos, podem ser resumidas da seguinte forma:

1. **Desenvolvimento de Algoritmo Genético para Arquiteturas Quânticas:** Criação de um algoritmo inovador capaz de gerar automaticamente arquiteturas quânticas variacionais ideais, contribuindo para o avanço do design de circuitos quânticos.
2. **Definição do Modelo de Problema de Otimização:** Formulação clara de um problema de otimização que valida experimentalmente a técnica proposta, contribuindo para a aplicação prática de algoritmos quânticos em problemas reais e sintéticos.
3. **Função de Qualidade para Avaliação:** Desenvolvimento de uma função de qualidade robusta para avaliar as soluções geradas pelo AG, oferecendo uma métrica precisa para comparação e melhoria das arquiteturas quânticas.
4. **Integração de Algoritmos Auxiliares:** Implementação de algoritmos auxiliares para garantir a modularidade e escalabilidade do sistema, facilitando a execução eficiente do AG.
5. **Automatização da Simulação de Circuitos Quânticos:** Desenvolvimento de um sistema automatizado que simplifica a criação e simulação de circuitos quânticos variacionais, acelerando o processo de experimentação e pesquisa na computação quântica.
6. **Análise Quantitativa dos Resultados:** Realização de uma análise quantitativa detalhada dos resultados obtidos pelo algoritmo genético, fornecendo percepções sobre seu desempenho e possíveis aplicações práticas.
7. **Contribuição para a Comunidade de Pesquisa:** A documentação e divulgação dos métodos e resultados contribuirão para o avanço da pesquisa em computação quântica, incentivando outros pesquisadores a explorar e aprimorar novas abordagens e técnicas.

1.4 ORGANIZAÇÃO DO DOCUMENTO

Este projeto está organizado em capítulos conforme descrito a seguir:

- **Capítulo 2** - Apresenta os fundamentos teóricos do estudo, estabelecendo a conceituação dos principais elementos necessários para o entendimento da proposta deste projeto de conclusão de curso;
- **Capítulo 3** - Discute os estudos relacionados à busca de arquiteturas quânticas no contexto da inteligência artificial, utilizando algoritmos genéticos;
- **Capítulo 4** - Descreve a construção do projeto, abordando os conjuntos de dados utilizados, os procedimentos de tratamento desses dados e a modelagem do algoritmo genético projetado;

- **Capítulo 5** - Apresenta a metodologia experimental utilizada, os resultados obtidos e uma discussão detalhada sobre eles;
- **Capítulo 6** - Oferece uma visão geral dos resultados da pesquisa, com considerações finais e sugestões para trabalhos futuros.

2

FUNDAMENTAÇÃO

Esta seção apresenta uma visão geral dos conceitos fundamentais de computação quântica, circuitos variacionais quânticos, algoritmos genéticos e otimização por gradiente, conforme aplicados no contexto do projeto em questão.

2.1 COMPUTAÇÃO QUÂNTICA

A computação quântica é um campo emergente da ciência da computação que explora as leis da mecânica quântica para realizar cálculos de maneiras que seriam impossíveis ou impraticáveis para os computadores clássicos. A computação quântica difere da computação clássica em termos dos princípios fundamentais e da maneira como os dados são processados.

2.1.1 Qubits e Superposição

Um *qubit* é a unidade básica de informação na computação quântica, análogo ao bit clássico. Matematicamente, um qubit é representado como um vetor em um espaço vetorial de duas dimensões, conhecido como espaço de Hilbert $\mathbb{C}^2$. Os estados básicos de um qubit são $|0\rangle$ e $|1\rangle$, que formam uma base ortonormal desse espaço.

Um estado geral de um qubit pode ser escrito como uma *superposição* desses dois estados básicos, como mostrado na equação (2.1):

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{2.1}$$

onde α e β são números complexos que satisfazem a condição de normalização:

$$|\alpha|^2 + |\beta|^2 = 1 \tag{2.2}$$

Essa equação (2.2) expressa que o qubit pode estar em ambos os estados $|0\rangle$ e $|1\rangle$ simultaneamente, uma propriedade essencial da mecânica quântica. As probabilidades de medir o qubit em $|0\rangle$ ou $|1\rangle$ são dadas por $|\alpha|^2$ e $|\beta|^2$, respectivamente.

Além disso, um qubit pode ser visualizado geometricamente por meio da *esfera de Bloch*, uma representação tridimensional em que qualquer estado quântico puro pode ser representado

como um ponto na superfície da esfera. Na esfera de Bloch, os estados $|0\rangle$ e $|1\rangle$ correspondem aos polos norte e sul, respectivamente, como ilustrado na Figura 1.

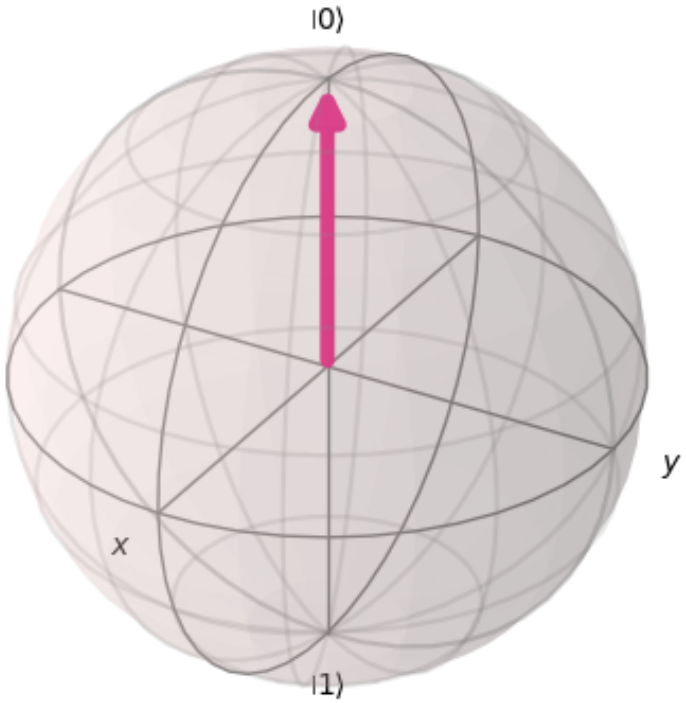


Figura 1: Esfera de Bloch apontando para o estado $|0\rangle$.

Qualquer estado de superposição de um qubit pode ser descrito por dois ângulos: o *ângulo polar* θ, que define a inclinação em relação ao eixo z, e o *ângulo azimutal* ϕ, que descreve a rotação em torno desse eixo. A representação matemática do estado quântico é:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle \tag{2.3}$$

Aqui, $0 \leq \theta \leq \pi$ controla a superposição entre os estados $|0\rangle$ e $|1\rangle$, enquanto $0 \leq \phi < 2\pi$ determina a fase relativa entre os dois estados. Essa representação compacta permite entender tanto a amplitude quanto a fase do estado quântico de maneira visual e intuitiva. Nielsen & Chuang (2010)

2.1.2 Portas Lógicas Quânticas

As portas lógicas quânticas são o equivalente quântico das portas lógicas clássicas, mas, em vez de operar sobre bits clássicos, elas atuam sobre qubits. Cada porta lógica quântica é representada por um operador unitário que age sobre um ou mais qubits Nielsen & Chuang (2010). A seguir, discutimos algumas das principais portas lógicas quânticas.

2.1.2.1 *Porta NOT Quântica (X)*

A porta NOT quântica, conhecida como porta X, atua de maneira análoga à porta NOT clássica, invertendo o estado de um qubit. Se o qubit estiver no estado $|0\rangle$, após a aplicação da porta X, ele será convertido no estado $|1\rangle$, e vice-versa. Sua representação matricial é dada na equação (2.4) Nielsen & Chuang (2010):

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{2.4}$$

Essa porta troca os estados $|0\rangle$ e $|1\rangle$. A Figura 2 mostra o símbolo da porta X em um circuito quântico típico.

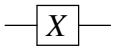


Figura 2: Porta NOT quântica (X) aplicada a um qubit.

2.1.2.2 *Porta Hadamard (H)*

A porta Hadamard, denotada por H, é uma das operações fundamentais da computação quântica, ao transformar o estado de um qubit em uma superposição equitativa dos estados de base $|0\rangle$ e $|1\rangle$. Sua representação matricial é expressa pela equação (2.5) Nielsen & Chuang (2010):

Quando aplicada ao estado $|0\rangle$, a porta Hadamard resulta na superposição descrita pela equação (2.6):

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{2.5}$$

$$H|0\rangle = \frac{1}{\sqrt{2}} \left(|0\rangle + |1\rangle\right) \tag{2.6}$$

Essa transformação é de extrema relevância em algoritmos quânticos, uma vez que habilita um qubit a alcançar uma superposição linear de estados, sendo um componente essencial em vários protocolos quânticos Nielsen & Chuang (2010). A Figura 3 apresenta o símbolo da porta Hadamard no contexto de um circuito quântico típico.

Figura 3: Símbolo da porta Hadamard aplicada a um qubit.

2.1.2.3 *Porta CNOT (Controlled-NOT)*

A porta CNOT (Controlled-NOT) é uma operação quântica de dois qubits, onde um qubit atua como controle e o outro como alvo. A operação consiste em aplicar a porta X (NOT) no qubit alvo se, e somente se, o qubit de controle estiver no estado $|1\rangle$. A matriz que representa a operação CNOT é dada pela equação (2.7) Nielsen & Chuang (2010):

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.7}$$

A porta CNOT é crucial para a criação de emaranhamento entre qubits, sendo um dos elementos centrais de muitos algoritmos quânticos, incluindo o algoritmo de Shor Shor (1997). A Figura 4 ilustra a implementação da porta CNOT em um circuito quântico, com o qubit de controle e o qubit alvo explicitamente indicados.

Figura 4: Representação da porta CNOT em um circuito quântico, onde o primeiro qubit é o controle e o segundo é o alvo.

2.1.2.4 *Portas Parametrizadas: R_x, R_y, R_z*

As portas de rotação parametrizadas R_x, R_y, e R_z aplicam rotações arbitrárias em um qubit ao longo dos eixos x, y, e z da esfera de Bloch, respectivamente. Essas portas são essenciais em algoritmos variacionais e circuitos quânticos, pois permitem manipulações precisas do estado quântico Bergholm *et al.* (2018).

O parâmetro θ determina o ângulo de rotação e, portanto, o quanto o estado quântico é modificado em torno de um dos eixos escolhidos. Essas rotações desempenham um papel fundamental ao ajustar o vetor de estado do qubit, permitindo uma manipulação controlada no espaço de Hilbert. As matrizes de rotação são definidas da seguinte forma:

$$R_x(\theta) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{2.8}$$

$$R_y(\theta) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{2.9}$$

$$R_z(\theta) = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{pmatrix} \tag{2.10}$$

Essas operações podem ser visualizadas na esfera de Bloch, onde R_x e R_y modificam a posição do vetor de estado, enquanto R_z altera apenas a fase do estado quântico. A Figura 6 mostra a esfera de Bloch após rotações de 90 graus nos três eixos.

Na Figura 5, apresentamos as três operações aplicadas sequencialmente em um qubit dentro de um circuito quântico. Essas portas são amplamente utilizadas em algoritmos de otimização e aprendizado de máquina quântica, permitindo a manipulação eficiente dos estados quânticos.

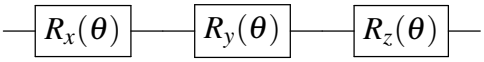
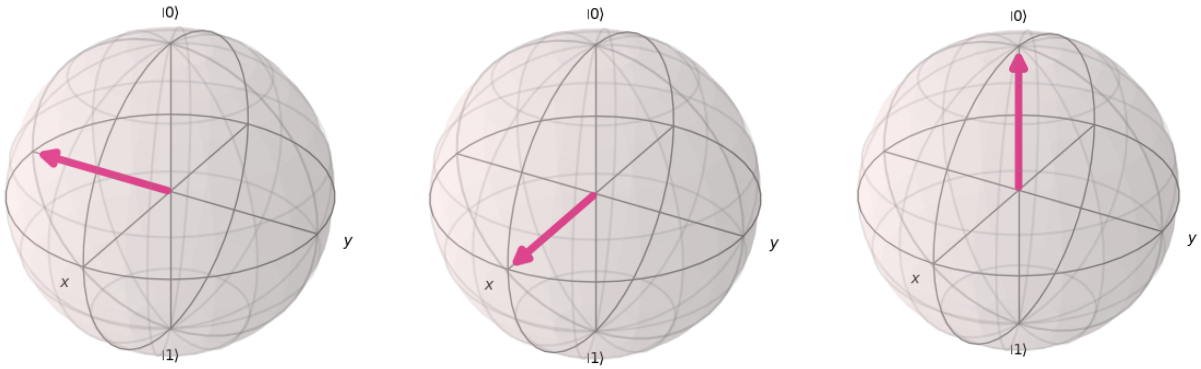


Figura 5: Representação das portas de rotação $R_x(\theta)$, $R_y(\theta)$, e $R_z(\theta)$ aplicadas sequencialmente em um qubit.



