QISKIT E A PLATAFORMA IBM Q: UMA ABORDAGEM DE CIRCUITO PARA O ENSINO DA COMPUTAÇÃO QUÂNTICA

Adriana de Nazaré Souza Cordeiro – drik.cordeiro18@gmail.com
Universidade Federal do Pará, Faculdade de Engenharia de Computação
Rua Augusto Corrêa, 01 – Guamá
66075-110 – Belém – PA

Maria Lúcia M. Costa – luciacosta@ufpa.br
Universidade Federal do Pará, Faculdade de Física
Rua Augusto Corrêa, 01 – Guamá
66075-110 – Belém – PA

Wilson R. M. Rabelo – rabelo@ufpa.br
Universidade Federal do Pará, Instituto de Tecnologia
Rua Augusto Corrêa, 01 – Guamá
66075-110 – Belém – PA

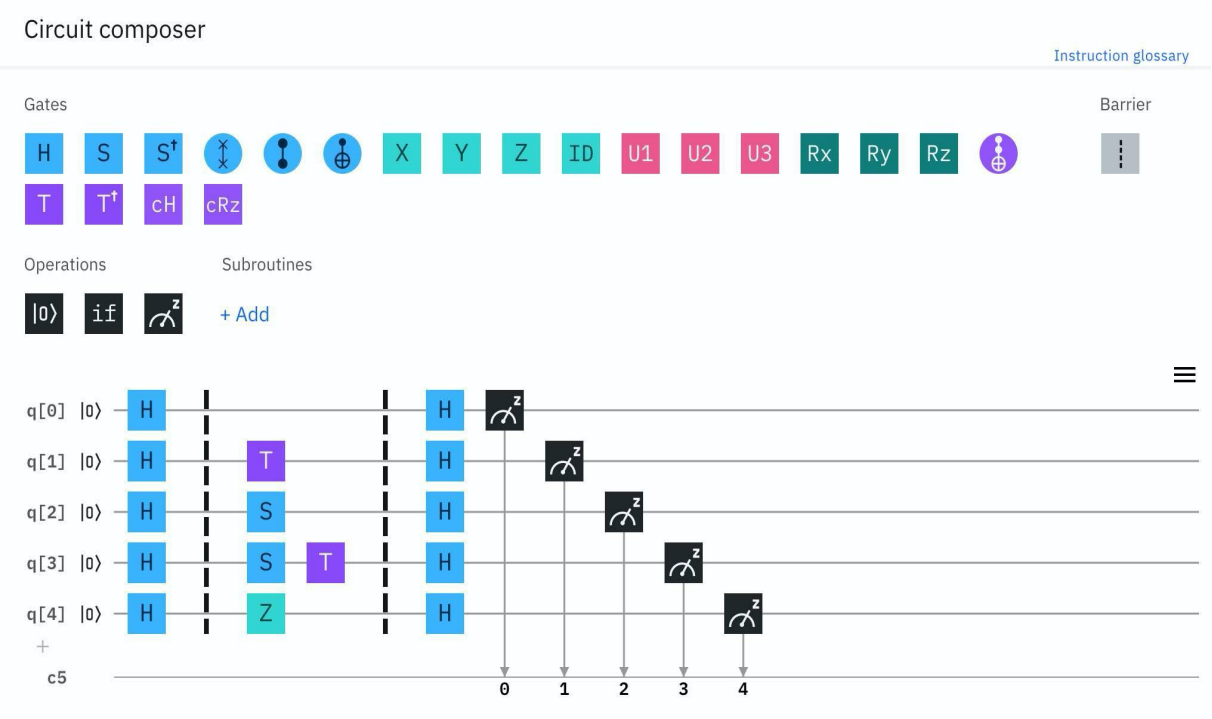
Resumo: *Este trabalho aborda como utilizar a plataforma de computação quântica da empresa IBM e seu framework, conhecido na literatura como IBM Q Experience e Qiskit, respectivamente. Esta plataforma fornece um conjunto de processadores quânticos acessados via nuvem. Esses processadores quânticos podem ser acessados gratuitamente por pesquisadores de diversas áreas: físicos, matemáticos, engenheiros, educadores. Qiskit é um framework de desenvolvimento fornecido pela mesma empresa, que fornece, além da simulação local, a conexão entre o usuário e o processador quântico onde se deseja executar circuitos quânticos ou algoritmos, voltados ao ensino de computação quântica. Nosso trabalho teve como objetivo fornecer um circuito simples que emaranha 2-qubits, utilizando o Qiskit como recurso pedagógico para estudar conceitos básicos da computação quântica como portas, circuitos e, além disso, ainda temos o processo de medição de qubits disponível neste framework. Após a revisão teórica, implementamos as portas e circuitos quânticos no Qiskit, fornecendo o código, o resultado da simulação e, a visualização gráfica desses resultados. Concluímos que o Qiskit provou ser um importante recurso pedagógico para explorar vários conceitos da mecânica quântica e da computação, como portas, circuitos, emaranhamento e medidas quânticas. Na visualização gráfica, o Qiskit pode auxiliar na discussão dos resultados obtidos em circuitos simples e complexos, com foco na pesquisa e ensino da computação quântica.*

