Estudos sobre o algoritmo de Grover e sua implementação*

Liliana Souza do Carmo[1], **Gisele Bosso de Freitas**[1]

[1] Centro de Ciências Exatas, Naturais e Tecnológicas (CCENT)
Universidade Estadual da Região Tocantina do Maranhão (UEMASUL)
Imperatriz, MA – Brasil

liliana.uemasul@gmail.com, giseleuemasul@gmail.com

Abstract. *The Quantum Computation is based on the principles of Quantum Mechanics. Unlike classical computers, quantum ones have parallelism capabilities by default, which makes their processing speed for solving problems much higher. For this performance, the use of certain algorithms is necessary. In this text, we present some studies on the Grover algorithm, with the aim of knowing and learning how to implement it in quantum simulators, in addition to analyzing and interpreting the results obtained.*

Resumo. *A Computação Quântica se baseia nos princípios da Mecânica Quântica. Ao contrário dos computadores clássicos, os quânticos possuem capacidade de paralelismo como padrão, o que torna sua velocidade de processamento para resolução de problemas muito maior. Para esse desempenho, o uso de determinados algoritmos faz-se necessário. Neste texto, apresenta-se alguns estudos sobre o algoritmo de Grover, com o objetivo de conhecer e aprender a implementá-lo nos simuladores quânticos, além de analisar e interpretar os resultados obtidos.*

1. Introdução

Os computadores quânticos são projetados para terem mais eficiência do que os computadores clássicos, podendo ser bastante utilizados em áreas como criptografia, otimização, simulação de sistemas quânticos e resolução de grandes sistemas de equações lineares. Um exemplo disso é o algoritmo de Grover [Grover 1996], um dos primeiros algoritmos quânticos a serem apresentados, é utilizado para realizar busca em banco de dados não-ordenado e forma a base da computação quântica.

A generalização do algoritmo de Grover, também chamada de estimativa de amplitude [Brassard et al. 2002], é um algoritmo quântico importante porque serve de base para outros que foram propostos posteriormente [Jordan 2021]. Estimar a amplitude está na base da maioria dos algoritmos quânticos conhecidos relacionados aos problemas de colisões e propriedades de gráficos.

Com este trabalho, deseja-se estudar o algoritmo de Grover, sua generalização e aprender a implementá-lo no Qiskit, além de analisar e interpretar os resultados obtidos com esta implementação, a escolha da utilização do simulador Qiskit foi por ser um dos mais conhecidos e utilizados além de estar disponível de forma gratuita atualmente.

Para a implementação do algoritmo de Grover, utiliza-se o Qiskit (Quantum Information Science Kit) [Qiskit 2021], um Software Development Kit (SDK) open-source

*Trabalho Concluído.

desenvolvido pela IBM para construção, otimização e execução de circuitos e algoritmos em computação quântica. Pode-se compará-lo a um pacote da linguagem Python, porque permite uma integração com o código Python que facilita muito a construção de algoritmos, dada a simplicidade da sintaxe e a comum necessidade de se executar parte dos algoritmos em conjunto com recursos de computação clássica, para resolver problemas de otimização, química quântica, física, aprendizado de máquina e finanças [Javadi-Abhari and Gambetta 2021].

2. Fundamentos e metodologia

Para a realização deste trabalho, iniciamos com estudos dos conceitos básicos de computação quântica, seguida do algoritmo de Grover e de estudos sobre simulação em circuitos quânticos, mais especificamente no Qiskit (IBM, 2021), em seguida implementar algoritmos e analisar seus funcionamento e resultados.

Um qubit representa um estado quântico ($|s\rangle = a\,|0\rangle + b\,|1\rangle$, com $a, b \in C$ e $a^2 + b^2 = 1$) e está numa superposição de zeros e uns num determinado instante.

Em um computador quântico, para que o processamento da informação aconteça, utiliza-se as portas lógicas quânticas, que são fundamentais para a construção de circuitos e que atendem a condições de normalização e implementam operações inversíveis [de Castro 2007]. Alguns exemplos de portas lógicas são:

- NOT quântica: é representada por *X* (troca o bit quântico), ou seja,

$$X\,|0\rangle = |1\rangle \;\; e\; X\,|1\rangle = |0\rangle\; X(\alpha\,|0\rangle + \beta\,|1\rangle) = \alpha X\,|0\rangle + \beta X\,|1\rangle = \alpha\,|1\rangle + \beta\,|0\rangle \tag{1}$$

- *Z*: não altera o estado $|0\rangle$, mas muda o sinal do estado $|1\rangle$ para $|-1\rangle$:

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

- *H* (porta de Hadamard): Conhecida como porta H ou raiz quadrada da porta NOT.