Rotação no eixo X (90 graus) Rotação no eixo Y (90 graus) Rotação no eixo Z (90 graus)

Figura 6: Esfera de Bloch após rotações de 90 graus nos eixos X, Y e Z. As rotações no eixo X e Y modificam a posição do vetor de estado na esfera de Bloch, enquanto a rotação no eixo Z altera apenas a fase do estado quântico.

2.1.2.5 Portas Controladas

2.1.2.6 Portas Controladas

As versões controladas das portas de rotação, como CR_x, CR_y, e CR_z, funcionam de maneira similar às portas controladas clássicas, como a $CNOT$. Nessas operações, um qubit de controle determina se a rotação parametrizada será aplicada ao qubit alvo. Por exemplo, a porta $CR_z(\theta)$ aplica a rotação $R_z(\theta)$ ao qubit alvo somente se o qubit de controle estiver no estado $|1\rangle$.

Essas portas controladas são fundamentais para a criação de emaranhamento entre qubits e para a implementação de transformações dependentes de múltiplos qubits, cruciais para algoritmos quânticos mais complexos. A Figura 7 apresenta a representação das portas CR_x, CR_y, e CR_z em um circuito quântico.

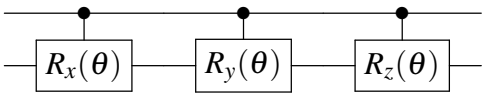


Figura 7: Representação das portas controladas $CR_x(\theta)$, $CR_y(\theta)$, e $CR_z(\theta)$ aplicadas em qubits. O qubit de controle determina a aplicação das rotações nos qubits-alvo.

2.1.3 Medição em Circuitos Quânticos

A computação quântica geralmente termina com a *medição* dos qubits. A medição colapsa o estado quântico de um qubit em um dos estados básicos, $|0\rangle$ ou $|1\rangle$, com probabilidades dependentes do estado final do circuito. Esse processo pode ser representado matematicamente por um conjunto de operadores de projeção que determinam a probabilidade de colapso de um qubit em um determinado estado Nielsen & Chuang (2010).

Se o estado de um qubit antes da medição for dado por:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{2.11}$$

onde α e β são números complexos que representam as amplitudes de probabilidade associadas aos estados $|0\rangle$ e $|1\rangle$, a medição pode ser descrita pelos operadores de projeção $P_0 = |0\rangle\langle 0|$ e $P_1 = |1\rangle\langle 1|$, que projetam o estado nos subespaços correspondentes a $|0\rangle$ e $|1\rangle$.

A probabilidade de colapsar no estado $|0\rangle$ ou $|1\rangle$ após a medição é dada pelas seguintes expressões:

$$P(0) = \langle\psi|P_0|\psi\rangle = |\alpha|^2, \tag{2.12}$$

$$P(1) = \langle\psi|P_1|\psi\rangle = |\beta|^2, \tag{2.13}$$

onde $P(0)$ e $P(1)$ representam as probabilidades de medir o qubit nos estados $|0\rangle$ e $|1\rangle$, respectivamente.

A Figura 8 ilustra o símbolo utilizado para representar o processo de medição em um circuito quântico, onde o estado do qubit é medido após a execução das operações quânticas.

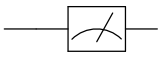


Figura 8: Símbolo de medição em um circuito quântico, representando o colapso do estado quântico em um dos estados básicos após a medição.

2.1.3.1 Operador Pauli-Z

Um dos observáveis mais utilizados na medição de qubits é o operador de Pauli-Z, que mede o estado quântico ao longo do eixo Z na esfera de Bloch. O operador Z é representado matricialmente pela Equação (2.14):

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{2.14}$$

Quando um qubit é medido no observável Z, o estado é projetado nos autovalores $+1$ (correspondente ao estado $|0\rangle$) ou -1 (correspondente ao estado $|1\rangle$). O observável Z é amplamente utilizado porque mede diretamente os estados de base computacional $|0\rangle$ e $|1\rangle$, que são os estados típicos de saída dos circuitos quânticos Nielsen & Chuang (2010).

2.1.4 Exemplo: Circuito de Teletransporte Quântico

O circuito de *teletransporte quântico* é um dos exemplos mais emblemáticos da computação quântica, permitindo a transferência do estado de um qubit para outro qubit distante sem a transmissão física da partícula. Esse processo é possível graças ao emaranhamento quântico e à comunicação clássica. O teletransporte quântico tem várias aplicações em redes de comunicação quântica e criptografia quântica Bennett *et al.* (1993).

2.1.4.1 Operações do circuito de teletransporte

O circuito de teletransporte quântico, conforme mostrado na Figura 9, consiste em três qubits:

- O **qubit 1** (denotado como $|\psi\rangle$) contém o estado quântico que será teletransportado.

- O **qubit 2** e o **qubit 3** são inicialmente emaranhados, formando um par de EPR (Einstein-Podolsky-Rosen).

As operações no circuito de teletransporte seguem as etapas abaixo:

1. **Criação do par emaranhado**: - O qubit 2 e o qubit 3 são preparados em um estado emaranhado. Primeiro, uma porta Hadamard é aplicada ao qubit 2, colocando-o em superposição. Em seguida, uma porta CNOT é aplicada entre o qubit 2 (controle) e o qubit 3 (alvo), criando o estado emaranhado entre eles, conforme a Equação (2.15).

$$|\psi_{23}\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{2.15}$$

2. **Interação entre o qubit 1 e o par emaranhado**: - O próximo passo é transferir a informação quântica do estado $|\psi\rangle$ (qubit 1) para o sistema composto pelos qubits 2 e 3. Uma porta CNOT é aplicada entre o qubit 1 (estado a ser teletransportado) e o qubit 2 (controle). Após a operação CNOT, uma porta Hadamard é aplicada ao qubit 1.

3. **Medição de qubits 1 e 2**: - Os qubits 1 e 2 são medidos em suas bases computacionais ($|0\rangle/|1\rangle$). Essas medições colapsam os estados dos qubits e geram dois bits clássicos, que são enviados ao qubit 3. O estado de $|\psi\rangle$ é representado pela Equação (2.16).

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{2.16}$$

4. **Correções condicionais**: - Dependendo dos resultados das medições dos qubits 1 e 2, operações corretivas são aplicadas ao qubit 3. Se o resultado da medição do qubit 1 for 1, uma porta X é aplicada ao qubit 3. Se o resultado do qubit 2 for 1, uma porta Z é aplicada ao qubit 3. Essas operações garantem que o estado original $|\psi\rangle$ seja corretamente transferido para o qubit 3.

2.1.4.2 Diagrama do Circuito

Na Figura 9, mostramos o diagrama do circuito de teletransporte quântico. As portas H, $CNOT$, X, e Z são aplicadas conforme descrito acima.

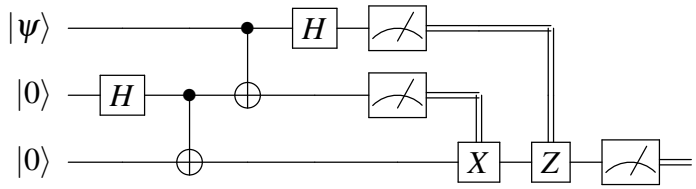


Figura 9: Circuito de teletransporte quântico Bennett *et al.* (1993). O estado do qubit $|\psi\rangle$ é transferido para o terceiro qubit através de emaranhamento e medições.

2.2 CIRCUITOS QUÂNTICOS VARIACIONAIS

No contexto da computação quântica, os **circuitos quânticos variacionais** (ou VQC, *Variational Quantum Circuits*) são um dos conceitos mais importantes que combinam computação quântica com métodos de otimização clássica. Esses circuitos são essenciais para superar as limitações atuais dos computadores quânticos ruidosos de escala intermediária (NISQ, *Noisy Intermediate-Scale Quantum*) Preskill (2018).

Um circuito quântico variacional é uma abordagem híbrida que utiliza tanto computadores quânticos quanto computadores clássicos. A ideia básica é construir um circuito quântico

parametrizado, com **portas unitárias** que dependem de um conjunto de parâmetros clássicos. Esses parâmetros são otimizados de maneira iterativa por um algoritmo clássico de otimização, que ajusta as portas do circuito quântico visando minimizar ou maximizar uma função de custo específica, como apresentada na Figura 10. Isso permite que problemas difíceis sejam resolvidos utilizando uma quantidade limitada de qubits e portas quânticas.

2.2.1 Estrutura de um Circuito Quântico Variacional

Um circuito quântico variacional é composto por três partes principais McClean *et al.* (2016):

1. **Preparação do Estado Inicial**: O circuito começa com a preparação de um estado quântico inicial $|\psi_{\text{inicial}}\rangle$, que pode ser um estado simples como $|0\rangle^{\otimes n}$ (todos os qubits no estado $|0\rangle$) ou um estado mais complexo.

2. **Camadas Parametrizadas**: O núcleo do circuito consiste em um conjunto de portas quânticas que dependem de parâmetros $\theta = \{\theta_1, \theta_2, \ldots, \theta_k\}$. Essas portas formam uma sequência de camadas quânticas, onde cada camada pode consistir em portas de um único qubit, como rotações $R_x(\theta)$, $R_y(\theta)$, ou $R_z(\theta)$, e também portas de dois qubits, como a porta CNOT, que induzem emaranhamento entre os qubits Schuld *et al.* (2019). Cada porta unitária parametrizada pode ser descrita matematicamente como:

$$U(\theta) = U_k(\theta_k)U_{k-1}(\theta_{k-1})\ldots U_1(\theta_1) \tag{2.17}$$

onde $U(\theta)$ é o operador unitário total que depende de todos os parâmetros θ_i.

3. **Medição e Otimização Clássica**: Após a execução do circuito quântico, os qubits são medidos, e o resultado da medição é utilizado para calcular uma **função de custo** $C(\theta)$. Essa função de custo é geralmente uma expectativa de um operador observável O, como a energia de um sistema quântico ou o valor esperado de um operador de Pauli. A função de custo é dada por:

$$C(\theta) = \langle\psi(\theta)|O|\psi(\theta)\rangle \tag{2.18}$$

A função de custo é então minimizada por um algoritmo de otimização clássica, como o gradiente descendente ou o método de Nelder-Mead Cerezo *et al.* (2021). O objetivo é ajustar os parâmetros θ para encontrar o valor mínimo (ou máximo) da função de custo, o que representa a solução do problema.

2.2.2 Aplicações dos Circuitos Quânticos Variacionais

Os circuitos quânticos variacionais são usados em uma ampla gama de aplicações na computação quântica, incluindo:

- **VQE (Variational Quantum Eigensolver)**: Um dos algoritmos mais proeminentes baseados em circuitos quânticos variacionais é o **VQE**. Esse algoritmo é utilizado para encontrar o

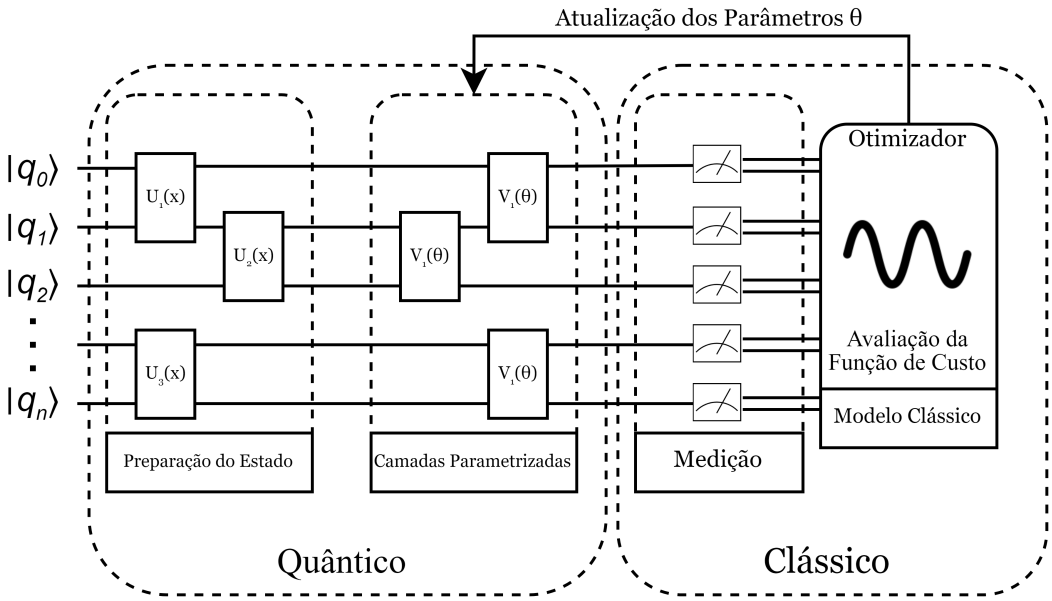


Figura 10: Representação esquemática de um circuito quântico variacional. Inclui a implementação iterativa dos componentes quânticos e clássicos do circuito.

estado fundamental de um sistema quântico ao minimizar a energia esperada de um Hamiltoniano quântico. Ele é especialmente útil em simulações de química quântica e problemas de física de muitos corpos Peruzzo *et al.* (2014).