Palavras-chave: *Qiskit. IBM Q. Simulador Quântico. Computação Quântica.*

1. INTRODUÇÃO

A área da computação quântica vem chamando atenção não somente por propor soluções teóricas para problemas considerados inviáveis com o uso da computação clássica, mas também pelos avanços experimentais, que estão permitindo tornar a computação quântica uma realidade em vários laboratórios do mundo (NIELSEN e CHUANG, 2010). Como exemplos de arquiteturas quânticas com poucos qubits, vamos citar dois exemplos: D-Wave One, disponibilizado em 2011 pela empresa de tecnologia D-Wave (D-Wave, 2020), e da plataforma IBM Q, lançado em 2016 pela empresa IBM (IBM Q, 2020). Abordando mais especificamente a arquitetura da IBM, conhecida na literatura como *IBM Q Experience*, a mesma forneceu um conjunto de vários processadores quânticos que podem ser acessados via nuvem para fins científicos, educacionais e comerciais. Para trabalhos científicos e educacionais a empresa disponibiliza processadores de 1-qubit, 5-qubits e 14-qubits, além dos servidores clássicos (computadores convencionais de alto desempenho) para rodar projetos dos usuários.

Figura 1 – A plataforma de acesso *IBM Q Experience*.



Fonte: Imagem de captura da plataforma IBM Q (IBM Q, 2020).

Na Figura 1, temos a plataforma de acesso do IBM Q e a representação computacional de um circuito na computação quântica. Após a montagem de um dado circuito de interesse, usando as portas quânticas disponíveis na parte superior da Figura 1, ou seja, as portas H, S, X, Y, S, etc. A plataforma também permite duas possibilidades: simular os circuitos quânticos em computadores convencionais ou realizar o processamento em um hardware quântico universal de 5-qubits supercondutores (IBM Q, 2020). Portanto, um experimento no IBM Q consiste: (a) especificar o circuito, usando a interface gráfica ou um editor de texto disponível pela plataforma; (b) executar o circuito de interesse no simulador ou dispositivo real; (c) Realizar as medidas sobre os qubits.

Uma outra forma de acesso é através do framework Qiskit (QISKIT, 2020). Esse framework é composto por quatro pacotes: Terra, Aer, Ignis e Aqua. Cada um desses componentes possuem uma dada finalidade. Entretanto, o pacote Qiskit Terra fornece a base necessária para programar quanticamente os seus processadores em dois estágios: construir e executar.

1.1. Download, instalação e configuração do framework Qiskit

A instalação dos recursos necessários para o Qiskit é dada pela distribuição Anaconda (ANACONDA, 2020). Essa distribuição é um conjunto de plataformas e bibliotecas, gratuita e de código aberto, da linguagem de programação Python e de vários outros pacotes de desenvolvimento destinados a computação científica e ciência de dados, como o Jupyter Notebook (JUPYTER, 2020). Anaconda também está disponível para os sistemas operacionais Linux (Ubuntu), macOS e Windows.