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

É uma porta lógica que tem como saída, os dois estados fundamentais, isto é

$$|0\rangle \Rightarrow \frac{(|0\rangle + |1\rangle)}{\sqrt{2}}; |1\rangle \Rightarrow \frac{(|0\rangle - |1\rangle)}{\sqrt{2}} \tag{2}$$

Pode-se considerar como circuitos quânticos, um mecanismo que é constituído de vários tipos de portas quânticas. À vista disso, é demonstrado a seguir a superposição de dois estados fundamentais de um computador quântico, representado pelo estado genérico $|s\rangle$. Para isso, é aplicado o produto tensorial entre duas portas lógicas quânticas, porta Hadarmard e dois estados quânticos, $|0\rangle$ e $|1\rangle$,

$$|s\rangle = (H \otimes H)(|0\rangle\,|0\rangle) \tag{3}$$

reescrevendo-os:

$$|s\rangle = H^{\otimes 2}\,|00\rangle \tag{4}$$

aplicando o produto tensorial:

$$|s\rangle = H\,|0\rangle \otimes H\,|0\rangle \tag{5}$$

substituindo o valor que correspondem a $H\,|0\rangle$, temos que:

$$|s\rangle = \frac{(|0\rangle + |1\rangle)}{\sqrt{2}} \otimes \frac{(|0\rangle + |1\rangle)}{\sqrt{2}} \tag{6}$$

resolvendo o produto tensorial, temos:

$$|s\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle) \tag{7}$$

Representando, em uma base de quatro ou mais, têm-se:

$$|s\rangle = \frac{1}{2}(|0\rangle + |1\rangle + |2\rangle + |3\rangle) \tag{8}$$

Diante disso, equação [7], é constatado a superposição de estados, através de dois estados quânticos e a atuação de duas portas lógicas.

3. Algoritmo de Grover

A habilidade do computador quântico em comparação ao computador clássico é de fato notável, principalmente quando se trata da velocidade em realizar pesquisas em um determinado banco de dados. Um exemplo disso, pode ser retratado pelo algoritmo de Grover [Qiskit 2021], que usa o fenômeno de paralelismo quântico para encontrar soluções para um problema de busca não-ordenada. Nesse contexto, considere uma lista com N elementos, dos quais é necessário localizar somente um, tendo em vista, que esse elemento possui características exclusivas, que será denominado de X, conforme se apresenta em [Qiskit 2021].

Agora, imagine que cada elemento seja uma moeda de mesmo valor, ou seja, um contador $N-1$ moedas de valor 10, salvo o item X, nesse caso, a única moeda de valor 5, lembrando que esses valores foram escolhidos aleatoriamente. Essa situação se encontra na figura [1].

Figura 1. Lista com N moedas. Fonte: [Qiskit 2021]

Para resolver esse problema utilizando um computador clássico será preciso analisar, aproximadamente, $\frac{N}{2}$ ou quem sabe, olhar uma moeda por vez; já no computador quântico, o mesmo problema poderá ser solucionado a partir da amplificação de amplitude de Grover, o qual haverá uma economia de tempo considerável em relação ao computador clássico, [Qiskit 2021]. Esse modo de amplificar a amplitude é dado pelo crescimento da probabilidade de encontrar a posição do elemento X. Na figura [2], é exposto a amplitude

do objeto a ser encontrado, que será maior do que os demais, e então, resultará no item procurado.

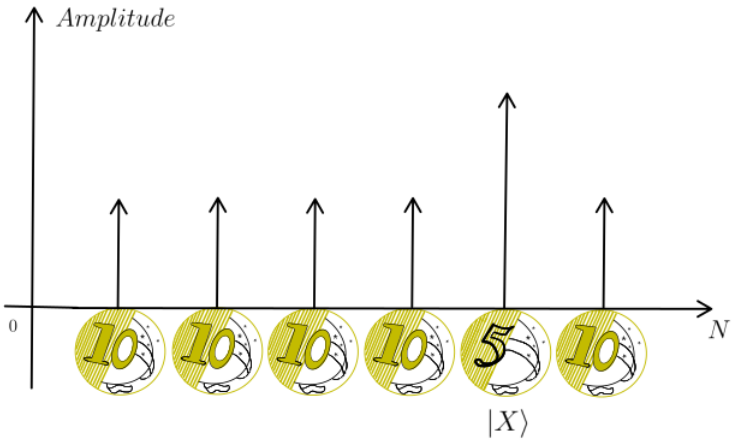

Figura 2. Representação: Amplificação de Amplitude. Fonte: [Qiskit 2021]