- **QAOA (Quantum Approximate Optimization Algorithm)**: O QAOA é outro algoritmo importante baseado em circuitos quânticos variacionais. Ele é projetado para resolver problemas de otimização combinatória, utilizando uma sequência parametrizada de operações quânticas e um processo de otimização clássica para aproximar a solução do problema Farhi *et al.* (2014).

- **Classificadores Quânticos Variacionais**: Em aprendizado de máquina quântico, os circuitos quânticos variacionais podem ser usados como classificadores quânticos, onde os parâmetros do circuito são otimizados para classificar dados quânticos ou clássicos Schuld & Petruccione (2019).

2.2.3 Vantagens e Desafios

Os circuitos quânticos variacionais apresentam várias vantagens em comparação com outras abordagens quânticas e clássicas Cerezo *et al.* (2021):

- **Redução de Ruído**: Como os VQCs requerem apenas a execução de circuitos relativamente curtos e a dependência da otimização clássica, eles podem ser implementados em computadores quânticos NISQ, que são suscetíveis ao ruído e à decoerência.

- **Adaptabilidade**: Eles podem ser adaptados para uma ampla variedade de problemas, desde química quântica até aprendizado de máquina e otimização combinatória.

No entanto, também existem desafios significativos:

- **Barren Plateaus**: Um dos principais problemas é a presença de *barren plateaus*, regiões da paisagem da função de custo onde o gradiente é muito pequeno, tornando a otimização extremamente difícil McClean *et al.* (2018).

- **Número de Parâmetros**: A quantidade de parâmetros a ser ajustada pode crescer rapidamente com o número de qubits e camadas, levando a uma complexidade computacional crescente no processo de otimização clássica.

2.3 ALGORITMOS GENÉTICOS

Os **algoritmos genéticos (AG)** são uma técnica de otimização inspirada nos processos de evolução natural, como a **seleção natural** e a **reprodução** Holland (1992). Eles foram introduzidos por John Holland em 1975 como uma maneira de abordar problemas complexos de busca e otimização. Os algoritmos genéticos são particularmente úteis em situações onde o espaço de busca é muito grande ou mal definido, e onde os métodos tradicionais de otimização falham em encontrar soluções adequadas.

O funcionamento dos AG baseia-se na simulação do processo evolutivo. Um **conjunto inicial de soluções** (ou população) é criado aleatoriamente, e essas soluções são avaliadas por uma função de avaliação, também chamada de **função de aptidão**. A função de aptidão mede a qualidade de cada solução em relação ao problema sendo resolvido. Ao longo de várias iterações, ou **gerações**, a população de soluções é modificada por meio de operações como **seleção**, **crossover** (recombinação) e **mutação**, imitando os mecanismos biológicos de reprodução e evolução.

2.3.1 Componentes de um Algoritmo Genético

Os algoritmos genéticos têm três componentes principais Goldberg (1989a):

1. **Codificação (ou Representação)**: Cada solução potencial (indivíduo) é representada como um **cromossomo**, que pode ser uma cadeia de bits, números, ou outros tipos de dados. A escolha da representação é crucial para o desempenho do AG.

2. **Operadores Genéticos**: O comportamento evolutivo dos algoritmos genéticos depende de três operadores principais:

- **Seleção**: Seleciona os indivíduos mais aptos da população atual para serem pais da próxima geração. Diversos métodos de seleção podem ser usados, como a **roleta** ou a **seleção por torneio** Mitchell (1998).

- **Crossover (ou Recombinação)**: Combina partes dos cromossomos de dois pais para gerar novos indivíduos (filhos). O crossover permite a troca de informações genéticas entre soluções, permitindo que características vantajosas de diferentes soluções sejam combinadas.

- **Mutação**: Introduz pequenas mudanças aleatórias nos cromossomos para manter a diversidade genética da população e evitar a convergência prematura para soluções subótimas.

3. **Função de Avaliação (ou Função de Aptidão)**: Avalia cada solução com base na sua qualidade em resolver o problema. A função de aptidão orienta a evolução, permitindo que as soluções mais promissoras sejam selecionadas para a próxima geração.

2.3.2 Fluxo de um Algoritmo Genético

O fluxo básico de um algoritmo genético pode ser descrito da seguinte forma Holland (1992):

1. **Inicialização**: Cria-se uma população inicial de soluções aleatórias.
2. **Avaliação**: Cada indivíduo na população é avaliado pela função de aptidão.
3. **Seleção**: Os indivíduos mais aptos são selecionados para reprodução.
4. **Crossover e Mutação**: Novos indivíduos são gerados por crossover e mutação.
5. **Substituição**: A nova geração substitui a antiga.
6. **Terminação**: O processo se repete até que uma condição de parada seja atingida (por exemplo, número máximo de gerações ou uma solução satisfatória).

Esse processo é ilustrado na Figura 11:

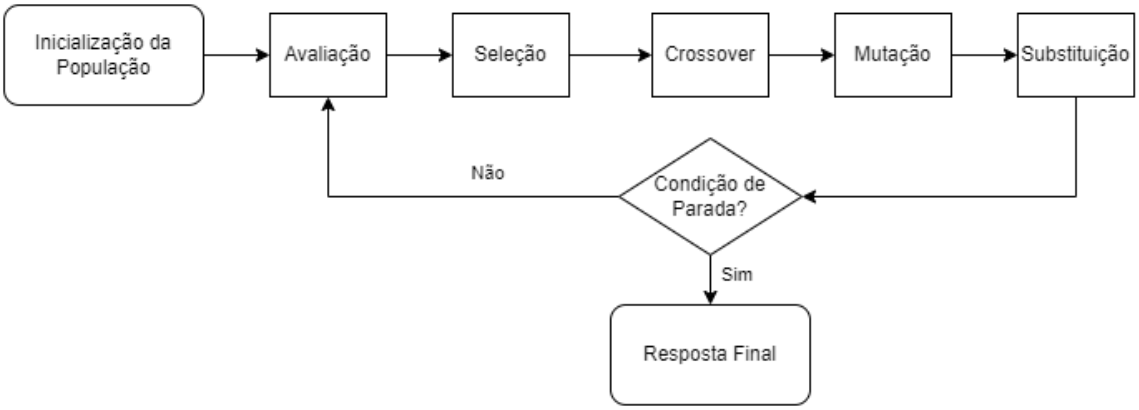


Figura 11: Fluxo de um algoritmo genético Holland (1992)

2.3.3 Aplicações dos Algoritmos Genéticos

Os algoritmos genéticos têm sido aplicados com sucesso em uma ampla variedade de áreas, incluindo Goldberg (1989a); Mitchell (1998):

- **Otimização**: Os AG são amplamente utilizados em problemas de otimização, onde o objetivo é maximizar ou minimizar uma função de custo.

- **Aprendizado de Máquina**: Os AG têm sido usados para otimizar modelos de aprendizado de máquina, como redes neurais, ajustando seus parâmetros.

- **Planejamento e Agendamento**: Em problemas de planejamento e agendamento, os AG são úteis para encontrar soluções eficientes e viáveis.

- **Evolução de Autômatos Celulares**: Os algoritmos genéticos também são usados em sistemas de autômatos celulares para encontrar regras ótimas para comportamentos específicos.

2.3.4 Vantagens e Desafios

Os algoritmos genéticos oferecem diversas vantagens Holland (1992); Goldberg (1989a):

- **Robustez**: Eles são robustos e podem ser aplicados em problemas altamente não lineares, com múltiplos picos e vales na paisagem de busca.

- **Exploração Global**: AGs são eficientes na exploração global do espaço de busca, o que os torna úteis para evitar mínimos locais.

Por outro lado, também existem desafios Mitchell (1998):

- **Convergência Prematura**: Um dos principais problemas é a convergência prematura, onde a população converte-se rapidamente para uma solução subótima.

- **Custo Computacional**: Os AGs podem ser computacionalmente caros, especialmente se o número de indivíduos e gerações for grande.

2.4 OTIMIZAÇÃO POR GRADIENTE EM CIRCUITOS VARIACIONAIS QUÂNTICOS

Em circuitos variacionais quânticos, a otimização dos parâmetros clássicos, como os ângulos de rotação das portas quânticas, é crucial para alcançar o objetivo do algoritmo quântico. Esses parâmetros controlam a evolução do estado quântico, e sua otimização visa minimizar uma função de custo associada ao problema sendo resolvido, como a energia de um sistema físico ou a acurácia de um modelo de aprendizado de máquina.

O método mais comumente utilizado para essa otimização é o **gradiente descendente**, que ajusta os parâmetros θ de maneira iterativa com base nas derivadas parciais da função de custo $C(\theta)$ em relação a cada parâmetro.

2.4.1 Gradiente Descendente

O gradiente descendente busca minimizar a função de custo $C(\theta)$, que depende dos parâmetros do circuito variacional. A ideia básica é ajustar os parâmetros na direção oposta ao gradiente da função de custo, ou seja, na direção em que a função de custo diminui mais rapidamente. A atualização de cada parâmetro θ_j é realizada de acordo com a seguinte equação:

$$\theta_j := \theta_j - \alpha \frac{\partial C(\theta)}{\partial \theta_j} \tag{2.19}$$

Onde:

- α é a taxa de aprendizado, que determina o tamanho do passo de cada atualização;

- $\frac{\partial C(\theta)}{\partial \theta_j}$ é o gradiente da função de custo em relação ao parâmetro θ_j.

O valor do gradiente $\frac{\partial C(\theta)}{\partial \theta_j}$ indica a inclinação da função de custo $C(\theta)$ em relação ao parâmetro θ_j. Se o gradiente for positivo, o parâmetro é reduzido; se for negativo, o parâmetro é aumentado. O processo continua até que a função de custo alcance um valor mínimo.

2.4.2 Cálculo do Gradiente: Regra do Deslocamento de Parâmetro

Em circuitos quânticos, calcular diretamente o gradiente de uma função de custo é um desafio, já que a função depende de medições probabilísticas no final do circuito quântico. Para contornar isso, a técnica mais utilizada é a **Regra do Deslocamento de Parâmetro** (*Parameter-Shift Rule*), que permite calcular o gradiente sem a necessidade de derivações numéricas aproximadas.

Essa regra requer duas execuções adicionais do circuito para cada parâmetro θ_j, deslocando o valor do parâmetro em $\pm\frac{\pi}{2}$. O gradiente é então calculado da seguinte forma:

$$\frac{\partial C(\theta)}{\partial \theta_j} = \frac{C(\theta_j + \frac{\pi}{2}) - C(\theta_j - \frac{\pi}{2})}{2} \tag{2.20}$$

Essa abordagem permite estimar o gradiente de maneira eficiente, mesmo em dispositivos quânticos com ruído. Além disso, ela é exata para funções de custo baseadas em portas unitárias parametrizadas.

2.4.3 Variantes do Gradiente Descendente

Existem algumas variantes do gradiente descendente que podem ser aplicadas na otimização dos parâmetros em circuitos variacionais quânticos:

2.4.3.0.1 Gradiente Descendente Estocástico (SGD): No gradiente descendente estocástico, o gradiente é estimado com base em um pequeno subconjunto de medições (mini-batch) em vez de utilizar todas as medições. Isso torna o processo mais rápido, mas também mais ruidoso. Apesar do ruído, o SGD pode ser útil em problemas de otimização de grande escala, como em algoritmos de aprendizado de máquina quântica.

2.4.3.0.2 Gradiente Descendente com Momento: Essa variante adiciona um termo de "memória"que acumula o gradiente das iterações anteriores, ajudando o algoritmo a escapar de mínimos locais e acelerando a convergência em superfícies de custo complexas. Ele é útil quando o espaço de parâmetros é difícil de navegar devido à presença de "barren plateaus", regiões onde o gradiente é muito pequeno para guiar a otimização.

2.4.4 Desafios na Otimização

A otimização de circuitos variacionais quânticos enfrenta desafios únicos. Um dos maiores problemas é a presença de *barren plateaus*, regiões da paisagem da função de custo onde o gradiente se aproxima de zero, tornando a otimização extremamente lenta. Quanto maior o número de qubits e portas no circuito, mais provável é que esses *barren plateaus* apareçam McClean *et al.* (2018).

Além disso, a escolha da taxa de aprendizado α é um fator crítico. Se a taxa for muito alta, o algoritmo pode saltar sobre o mínimo da função de custo, enquanto uma taxa muito baixa pode fazer com que o processo de otimização seja extremamente lento. O ajuste fino desse parâmetro é, portanto, essencial para o sucesso do processo de otimização.

3

TRABALHOS RELACIONADOS

Nesta seção, revisamos os trabalhos mais relevantes que utilizam algoritmos genéticos e evolutivos para a busca de arquiteturas de circuitos quânticos variacionais (VQCs). VQCs têm se mostrado promissores na execução de algoritmos quânticos em dispositivos NISQ (*Noisy Intermediate-Scale Quantum*), mas encontrar a arquitetura ideal de circuitos ainda é um desafio significativo. Diversos trabalhos exploram o uso de algoritmos evolutivos para otimizar as arquiteturas de VQCs, com foco na redução de profundidade do circuito, aumento da precisão e robustez contra ruído.

3.1 BUSCA DE ARQUITETURAS PARAMETRIZADAS

O artigo Ding & Spector (2022) apresenta um esquema de busca de arquitetura quântica baseado em algoritmos evolutivos, chamado EQAS-PQC, focado em otimizar o layout de circuitos quânticos parametrizados em ambientes de aprendizagem por reforço (RL). O objetivo é melhorar a eficiência da busca de arquiteturas quânticas ao longo de interações contínuas com o ambiente de RL, onde o agente aprende a selecionar arquiteturas que maximizam o desempenho da tarefa, como encontrar a melhor configuração de parâmetros quânticos.

3.2 OTIMIZAÇÃO MULTIOBJETIVO DE CIRCUITOS QUÂNTICOS

O artigo Chivilikhin *et al.* (2020) propõe o **MoG-VQE**, um algoritmo multiobjetivo que otimiza simultaneamente a topologia do circuito e os ângulos de rotação de qubits, utilizando o NSGA-II para a otimização da estrutura do *ansatz* e o CMA-ES para ajustar os ângulos. O objetivo é minimizar o número de portas CNOT e alcançar precisão química em simulações de moléculas, como H_2, H_4, H_6, BeH_2 e LiH.

3.3 ROBUSTEZ E EFICIÊNCIA DE CIRCUITOS VQA

O artigo Huang *et al.* (2022) propõe um algoritmo evolutivo com comprimento de genoma ajustável para o design de circuitos VQA (*Variational Quantum Algorithms*) robustos

e eficientes em termos de recursos. O foco está na otimização tanto da estrutura do *ansatz* do circuito quanto dos parâmetros das portas, sem assumir uma estrutura ou profundidade fixa do circuito. O algoritmo foi aplicado a modelos típicos de erro em VQA, como o cálculo da energia fundamental de moléculas como hidrogênio e água, bem como no modelo de Heisenberg.

3.4 APLICAÇÕES EM QUÍMICA QUÂNTICA

O trabalho Rattew *et al.* (2020) propõe o **EVQE** (*Evolutionary Variational Quantum Eigensolver*), um algoritmo evolutivo aplicado ao VQE visando reduzir o valor de um Hamiltoniano, o que é fundamental para determinar o estado fundamental de sistemas moleculares. Os testes do algoritmo foram conduzidos em aplicações de química quântica, como a simulação de moléculas. O EVQE busca otimizar o número de portas quânticas e a profundidade do circuito, considerando o ambiente ruidoso dos dispositivos NISQ.

3.5 NEUROEVOLUÇÃO QUÂNTICA

O estudo de Lu *et al.* (2021) propõe o **MQNE** (*Markovian Quantum Neuroevolution*), um algoritmo neuroevolutivo que constrói redes neurais quânticas otimizadas via processos evolutivos. Uma característica importante desse trabalho é que os circuitos quânticos são modelados como *Grafos de Markov*, onde as transições entre estados são usadas para evoluir as arquiteturas dos circuitos. O objetivo é construir redes neurais quânticas precisas e eficientes, aplicadas a tarefas de aprendizado de máquina, como o reconhecimento de padrões e a classificação de dados. O algoritmo busca arquiteturas que maximizam a acurácia das tarefas enquanto minimizam o custo computacional.

3.6 MAPAS DE CARACTERÍSTICAS QUÂNTICAS

O artigo Altares-López *et al.* (2021) apresenta um método evolutivo para o design automático de mapas de características quânticas (*quantum feature maps*) aplicados a máquinas de vetor de suporte quântico (QSVM). O objetivo é otimizar tanto a estrutura quanto os parâmetros dos mapas de características quânticas, melhorando o desempenho das máquinas de vetor de suporte quântico em tarefas de classificação, como o reconhecimento de padrões em grandes conjuntos de dados.

3.7 BUSCA DE ARQUITETURAS EM APRENDIZADO DE MÁQUINA QUÂNTICO

O artigo Zhang & Zhao (2022) apresenta o **EQAS** (*Evolutionary-based Quantum Architecture Search*), um esquema evolutivo para a busca de arquiteturas quânticas visando equilibrar

o poder expressivo e a capacidade de treinamento em circuitos quânticos na era dos dispositivos NISQ. O EQAS foi aplicado para tarefas de classificação em aprendizado de máquina quântico.

3.8 DISCUSSÃO FINAL SOBRE A LITERATURA REVISADA

A proposta desta dissertação distingue-se das soluções presentes no estado da arte por sua abordagem inovadora na busca pela arquitetura de circuitos. O Circuito Variacional Quântico é otimizado não apenas em relação às portas lógicas utilizadas no processamento dos dados, mas também na forma como esses dados são carregados no circuito. Diferentemente de outros métodos, não se assume previamente um esquema específico para o carregamento de dados. Em vez disso, o circuito aprende, de maneira autônoma, a forma mais eficiente de realizar essa tarefa com as portas disponíveis. Esse processo de otimização do carregamento de dados ocorre em paralelo ao processamento quântico, tornando a solução mais flexível e adaptativa. O algoritmo empregado é um algoritmo genético, adaptado especificamente para resolver o problema proposto, assegurando uma busca eficiente por arquiteturas otimizadas.

Artigo	Tipo de Aplicação	Objetivo
Ding & Spector (2022)	Otimização em Aprendizagem por Reforço	Melhorar a eficiência da arquitetura quântica com aprendizado de reforço, maximizando o desempenho da tarefa
Chivilikhin *et al.* (2020)	Simulação de moléculas	Otimizar a topologia e ângulos do circuito, reduzindo o número de portas CNOT enquanto atinge precisão química
Huang *et al.* (2022)	Sistemas Quânticos	Gerar circuitos de baixa profundidade e resistentes ao ruído, otimizando a estrutura do *ansatz* e parâmetros de portas
Rattew *et al.* (2020)	Redução de Hamiltoniano em aplicações de química quântica	Minimizar o valor do Hamiltoniano em sistemas moleculares, otimizando a execução do VQE em ambientes ruidosos
Lu *et al.* (2021)	Redes neurais quânticas modeladas como Grafos de Markov	Modelar e evoluir redes neurais quânticas através de Grafos de Markov para melhorar a acurácia e reduzir o custo computacional
Altares-López *et al.* (2021)	Máquinas de Vetor de Suporte Quântico (QSVM)	Otimizar mapas de características quânticas para melhorar o desempenho das QSVM em tarefas de classificação
Zhang & Zhao (2022)	Classificação em aprendizado de máquina quântico	Buscar arquiteturas quânticas ótimas com menos portas parametrizadas e maior eficiência no aprendizado
Proposta	Circuitos Variacionais Quânticos	Otimizar circuitos quânticos variacionais, bem como o carregamento e processamento dos dados.

Tabela 1: Resumo dos trabalhos relacionados sobre algoritmos evolutivos para busca de arquiteturas de circuitos quânticos variacionais.

4

PROPOSTA

Esta seção descreve os componentes utilizados na construção da solução proposta, dividida em duas partes principais: os conjuntos de dados e o algoritmo genético.

4.1 CONJUNTO DE DADOS

Com o intuito de avaliar o desempenho do algoritmo de busca em diferentes cenários, foram utilizados dois grupos de conjuntos de dados: o primeiro consiste em dados sintéticos bidimensionais, que permitem a visualização em gráficos de dispersão 2D, e o segundo é composto por dados de problemas reais, utilizados para avaliar o desempenho do modelo em cenários mais realistas e complexos.

4.1.1 Dados Sintéticos

Para a criação dos dados, foi utilizada a biblioteca *scikit-learn* Pedregosa *et al.* (2011) juntamente com a linguagem de programação *Python* Python Software Foundation (2023). Foram gerados, ao todo, cinco conjuntos de dados, cada um contendo 200 elementos, sendo 100 de cada classe. Todos os problemas são de classificação binária. Os conjuntos de dados mencionados estão ilustrados na Figura 12.

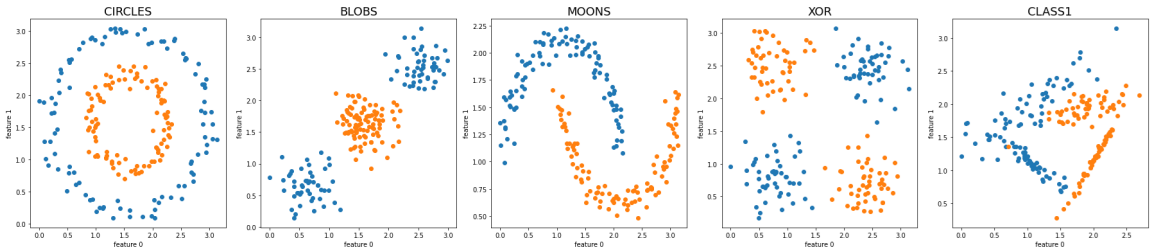


Figura 12: Conjuntos de dados sintéticos criados para a avaliação do modelo.

4.1.2 Dados Reais

Foram selecionados seis conjuntos de dados reais disponíveis no Repositório de Aprendizado de Máquina da UC Irvine Dua & Graff (2019), cada um representando um problema distinto.

4.1.2.1 Iris

O conjunto de dados *Iris* é um dos mais amplamente utilizados em problemas de aprendizado de máquina, especialmente em tarefas de classificação. Ele consiste em 150 amostras de flores da espécie *Iris*, distribuídas igualmente entre três classes: *Iris setosa*, *Iris versicolor* e *Iris virginica*. Cada amostra é descrita por quatro atributos contínuos: comprimento da sépala, largura da sépala, comprimento da pétala e largura da pétala, todos medidos em centímetros. O objetivo é classificar corretamente a espécie da flor com base nesses atributos. O conjunto de dados é frequentemente utilizado como benchmark para testar algoritmos de aprendizado supervisionado, particularmente em problemas de classificação multiclasse. Uma ilustração das três classes da flor pode ser vista na Figura 13.

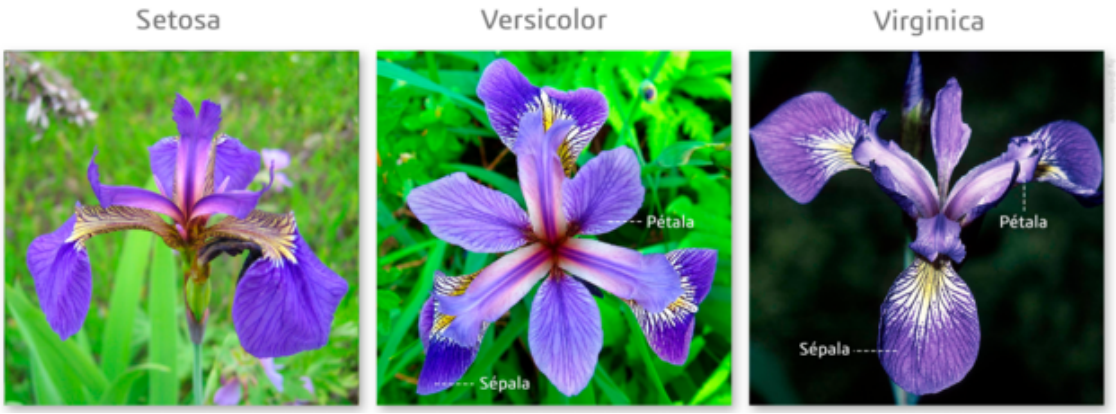


Figura 13: Ilustração das três classes de flores presentes no conjunto de dados *Iris*.

4.1.2.2 Palmer Penguins

O conjunto de dados *Palmer Penguins* é utilizado para tarefas de classificação e regressão em problemas de aprendizado de máquina. Ele contém informações sobre três espécies de pinguins (Adelie, Gentoo e Chinstrap) coletadas nas ilhas Palmer, na Antártica. O conjunto de dados é composto por 344 amostras e oito variáveis descritivas, que incluem: o comprimento do bico, a profundidade do bico, o comprimento da nadadeira, o peso corporal. Uma ilustração das três espécies de pinguins pode ser vista na Figura 14.