Em conformidade ao exemplo da moeda, é retratado na seção [4], a atuação do algoritmo de Grover em busca do estado $|X\rangle = |11\rangle$, o mesmo exibido na equação [7], em que foi usado 2 qubits, [Qiskit 2021]. É importante ser ressaltado, que a quantidade de "elementos", N consistirá em $N = 2^n$, sendo que n é o número de qubits. Portanto, N será igual a 4 estado. Precisamente como descrito na demonstração algébrica de superposição de estados em [8].

4. Implementação do algoritmo de Grover

Para implementar esse algoritmo num simulador quântico, utiliza-se a linguagem Python juntamente com a biblioteca Project Jupyter [Jupyter 2021] [1]. Por conseguinte, importa-se a biblioteca Qiskit[2] instanciamos essa classe em uma variável, tal como $circuit = QuantumCircuit\,()$. A biblioteca Quiskit exige um valor inteiro para determinar a quantidade de qbits necessários no circuito. O valor empregado na entrada dessa variável compreende a quantidade de qubits que serão usados nesse algoritmo, nesse caso, dois qubits os quais foram adicionados no objeto da variável. Em seguida, é escrito o algoritmo para desenhar essa representação, figura [3].

```
from qiskit import *
%matplotlib inline
circuit = QuantumCircuit(2)
circuit.draw(output = 'mpl')
```

Então, aplica-se a porta Hadamard no qubit 0 e no qubit 1. Para apresentar em forma de figura o circuito contruído, até o momento, utiliza-se o comando $circuit.draw(output = 'mpl')$ da classe Quiskit, como na figura [3].

```
circuit.h([0,1])
circuit.draw(output = 'mpl')
```

[1]Essa implementação do algoritmo de Grover no Qiskit, está baseada no guia, disponível em [Qiskit 2021].

[2]Para download do Qiskit, basta acessar o site `https://qiskit.org/` e seguir todos os passos apresentados pelo mesmo.

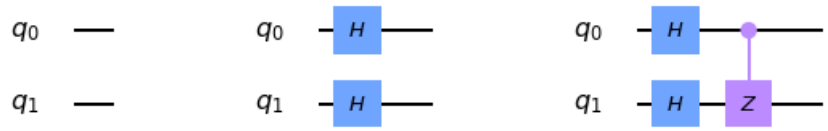

Figura 3. Podemos ver, da esquerda para a direita, a representação de dois Qubits, da aplicação das portas Hadamard e Controlada z.

Insere-se a porta controlada z, representada por cz, que será o oráculo, que tens como propósito "amplificar a amplitude", como visto no exemplo da moeda, figura [3].

```
circuit.cz(0,1)
circuit.draw(output = 'mpl')
```

Em seguida, é aplicado a porta Hadamard e a porta Z, e ainda foi adicionado uma barreira, a fim de exibir um diagrama ordenado, figura [4] e por fim, realizou-se uma medida em todos os qubits através de $circuit.measure\_all()$, figura [5].

```
circuit.h([0,1])
circuit.z([0,1])
circuit.barrier()
circuit.cz(0,1)
circuit.barrier()
circuit.h([0,1])
circuit.draw(output = 'mpl')
```

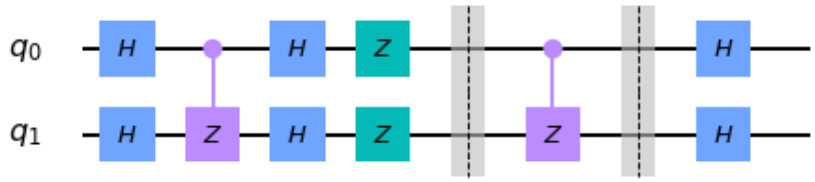

Figura 4. Aplicação de Portas Quânticas no Circuito.

```
circuit.measure_all()
circuit.draw(output = 'mpl')
```

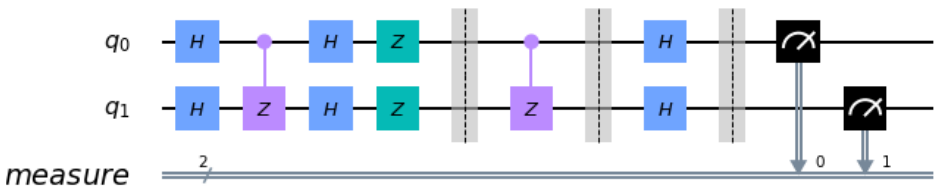

Figura 5. Medição no Circuito Quântico.