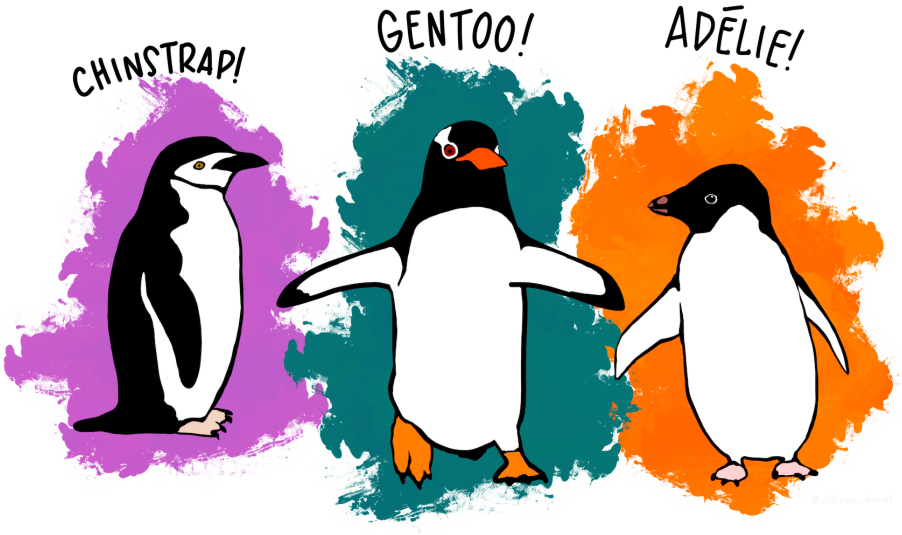


Figura 14: Ilustração das três espécies de pinguins presentes no conjunto de dados *Palmer Penguins*.

4.1.2.3 Banknote Authentication

O conjunto de dados *Banknote Authentication* é amplamente utilizado em problemas de classificação binária. Ele contém informações extraídas de imagens digitalizadas de cédulas de dinheiro genuínas e falsificadas, visando identificar a autenticidade das notas. As características foram obtidas a partir de imagens em escala de cinza, capturadas por uma câmera industrial, sendo descritas por quatro variáveis principais: variância da transformada de *wavelet*, simetria, curtose e entropia. O conjunto de dados é composto por 1.372 amostras, onde cada instância é rotulada como cédula genuína ou falsificada.

O objetivo desse conjunto de dados é desenvolver modelos de aprendizado supervisionado que possam classificar corretamente as cédulas em autênticas ou falsas, com base nas características fornecidas.

4.1.2.4 Blood Transfusion Service Center

O conjunto de dados *Blood Transfusion Service Center* foi coletado do Banco de Sangue de Taiwan e é utilizado em problemas de classificação binária. O objetivo do conjunto de dados é prever se um doador de sangue fará uma doação dentro de um determinado período após a última doação. Ele é composto por 748 amostras, cada uma representando um doador de sangue, e quatro variáveis descritivas contínuas: frequência (número total de doações), volume de sangue doado (constante para todas as amostras), primeira doação (meses desde a primeira doação) e meses desde a última doação. A variável alvo é binária, indicando se o doador realizou uma nova doação nos últimos seis meses (valor 1) ou não (valor 0). Este conjunto de dados é amplamente utilizado para o treinamento e avaliação de modelos de classificação em aprendizado de máquina,

especialmente em problemas de previsão de comportamento de doadores.

4.1.2.5 Heart Disease Cleveland

O conjunto de dados *Heart Disease Cleveland* é amplamente utilizado em problemas de classificação em aprendizado de máquina, visando prever a presença de doenças cardíacas em pacientes com base em uma série de atributos clínicos. Este conjunto de dados contém 303 amostras, cada uma representando um paciente, e 13 variáveis descritivas, que incluem: idade, sexo, pressão arterial em repouso, colesterol sérico, níveis de açúcar no sangue em jejum, resultados de eletrocardiograma em repouso, frequência cardíaca máxima alcançada, angina induzida por exercício, depressão do segmento ST, inclinação do segmento ST, número de vasos principais e resultados do teste de talassemia.

A variável alvo é categórica e originalmente contém cinco valores (0 a 4), que indicam a severidade da doença cardíaca. Este conjunto de dados é amplamente utilizado como benchmark para avaliar o desempenho de modelos de aprendizado supervisionado, especialmente em problemas de saúde e diagnóstico médico.

4.1.2.6 Breast Cancer Wisconsin

O conjunto de dados *Breast Cancer Wisconsin (Original)* é amplamente utilizado para problemas de classificação binária em aprendizado de máquina, visando prever se uma amostra de células mamárias é benigna ou maligna. O conjunto de dados contém 699 amostras, cada uma representando uma instância de células extraídas por aspiração com agulha fina. Existem 10 variáveis descritivas, que incluem: tamanho do núcleo, forma do núcleo, adesão marginal, tamanho celular, epitélio nu, cromatina suave, nucléolos normais e mitoses. Cada uma dessas variáveis é avaliada em uma escala de 1 a 10, com base nas características observadas nas células.

A variável alvo é binária, onde 2 representa uma amostra benigna e 4 representa uma amostra maligna. Este conjunto de dados é amplamente utilizado como benchmark para modelos de aprendizado supervisionado, particularmente em contextos de diagnóstico médico e predição de câncer.

4.1.3 Tratamento dos Conjuntos de Dados

A abordagem adotada para o carregamento dos dados nos circuitos variacionais quânticos utiliza rotações elementares R_x, R_y e R_z. Para que os dados sejam corretamente inseridos nesses operadores, é necessário que eles estejam dentro do intervalo $(0, \pi)$. Para alcançar esse objetivo, foi aplicada a técnica de normalização *min-max* Jiawei Han & Pei (2011).

Além disso, o circuito quântico utilizado retorna valores dentro do intervalo $[-1, 1]$. Assim, para comparar a saída do circuito com as classes-alvo, todas as classes precisam ser mapeadas de maneira igualitária para esse intervalo, ou seja, cada classe deve ocupar o mesmo

espaço dentro do intervalo. Para isso, foi desenvolvido um algoritmo de *threshold* que realiza essa divisão.

4.1.3.1 Algoritmo de Threshold

O algoritmo `threshold` divide o intervalo contínuo $[-1,1]$ em n subintervalos de tamanho uniforme e atribui uma classe central a um valor de entrada x. Inicialmente, o tamanho de cada subintervalo é calculado como interval $= \frac{2}{n}$. O algoritmo então verifica se x está nos extremos do intervalo, retornando -1 para valores próximos a -1 e 1 para valores próximos a 1. Caso contrário, x é classificado de acordo com o subintervalo correspondente, sendo o valor central da classe calculado como classe $=$ prev $+ \frac{\text{interval}}{2}$. A função retorna o valor da classe para o subintervalo onde x se encontra, garantindo uma divisão igualitária e precisa do intervalo, conforme apresentado no Algoritmo 1.

Algoritmo 1: Algoritmo de classificação usando `threshold`

Result: Classificação de x em subintervalos de $[-1,1]$ com n intervalos

```
Inicialize o número de intervalos n e o valor de entrada x;
Calcule o tamanho do intervalo: interval = 2/n;
Defina o valor inicial prev = −1 + interval;
for cada i de 0 até n − 1 do
    if x ≤ −1 + interval then
        Retorne -1;
    if x ≥ 1 − interval then
        Retorne 1;
    Calcule o valor da classe: classe = prev + interval/2;
    Atualize prev+ = interval;
    if x ≤ prev then
        Retorne classe;
```

4.2 ALGORITMO GENÉTICO

A confecção do algoritmo genético pode ser dividida em duas partes principais: a modelagem dos indivíduos e a modelagem da evolução da população.

4.2.1 Modelagem dos Indivíduos

4.2.1.1 Codificação e Decodificação do Genoma

Primeiramente, foram definidos os níveis de abstração para modelar a informação genética, onde o indivíduo é representado por um genoma. O genoma contém diversos cromossomos

que correspondem às portas unitárias do circuito, e cada cromossomo é formado por genes representados como números binários, conforme mostrado na Tabela 2.

Nível de Abstração	O que representa
Genoma	Indivíduo
Cromossomo	Um operador unitário
Gene	Um número binário

Tabela 2: Informação genética

A etapa inicial da modelagem do algoritmo genético consistiu na codificação e decodificação da informação genética. Cada gene foi representado como um número binário, mapeando-se as portas quânticas unitárias para os valores possíveis desses genes, como ilustrado na Tabela 3. Como circuitos quânticos de apenas 1 qubit não utilizam portas controladas, foi necessário criar duas tabelas de decodificação distintas. Para circuitos de 1 qubit, os espaços não utilizados na tabela foram preenchidos com portas identidade. Já para circuitos com 2 ou mais qubits, as portas parametrizáveis foram representadas de forma redundante, pois elas são responsáveis tanto pelo mapeamento dos dados de entrada quanto pelo processamento. Novamente, os espaços restantes foram completados com operadores identidade para assegurar a integridade da tabela.

1 Qubit	
Binário	**Porta Quântica**
000	Identidade
001	Identidade
010	Identidade
011	Hadamard
100	Pauli_X
101	R_X
110	R_Y
111	R_Z

2 ou mais Qubits	
Binário	**Porta Quântica**
0000	Identidade
0001	Identidade
0010	Identidade
0011	Identidade
0100	Hadamard
0101	Pauli_X
0110	C_{NOT}
0111	C_Y
1000	C_Z
1001	C_H
1010	R_X
1011	R_Y
1100	R_Z
1101	R_X
1110	R_Y
1111	R_Z

Tabela 3: Decodificação de números binários em portas quânticas para 1 qubit e 2 ou mais qubits, organizadas lado a lado

O cromossomo deve conter informações sobre o *embedding*, que é o processo de inserção de dados clássicos no circuito quântico por meio de portas parametrizadas. As variáveis descritivas são inseridas sequencialmente, com a possibilidade de repetição. Para representar essa informação, utiliza-se um bit adicional. Em circuitos de 2 qubits, um bit extra identifica o

qubit de controle, e em circuitos de 3 qubits, são usados 2 bits adicionais, conforme ilustrado na Figura 15.

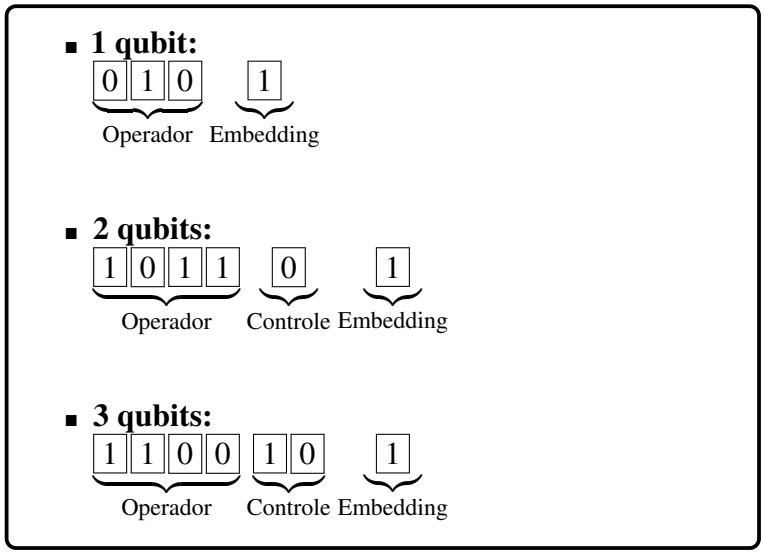


Figura 15: Representação genética de circuitos quânticos para 1, 2 e 3 qubits, com os bits de controle e embedding incluídos.

Os operadores quânticos descritos pelos cromossomos são aplicados sequencialmente, conforme a posição de controle estabelecida, que informa a posição na qual o operador estará. Em um circuito de três qubits, os operadores controlados são direcionados ao qubit imediatamente subsequente, exceto o terceiro qubit, que controla o primeiro. Esse arranjo resulta em uma estrutura cíclica de controle entre os qubits do circuito.

Para finalizar os circuitos, emprega-se o operador de observação **Z**. Em circuitos com múltiplos qubits, realiza-se o produto tensorial dos observáveis individuais para determinar a medição resultante.

4.2.1.2 Exemplo de Decoficação do Genoma

Um exemplo do funcionamento da decodificação pode ser observado na Figura 15, onde um genoma com 42 genes é representado. Ao interpretar esses genes como um circuito de 3 qubits, identificamos 6 cromossomos. Seguindo a Tabela 4, podemos determinar as portas correspondentes a cada cromossomo. A informação de controle é utilizada para descrever a posição das portas no circuito, e, no caso das portas paramétricas, o último gene do cromossomo indica se o parâmetro será treinável ou não. Nesse contexto, o parâmetro treinável é denotado por θ_n, enquanto f_n representa um atributo de entrada do conjunto de dados. Toda essa decodificação está exemplificada na Tabela 4, com o circuito resultante da decodificação apresentado na Figura 16.

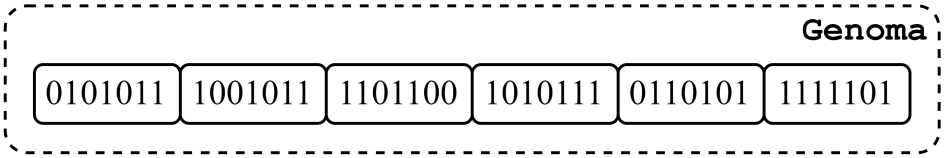


Figura 16: Um exemplo de genoma para um circuito com 3 qubits.

Cromossomo	**Porta**	**Posição (Controle)**	***Embedding* (Carregamento)**
0101 01 1	Pauli_x	1	Não se aplica
1001 01 1	C_H	1	Não se aplica
1101 10 0	R_x	2	Não
1010 11 1	R_x	0	Sim
0110 10 1	C_{NOT}	2	Não se aplica
1111 10 1	R_z	2	Sim

Tabela 4: Tabela de Decodificação

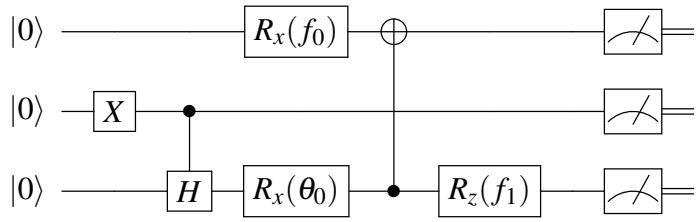


Figura 17: O circuito quântico de 3 qubits gerado após a decodificação do genoma apresentado na Figura 16.

4.2.1.3 Operador de Mutação

Iniciamos o operador resgatando um número aleatório de uma distribuição lognormal com média 0 e desvio padrão 1. Como a mutação deve sempre ocorrer, se o valor retornado dessa distribuição for menor ou igual a 1, uma mutação é realizada. Utilizando essa distribuição, conseguimos perturbar mais de um bit do gene com uma baixa probabilidade de ocorrência, conforme descrito na Tabela 5.

Após determinar quantos bits serão mutados, utilizamos uma distribuição uniforme sobre todo o gene para selecionar as n posições de mutação. Nessas posições, os bits serão invertidos, ou seja, os valores 1 serão alterados para 0 e os valores 0 para 1.

4.2.1.4 Operador de Crossover

Considerando que o gene é composto por n unidades genéticas, ou cromossomos, que contêm a informação necessária para posicionar uma porta específica em uma determinada posição, o operador de *crossover* tem a função de gerar um novo indivíduo a partir de dois

Tabela 5: Probabilidade de mutação de genes em uma distribuição lognormal com média 0 e desvio padrão 1

Genes Mutados	Probabilidade (%)
1	100.0
2	24.4
3	13.6
4	8.2
5	5.4
6	3.6
7	2.6
8	1.9
9	1.4
10	1.0

indivíduos distintos. O funcionamento desse operador pode ser observado no Algoritmo 2.

Algoritmo 2: Algoritmo de Crossover

Result: Gera o gene do filho a partir dos cromossomos da mãe e do pai
Entrada : Mãe, Pai
Saída : Gene do Filho

$n \leftarrow$ quantidade de cromossomos;
$gene\_filho \leftarrow [\,]$;
for $i \leftarrow 0$ **to** $n-1$ **do**
    $r \leftarrow$ número aleatório (0 ou 1) ;
    if $r = 0$ **then**
        $gene\_filho$.add(cromossomo pai$_i$);
    else
        $gene\_filho$.add(cromossomo mãe$_i$);
return $gene\_filho$

4.2.1.5 *Função de Fitness*

Para determinar o valor de aptidão, ou *fitness*, o modelo descrito pela informação genética será treinado três vezes, cada uma utilizando estados aleatórios distintos e predeterminados, incluindo os estados dos otimizadores e as divisões dos conjuntos de dados. Cada indivíduo será consistentemente avaliado com esses três mesmos estados aleatórios. O treinamento será realizado utilizando o otimizador de Nesterov Nesterov (2018), em conjunto com a função de custo MSE (*mean squared error*) Friedman *et al.* (2001), com um limite máximo de 100 iterações e um tamanho de *batch* de 5. O conjunto de dados será dividido em 75% para treinamento e 25%

para teste. Ao final de cada rodada de treinamento, serão registrados o custo e a acurácia no conjunto de teste.

Ao final das três rodadas, teremos três valores de acurácia e custo, os quais serão processados conforme a equação 4.1.

$$\text{Fitness} = \frac{1}{3}\left(\left(\frac{\text{acuracia}_1}{\text{custo}_1}\right)^2 + \left(\frac{\text{acuracia}_2}{\text{custo}_2}\right)^2 + \left(\frac{\text{acuracia}_3}{\text{custo}_3}\right)^2\right) \tag{4.1}$$

O objetivo dessa fórmula de *fitness* é selecionar modelos que apresentem menores custos e maiores acurácias.

4.2.2 Modelagem da Evolução da População

A evolução da população é controlada por um conjunto de hiperparâmetros que devem ser definidos antes da execução do algoritmo.

- **População Inicial**: Determina o número de indivíduos na população inicial (população 0). No presente projeto, o valor utilizado foi 100.
- **Tamanho da População**: Define o número de indivíduos existentes na população em um determinado momento. No projeto, o valor utilizado foi 25.
- **Tamanho de Continuação**: Determina a quantidade de indivíduos da população anterior que irão sobreviver para a próxima geração. O valor utilizado no projeto foi 15.
- **Tamanho da Prole**: Estabelece o número de filhos que serão gerados para a próxima geração. Neste projeto, o valor utilizado foi 10.
- **Taxa de Mutação**: Define a probabilidade de ocorrência de mutações. No projeto, a taxa de mutação utilizada foi de $\frac{1}{25}$.
- **Geração Máxima**: Determina o número máximo de gerações do algoritmo. No projeto, o valor utilizado foi 100.
- **Tempo de Espera**: Define o número máximo de gerações sem melhora que o algoritmo espera até que ocorra um evento de hiper mutação. No projeto, o valor utilizado foi 10.
- **Tolerância**: Estabelece o número máximo de eventos de hiper mutação consecutivos permitidos. Caso esse valor seja ultrapassado, o algoritmo é interrompido. No projeto, o valor utilizado foi 1.

O algoritmo completo pode ser visto em 3, ele possui um evento de hipermutação e uma seleção via roleta.

4.2.2.1 *Evento de Hipermutação*

O evento de hipermutação ocorre quando a quantidade de gerações máxima de espera para uma resposta global melhor surgir é ultrapassado, nele todos os sobreviventes da geração passada sofrem mutação. O intuito desse mecanismo é dispersar a população de resposta o suficiente para que a população como um toda consiga sair de um máximo local e ir em direção ao máximo global.

4.2.2.2 *Método de Roleta*

O método de roleta Goldberg (1989b) é utilizado para selecionar indivíduos em um algoritmo genético com base em suas aptidões relativas. A probabilidade P_i de um indivíduo i ser selecionado é proporcional ao seu valor de *fitness*, conforme mostrado na Equação 4.2:

$$P_i = \frac{\text{fitness}_i}{\sum_{j=1}^{N} \text{fitness}_j} \tag{4.2}$$

Nesse método, cada indivíduo ocupa uma porção da "roda"proporcional à sua aptidão. A roleta é girada repetidamente, selecionando indivíduos para a próxima geração. Essa técnica garante que indivíduos mais aptos tenham maior probabilidade de serem escolhidos, mas ainda permite que indivíduos menos aptos também sejam selecionados, preservando a diversidade genética.

Algoritmo 3: Algoritmo Genético

Result: Melhor indivíduo encontrado após a evolução
Entrada : População Inicial, Tamanho da População, Tamanho de Continuação, Taxa de Mutação, Tolerância, Tempo de Espera, Tamanho da Prole, Número de Gerações
Saída : Melhor indivíduo (*melhorGlobal*)

```
1  população ← [novo individuo × População Inicial] ;
2  população ← roleta(Tamanho da População);
3  população.sort();
4  melhorGlobal ← população[0];
5  contador1 ← 0;
6  contador2 ← 0;
7  for gen em gerações do
8      contador1 ← contador1 + 1;
9      if população[0].fitness > melhorGlobal.fitness then
10         melhorGlobal ← população[0];
11         contador1 ← 0;
12         contador2 ← 0;
13     if contador2 > tolerância then
14         break;
15     sobreviventes ← roleta(Tamanho de Continuação);
16     if contador1 > tempo de espera then
17         contador1 ← 0;
18         contador2 ← contador2 + 1;
19         nova_geração ← [sobrevivente.mutação(100%) para sobrevivente em
             sobreviventes];
20     else
21         nova_geração ← [sobrevivente.mutação(Taxa de Mutação) para sobrevivente
             em sobreviventes];
22     for _ em Tamanho da Prole do
23         (mãe, pai) ← roleta(2);
24         filho ← crossover(mãe, pai);
25         nova_geração.add(filho.mutação(Taxa de Mutação));
26     população ← nova_geração;
27 return melhorGlobal
```

5

RESULTADOS

Este capítulo apresenta os resultados obtidos ao longo deste trabalho. Primeiramente, são detalhadas as diferentes configurações experimentais utilizadas durante a implementação do algoritmo genético, conforme descrito na Seção 5.1. Posteriormente, na Seção 5.2, são discutidos os resultados numéricos alcançados e comparações com a literatura.

5.1 METODOLOGIA EXPERIMENTAL

O algoritmo genético foi avaliado em diferentes configurações de execução, variando o número de qubits (1, 2 e 3) e a quantidade de cromossomos, que variou entre 5 e 50, de acordo com as especificidades de cada conjunto de dados. Para garantir a robustez dos resultados, cada configuração foi repetida cinco vezes. As combinações de parâmetros testadas estão detalhadas na Tabela 6.

Tabela 6: Configurações utilizadas por conjunto de dados no algoritmo genético

Conjunto de Dados	**Nº de Qubits**	**Nº de Cromossomos**
Circles	1, 2 e 3	5, 10, 15 e 20
Blobs	1, 2 e 3	5, 10, 15 e 20
Moons	1, 2 e 3	5, 10, 15 e 20
XOR	1, 2 e 3	5, 10, 15 e 20
Class1	1, 2 e 3	5, 10, 15 e 20
Iris	1, 2 e 3	10 e 20
Penguins	1, 2 e 3	10 e 20
Banknote	1, 2 e 3	10 e 20
Transfusion	1, 2 e 3	10 e 20
Heart	1, 2 e 3	30 e 50
Breast Cancer	1, 2 e 3	30 e 50

A quantidade de cromossomos foi ajustada de acordo com o número de *features* de cada conjunto de dados. Para um conjunto de dados com *n features*, o número mínimo de portas

paramétricas necessárias para o carregamento e processamento de informação é $n+1$.

5.2 RESULTADOS NUMÉRICOS

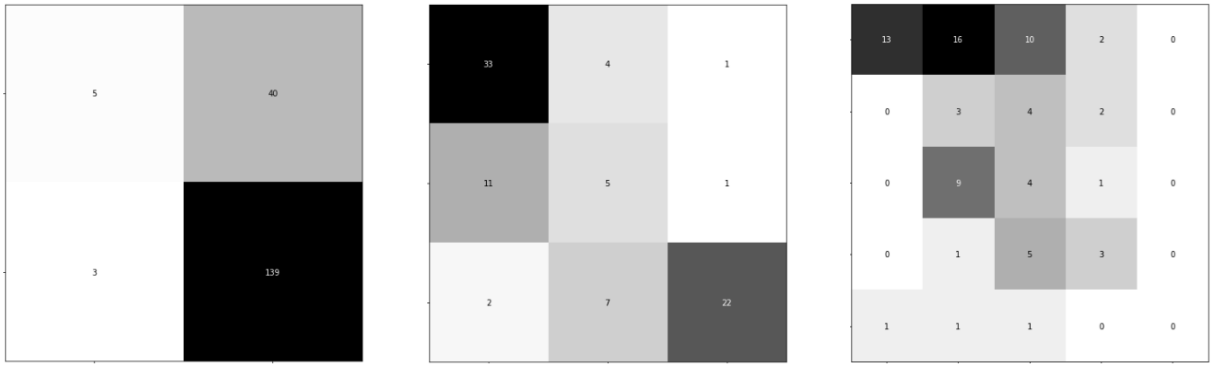
5.2.1 Análise dos Conjuntos de Dados Reais

Os valores médios de acurácia e seus respectivos desvios-padrão obtidos para os conjuntos de dados reais estão apresentados nas Tabelas 7. Observa-se que modelos que não possuíam portas parametrizáveis suficientes para processar adequadamente os dados foram considerados inválidos, sendo-lhes atribuídos valores de aptidão e acurácia iguais a 0. Esse comportamento justifica os baixos desempenhos observados, especialmente no conjunto de dados *Heart Disease Cleveland*, na configuração com 30 cromossomos.