Através do simulador $qasm\ simulator$, que faz parte da lista de simuladores de grande performance da IBM, frequentemente aplicado em modelagens comuns de circuitos quânticos, [IBM 2021], obtém-se todos os estados possíveis com suas respectivas

probabilidades. E portanto, encontrou-se o estado $|11\rangle$ com probabilidade de 100%, figura [6]. [3]

```
simulator = Aer.get_backend('qasm_simulator')
result = execute(circuit, backend = simulator, shots = 1).result()
counts = result.get_counts()
from qiskit.tools.visualization import plot_histogram
plot_histogram(counts)
```

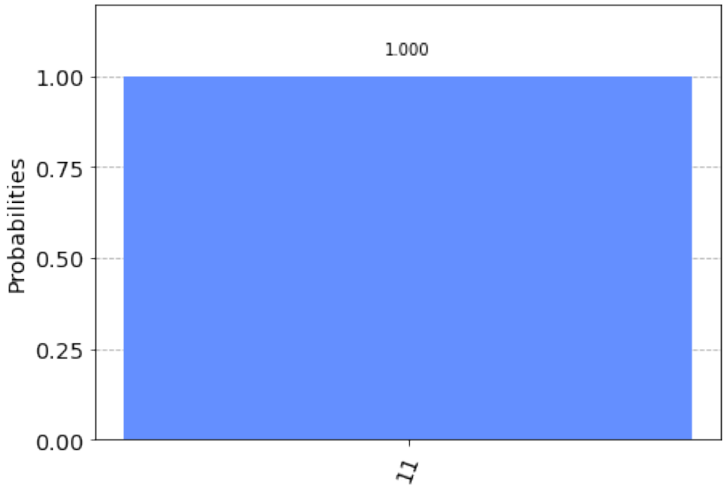

Figura 6. Representação do Estado Procurado $|11\rangle$ com probabilidade igual a 1.

5. Considerações finais

A importância de estudar o algoritmo de Grover reside no fato de que, além de ser um dos primeiros algoritmos da literatura e, portanto, servir de base para uma série de outros que foram propostos posteriormente, o algoritmo de Grover é considerado ótimo, ou seja, ele apresenta a menor complexidade, dentre todos os possíveis algoritmos que poderiam resolver o mesmo tipo de problema. Isso significa que qualquer algoritmo que acessa o banco de dados usando um dado operador matemático, este deve ser aplicado pelo menos tantas vezes quanto o algoritmo de Grover [Bennett et al. 1997], que executa uma busca em um conjunto de dados em tempo $\sqrt{N}$.

Dado o caráter probabilístico do algoritmo de Grover, a sua utilização quando se trata de um conjunto de dados não estruturado e muito grande traz agilidade à busca de forma que o resultado, certamente, tem alta probabilidade de ser o correto.

Referências

Bennett, C. H., Bernstein, E., Brassard, G., and Vazirani, U. (1997). Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523.

Brassard, G., Hoyer, P., Mosca, M., and Tapp, A. (2002). Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74.

[3]Para uma melhor compreensão, segue o código fonte dessa implementação: `https://github.com/lianafis/Algoritmo_Grover/blob/f2bac3baaa13b085adcec967d7b2e1874e18f5b3/Grover.ipynb`

de Castro, L. N. (2007). Fundamentals of natural computing: an overview. *Physics of Life Reviews*, 4(1):1–36.

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

IBM (2021 (Acessado: Setembro 30, 2021)). Ibm quantum simulators. Disponível em: https://quantum-computing.ibm.com/services/docs/services/manage/simulator/#qasm.

Javadi-Abhari, A.; Nation, P. and Gambetta, J. (2019 (Acessado: Setembro 23, 2021)). Qiskit – write once, target multiple architectures. Disponível em: https://www.ibm.com/blogs/research/2019/11/qiskit-for-multiple-architectures/.

Jordan, S. (2021 (Acessado: Setembro 23, 2021)). Quantum algorithm zoo, microsoft quantum. Disponível em: https://quantumalgorithmzoo.org/.

Jupyter, P. (2021 (Acessado: Abril 15, 2021)). Disponível em: https://jupyter.org/.

Qiskit (2020 (Acessado: Março 23, 2021)). Grover's algorithm. Disponível em: https://qiskit.org/textbook/ch-algorithms/grover.html.