Outro fator importante a ser considerado é o desbalanceamento das classes em alguns dos conjuntos de dados. No conjunto de dados *Palmer Penguins*, por exemplo, uma das classes corresponde a 44% das respostas possíveis, enquanto no conjunto *Blood Transfusion Service Center*, 76% das saídas pertencem a uma única classe. No caso do conjunto de dados *Heart Disease Cleveland*, 54% das classes estão concentradas em uma única resposta. Esse desbalanceamento pode levar os modelos a ficarem presos em máximos locais, o que é evidenciado pelas matrizes de confusão na Figura 18, que mostram a dificuldade dos modelos em separar adequadamente as classes menos representadas.

Tabela 7: Resultados para os diferentes conjuntos de dados reais com diferentes configurações de circuito. As arquiteturas que demonstraram os melhores resultados, sendo também as mais simples, estão destacadas em negrito.

Conjuntos de dados reais				
Conjunto de Dados	**Nº Cromossomos**	**1 Qubit**	**2 Qubits**	**3 Qubits**
Iris	10	97.0 ± 1.31	96.0 ± 1.04	96.3 ± 0.88
	20	98.1 ± 0.63	**98.9 ± 0.36**	98.2 ± 0.54
Penguins	10	50.0 ± 0.0	50.0 ± 0.0	49.7 ± 0.64
	20	**64.6 ± 0.66**	64.2 ± 0.12	61.3 ± 5.68
Banknote	10	97.5 ± 0.25	95.4 ± 2.05	85.0 ± 10.76
	20	**98.0 ± 0.41**	97.5 ± 0.98	96.7 ± 1.33
Transfusion	10	77.0 ± 0.13	76.9 ± 0.1	76.8 ± 0.08
	20	**77.0 ± 0.0**	77.0 ± 0.08	76.9 ± 0.1
Heart	30	24.8 ± 20.29	14.3 ± 17.91	6.3 ± 12.64
	50	**42.3 ± 1.72**	40.9 ± 0.59	40.5 ± 0.44
Breast Cancer	30	93.2 ± 2.93	95.6 ± 0.52	95.2 ± 0.59
	50	94.5 ± 0.99	96.0 ± 1.64	**96.1 ± 0.81**



(a) *Transfusion*. (b) *Penguins*. (c) *Heart Disease*.

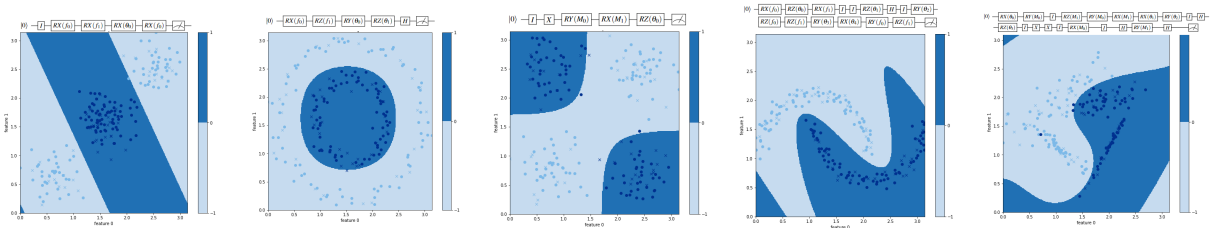
Figura 18: Matrizes de Confusão dos três conjuntos de dados diferentes: *Transfusion*, *Penguins*, e *Heart Disease*.

5.2.2 Análise dos Conjuntos de Dados Sintéticos

Nos conjuntos de dados sintéticos, os modelos que apresentaram o melhor desempenho tendem a ser aqueles com arquiteturas mais simples e menor capacidade de processamento, conforme ilustrado na Tabela 10. O desempenho superior desses modelos pode ser atribuído ao aumento exponencial do espaço de busca à medida que novos qubits são adicionados. Para um único qubit, existem 12 configurações possíveis para um cromossomo; esse número aumenta para 40 e 60 configurações em circuitos com 2 e 3 qubits, respectivamente. Além disso, a quantidade de portas distintas e as possíveis posições dessas portas no circuito contribuem significativamente para o crescimento do espaço de busca, como evidenciado pela Tabela 8.

Circuito	**Quantidade de Portas**	**Posições Distintas**	***Embedding***
1 qubit	6	1	2
2 qubits	10	2	2
3 qubits	10	3	2

Tabela 8: Configurações distintas de um cromossomo



(a) Blobs (b) Circles (c) XOR (d) Moons (e) Class1

Figura 19: Mapas de decisão do circuito quântico descrito. As bolinhas representam os dados de treino e os "X"indicam os dados de teste. As regiões sombreadas indicam as diferentes classes previstas pelo modelo, onde a intensidade da cor reflete a confiança na decisão do modelo.

Outro fator relevante é a especialização dos modelos gerados pelo algoritmo genético.

Tabela 9: Resultados para os diferentes conjuntos de dados sintéticos com diferentes configurações de circuito. As arquiteturas que demonstraram os melhores resultados, sendo também as mais simples, estão destacadas em negrito.

Conjuntos de dados sintéticos				
Conjunto de Dados	**Nº Cromossomos**	**1 Qubit**	**2 Qubits**	**3 Qubits**
Circles	5	**100.0 ± 0.0**	84.7 ± 12.51	85.3 ± 18.9
	10	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0
	15	100.0 ± 0.0	100.0 ± 0.0	99.9 ± 0.28
	20	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0
Blobs	5	99.4 ± 0.28	99.1 ± 0.29	89.7 ± 18.88
	10	**100.0 ± 0.0**	99.7 ± 0.34	99.9 ± 0.28
	15	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0
	20	100.0 ± 0.0	100.0 ± 0.0	99.9 ± 0.28
Moons	5	86.1 ± 1.15	84.3 ± 1.51	84.1 ± 0.28
	10	92.9 ± 2.23	86.4 ± 1.6	89.1 ± 3.0
	15	97.8 ± 0.78	94.7 ± 2.87	92.8 ± 3.19
	20	**98.2 ± 0.9**	97.8 ± 0.78	95.5 ± 4.13
Xor	5	**100.0 ± 0.0**	99.9 ± 0.28	100.0 ± 0.0
	10	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0
	15	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0
	20	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0
Class1	5	91.1 ± 0.29	87.4 ± 0.28	88.9 ± 1.53
	10	94.8 ± 1.35	94.3 ± 1.57	91.7 ± 1.89
	15	95.7 ± 0.89	95.5 ± 0.66	94.1 ± 0.79
	20	**96.9 ± 0.67**	95.7 ± 0.68	95.2 ± 0.48

Esse algoritmo ajusta a arquitetura do circuito conforme as características do conjunto de dados, permitindo que os modelos se adaptem às propriedades específicas do conjunto de treinamento. Como mostrado na Figura 19, circuitos com características distintas são gerados para a solução de cada problema, refletindo a capacidade do algoritmo em produzir arquiteturas especializadas para diferentes contextos.

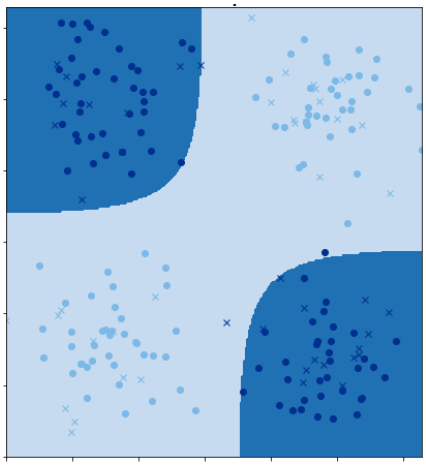
5.3 FENÔMENOS OBSERVADOS NO ALGORITMO GENÉTICO

Durante o processo de busca, observou-se a ocorrência de dois fenômenos importantes que afetam diretamente a desempenho do algoritmo. O primeiro fenômeno é a **convergência de diferentes circuitos em um mesmo mapa de decisão**, o que indica que múltiplas arquiteturas, mesmo que geneticamente distintas, podem chegar à mesma solução de classificação. Esse comportamento é evidenciado na Figura 20. Isso ocorre porque o algoritmo genético é agnóstico em relação à composição genética dos modelos, considerando apenas o fenótipo. Tal fenômeno é comum em problemas de otimização onde múltiplas soluções ótimas são possíveis.

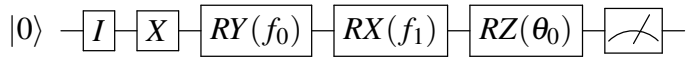
Tabela 10: Lista dos melhores modelos gerais por conjunto de dados

Conjunto de Dados	**Acurácia Média (%)**	**Modelo**
Circles	100.0	1 qubit, 5 cromossomos
Blobs	100.0	1 qubit, 5 cromossomos
Moons	99.3	1 qubit, 15 cromossomos
XOR	100.0	1 qubit, 5 cromossomos
Class1	98.0	1 qubit, 20 cromossomos
Iris	99.1	1 qubit, 20 cromossomos
Penguins	65.5	1 qubit, 20 cromossomos
Banknote	99.1	2 qubits, 20 cromossomos
Transfusion	77.2	1 qubit, 10 cromossomos
Heart	45.6	1 qubit, 50 cromossomos
Breast Cancer	97.3	2 qubits, 50 cromossomos

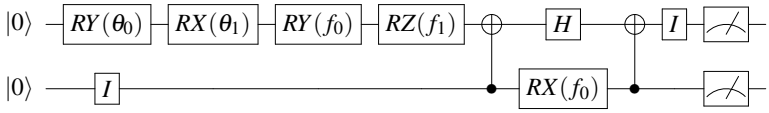
O segundo fenômeno observado é a **capacidade dos modelos de interpretar o mesmo conjunto de dados de diferentes maneiras**. Como demonstrado na Figura 21, os modelos podem gerar diferentes mapas de decisão para o mesmo conjunto de dados, o que pode ser explorado para a criação de modelos *ensemble*. Esses modelos combinam múltiplas interpretações dos dados para gerar uma solução menos enviesada e mais robusta, ao utilizar a diversidade de interpretações como vantagem para alcançar melhores resultados.



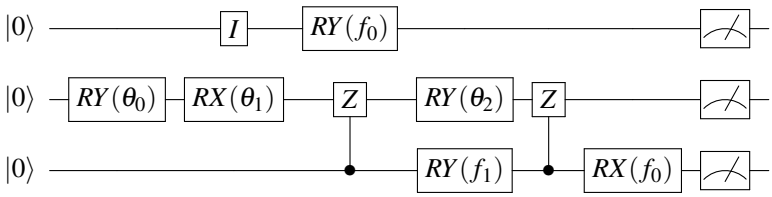
(a) Mapa de Decisão gerado pelos três circuitos quânticos.



(b) Circuito Quântico 1.

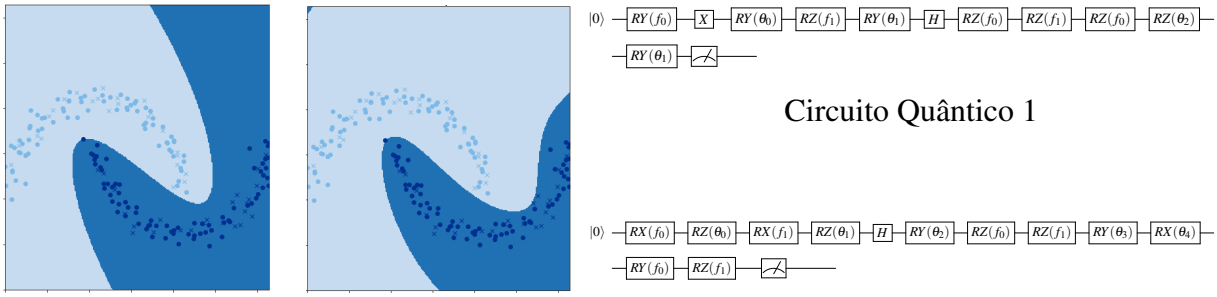


(c) Circuito Quântico 2.



(d) Circuito Quântico 3.

Figura 20: Mapa de Decisão e três circuitos quânticos diferentes usados para gerar o mesmo mapa de decisão.

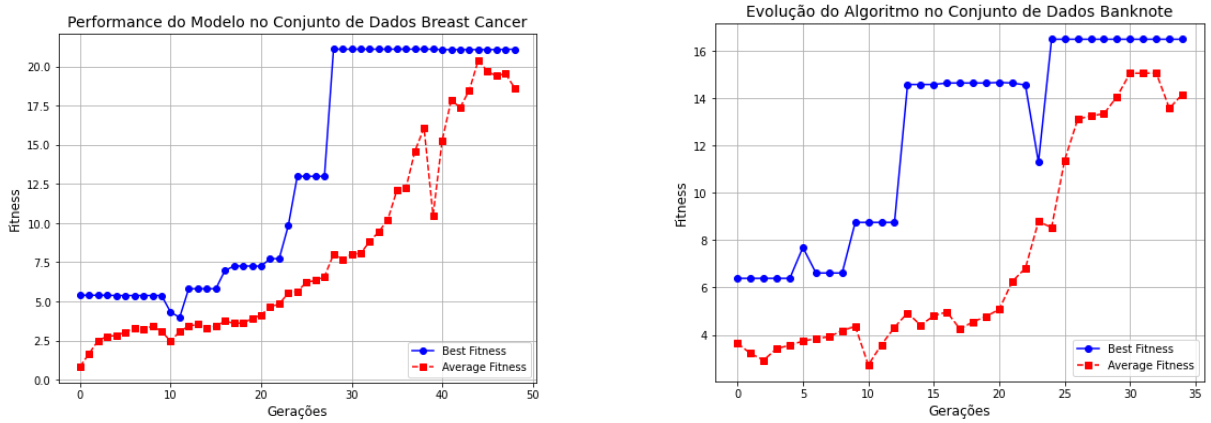


(a) Mapa de Decisão 1 (b) Mapa de Decisão 2

Figura 21: Mapas de Decisão e Circuitos Quânticos correspondentes, para o caso em que o algoritmo genético retorna duas interpretações distintas do mesmo conjunto de dados.

5.4 CONVERGÊNCIA E EVENTOS DE HIPERMUTAÇÃO

A convergência do algoritmo genético é outro aspecto relevante. Conforme demonstrado na Figura 22, o evento de hipermutação é empregado com o objetivo de remover a população de um possível mínimo local, caracterizado por uma redução abrupta no valor médio de *fitness* da população e de seu melhor indivíduo. Quando esse procedimento não é bem-sucedido e a população retorna ao mesmo mínimo, o algoritmo é interrompido, finalizando antes de alcançar o número máximo de gerações estipuladas. Esse comportamento auxilia na preservação de recursos computacionais, ao interromper a execução quando não há mais progresso significativo.



(a) Breast Cancer (b) Class1

Figura 22: Evolução do Algorítimo Genético

5.5 COMPARAÇÃO COM A LITERATURA

Ao comparar os resultados obtidos nos conjuntos de dados reais com os presentes na literatura, conforme mostrado na Tabela 11, nota-se que os modelos apresentam desempenho inferior nos conjuntos *Penguins* e *Heart Disease*. Esse resultado pode ser explicado por carac-

terísticas específicas desses conjuntos de dados. No entanto, em outros conjuntos, os modelos apresentam desempenho equivalente ao estado da arte.

Destaca-se, em particular, o desempenho dos modelos no conjunto de dados *Breast Cancer*. Mesmo utilizando apenas 1 qubit, os modelos alcançaram acurácias superiores a 90%, apesar da presença de 9 *features* de entrada. Esses resultados demonstram a alta capacidade de interpretação e processamento dos dados por parte desses modelos, mesmo em cenários de maior complexidade e com um número reduzido de parâmetros.

Tabela 11: Comparação de desempenho entre a proposta e a literatura em diferentes conjuntos de dados

Conjunto de Dados	**Proposta**	**Desempenho na Literatura**	**Método**	**Referência**
Iris	98.9% ± 0.36	100%	Máquina de Vetores de Suporte	Prathima & T (2023)
Penguins	64.6% ± 0.66	100%	Rede Neural	Hua & Goldsztein (2022)
Banknote	98.0% ± 0.41	100%	Rede Neural	S. Shahani & Priya (2018)
Blood	77.0% ± 0.0	77.0% ± 0.38	Rede Neural	Faris *et al.* (2016)
Heart Disease	42.3% ± 1.72	97.4%	K-Vizinhos Mais Próximos	Aljanabi *et al.* (2018)
Breast Cancer	96.1% ± 0.81	97.23% ± 0.35	Rede Neural	Faris *et al.* (2016)

6

CONCLUSÃO

A computação quântica tem atraído crescente interesse nos últimos anos, seja pela promessa de oferecer uma capacidade de processamento extraordinária, ou pela preocupação com a vulnerabilidade dos algoritmos de criptografia atuais. Contudo, na era NISQ (*Noisy Intermediate-Scale Quantum*), caracterizada por *qubits* ruidosos e tempos de decoerência reduzidos, surgem limitações que afetam a aplicabilidade dos sistemas quânticos. Nesse contexto, os circuitos quânticos variacionais, que integram recursos da computação clássica e quântica, têm se destacado como uma solução promissora, ainda que enfrentem os desafios inerentes à era NISQ.

Este trabalho teve como meta desenvolver e implementar um algoritmo genético capaz de criar circuitos quânticos variacionais otimizados para o carregamento e processamento de dados, com base em diferentes conjuntos de dados.

O algoritmo foi projetado para ajustar os circuitos quânticos conforme uma função de aptidão predefinida, resultando em arquiteturas mais eficientes e adequadas às limitações da computação quântica atual. Esses circuitos, caracterizados por uma estrutura simplificada e um número reduzido de portas, evitam redundâncias sem comprometer o desempenho.

Os resultados mostraram ser possível solucionar problemas complexos com circuitos quânticos simples, utilizando uma quantidade mínima de portas e parâmetros eficazmente. Em certos casos, o desempenho foi comparável ao estado da arte. No entanto, essa vantagem não foi uniformemente observada em todos os conjuntos de dados, sugerindo que questões como o desbalanceamento de classes e o elevado número de saídas impõem desafios às arquiteturas criadas.

Para trabalhos futuros, é essencial expandir o escopo de testes, aplicando o algoritmo a uma gama mais ampla de problemas reais e explorando as características específicas dos dados. Será importante identificar quais tipos de problemas podem ser resolvidos de forma eficiente com circuitos simples e quais demandam arquiteturas mais sofisticadas. Além disso, é necessário investigar outras funções de aptidão que considerem critérios adicionais para a otimização dos circuitos, bem como avaliar possíveis melhorias no algoritmo genético que aprimorem o processo de busca. Recomenda-se, ainda, uma análise formal da capacidade de mapeamento de dados no circuito, a fim de delimitar claramente as limitações da estratégia adotada.

Por fim, sugere-se aplicar o algoritmo a diferentes classes de problemas dentro da computação quântica, além dos circuitos variacionais. Sua utilização em outros contextos permitirá avaliar sua capacidade de identificar arquiteturas inovadoras para novos desafios, ampliando suas contribuições ao campo da computação quântica.

REFERÊNCIAS

Aljanabi, M., Qutqut, M., & Hijjawi, M. (2018). Machine learning classification techniques for heart disease prediction: A review. *International Journal of Engineering and Technology*, 7:5373–5379.

Altares-López, S., Ribeiro, A., & García-Ripoll, J. J. (2021). Automatic design of quantum feature maps. *Quantum Science and Technology*, 6(4):045015.

Bennett, C. H., Brassard, G., Crépeau, C., Jozsa, R., Peres, A., & Wootters, W. K. (1993). Teleporting an unknown quantum state via dual classical and einstein-podolsky-rosen channels. *Physical Review Letters*, 70(13):1895–1899.

Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., Alam, M. S., Ahmed, S., Arrazola, J. M., Blank, C., Delgado, A., Jahangiri, S., Kardas, K., Madsen, L. S., Niu, Z., Szava, A., Vargas-Hernández, J. F., Zaletel, Z., Bromley, T. R., & Killoran, N. (2018). Pennylane: Automatic differentiation of hybrid quantum-classical computations. *arXiv preprint arXiv:1811.04968*.

Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S. C., Endo, S., Fujii, K., McClean, J. R., Mitarai, K., Yuan, X., Cincio, L., & Coles, P. J. (2021). Variational quantum algorithms. *Nature Reviews Physics*, 3:625–644.

Chivilikhin, D., Samarin, A., Ulyantsev, V., Iorsh, I., Oganov, A. R., & Kyriienko, O. (2020). Mog-vqe: Multiobjective genetic variational quantum eigensolver.

Das, K. (2024). Quantum computing. *International Journal For Science Technology And Engineering*.

Ding, L. & Spector, L. (2022). Evolutionary quantum architecture search for parametrized quantum circuits. *Genetic and Evolutionary Computation Conference Companion (GECCO '22 Companion)*.

Dua, D. & Graff, C. (2019). UCI machine learning repository.

Farhi, E., Goldstone, J., & Gutmann, S. (2014). A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*.

Faris, H., Aljarah, I., & Mirjalili, S. (2016). Training feedforward neural networks using multi-verse optimizer for binary classification problems. *Applied Intelligence*, 45(2):322–332.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The Elements of Statistical Learning*. Springer, New York, 1st edition.

Goldberg, D. E. (1989a). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA.

Goldberg, D. E. (1989b). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Grover, L. K. (1996). A fast quantum mechanical algorithm for database search.

Hatakeyama-Sato, K., Igarashi, Y., Kashikawa, T., Kimura, K., & Oyaizu, K. (2023). Quantum circuit learning as a potential algorithm to predict experimental chemical properties. *Digital Discovery*, 2:165–176.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, Cambridge, MA.

Hua, A. & Goldsztein, G. (2022). Using machine learning to predict penguin species. *Journal of Student Research*, 11(4).

Huang, Y., Li, Q., Hou, X., Wu, R., Yung, M.-H., Bayat, A., & Wang, X. (2022). Robust resource-efficient quantum variational ansatz through an evolutionary algorithm. *Physical Review A*, 105(5).

Jiawei Han, M. k. & Pei, J. (2011). *Data Mining: Concepts and Techniques*. Elsevier, Burlington, MA, USA, 3rd edition.

Kyriienko, O., Paine, A. E., & Elfving, V. E. (2021). Solving nonlinear differential equations with differentiable quantum circuits. *Phys. Rev. A*, 103:052416.

Ladd, T. D., Jelezko, F., Laflamme, R., *et al.* (2010). Quantum computers. *Nature*, 464(7285):45–53.

Lu, Z., Shen, P.-X., & Deng, D.-L. (2021). Markovian quantum neuroevolution for machine learning. *Physical Review Applied*, 16(4).

Martyniuk, D., Jung, J., & Paschke, A. (2024). Quantum architecture search: A survey.

McClean, J. R., Boixo, S., Smelyanskiy, V. N., Babbush, R., & Neven, H. (2018). Barren plateaus in quantum neural network training landscapes. *Nature Communications*, 9(1):4812.

McClean, J. R., Romero, J., Babbush, R., & Aspuru-Guzik, A. (2016). The theory of variational hybrid quantum-classical algorithms. *New Journal of Physics*, 18(2):023023.

Mitarai, K., Negoro, M., Kitagawa, M., & Fujii, K. (2018). Quantum circuit learning. *Physical Review A*, 98(3).

Mitchell, M. (1998). *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA.

Nesterov, Y. (2018). *Lectures on Convex Optimization*. Springer.

Nielsen, M. A. & Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, UK, 10th edition.

Paine, A. E., Elfving, V. E., & Kyriienko, O. (2023). Quantum quantile mechanics: Solving stochastic differential equations for generating time-series. *Advanced Quantum Technologies*, 6(10).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.-H., Zhou, X.-Q., & Love, P. J. (2014). A variational eigenvalue solver on a photonic quantum processor. *Nature communications*, 5:4213.

Prathima, P. & T, R. K. (2023). Comparison on iris dataset using classification techniques. *Journal of Emerging Technologies and Innovative Research (JETIR)*.

Preskill, J. (2018). Quantum computing in the nisq era and beyond. *Quantum*, 2:79.

Python Software Foundation (2023). *Python Language Reference, version 3.8*. Available at `http://www.python.org`.

Rattew, A. G., Hu, S., Pistoia, M., Chen, R., & Wood, S. (2020). A domain-agnostic, noise-resistant, hardware-efficient evolutionary variational quantum eigensolver.

S. Shahani, A. J. & Priya, R. L. (2018). Analysis of banknote authentication system using machine learning techniques. *International Journal of Computer Applications*, 179:22–26.

Schillo, N. & Sturm, A. (2024). Quantum circuit learning on nisq hardware.

Schuld, M., Bergholm, V., Gogolin, C., Izaac, J., & Killoran, N. (2019). Evaluating analytic gradients on quantum hardware. *Physical Review A*, 99(3):032331.

Schuld, M. & Petruccione, F. (2019). *Quantum Machine Learning: An Applied Approach*. Springer, Cham, Switzerland.

Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509.

Zhang, A. & Zhao, S. (2022). Evolutionary-based quantum architecture search.