Após a instalação da plataforma Anaconda e de seus elementos, a IBM recomenda usar o Qiskit em um ambiente virtual Python separado de outras aplicações, e assim, manter as dependências necessárias na versão desejada. Para criar o ambiente com o Python, inicialize o terminal (prompt de comandos) no diretório em que se deseja trabalhar e use os comandos de criação e ativação do ambiente (macOS , Linux e Windows):

$ conda create -n nome_amb python=3

$ conda activate nome_amb

O nome do ambiente criado nos comandos acima é *nome_amb.* Dentro do ambiente criado e ativado, para instalar o pacote de desenvolvimento Qiskit usamos o comando:

$ pip install qiskit

Para fazer upgrade do Qiskit para uma versão mais recente é necessário desinstalar a versão atual, antes de instalar a nova versão. Portanto, use o comando:

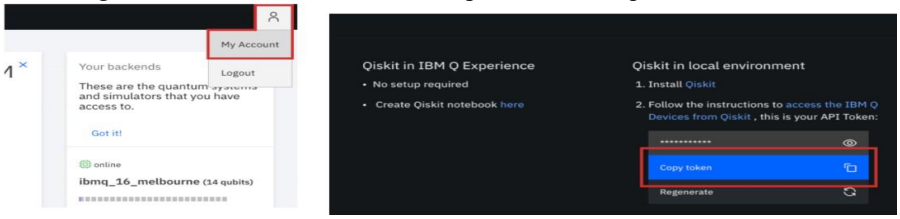
$ pip uninstall qiskit

Para otimizar sua experiência na utilização do Qiskit (QISKIT, 2020), a IBM recomenda também a instalação, através do comando Pip, dos seguintes pacotes no ambiente criado: *matplotlib, jupyter, ipywidgets, seaborn* e o *pygments*. Se a instalação ocorreu corretamente, os pacotes estarão visíveis através do comando *conda list* no ambiente criado pelo usuário no terminal.

1.2. Conexão com processadores quânticos na nuvem via Qiskit

Para utilizar gratuitamente a rede de processadores quânticos da IBM através do Qiskit é necessário configurar uma conta de acesso. O usuário deve criar uma conta, e assim, obter seu API Token individual. O API Token estará disponível em "My Account" no local "*Qiskit in local environment*", veja a Figura 2. É possível visualizar e copiar o API Token na opção "*Copy token*".

Figura 2 – API Token de acesso aos processadores quânticos da IBM.



Fonte: Imagem de captura do site IBM Q (IBM Q, 2020).

No terminal de prompt (macOS, Linux ou Windows) e, com o ambiente (***nome_amb***) criado e inicializado, inicialize o aplicativo Jupyter Notebook. Utilize os seguintes comandos para

armazenar o Token, e substituir "MY_API_TOKEN" pelo código adquirido na plataforma IBM Q:

> ***from qiskit import IBMQ***
> ***IBMQ.save_account('MY_API_TOKEN')***

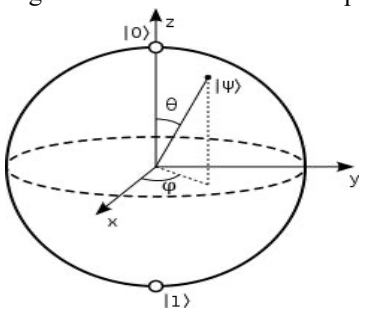
Isso irá criar um arquivo local de configuração que permitirá o acesso aos processadores quânticos da IBM. Isto é, o usuário terá acesso para rodar trabalhos remotamente pela nuvem nessa rede de dispositivos, usando o Qiskit como base de acesso.

Todos os procedimentos acima são requisitos básicos para o desenvolvimento de projetos com circuitos e algoritmos quânticos usando a plataforma IBM Q. Seja através do acesso a interface gráfica do usuário ou GUI (Graphical User Interface, veja a Figura 1) ou pelo framework Qiskit, o objetivo central é acessar os computadores quânticos disponíveis pela IBM e auxiliar na construção de circuitos quânticos. Portanto, nosso trabalho consistiu em estudar e analisar um circuito quântico simples de 2-qubits, conhecido como circuito quântico emaranhador. Esse projeto de circuito foi usado como um modelo pedagógico, para caracterizar os recursos do Qiskit em vários aspectos como: (1) um estudo teórico dos conceitos da computação quântica, (2) uma simulação ideal do projeto e (3) análise final dos resultados obtidos.

2. QUBIT: UNIDADE BÁSICA DA COMPUTAÇÃO QUÂNTICA

Uma unidade central na computação quântica é o qubit, ou seja, um bit de informação codificado nos estados quânticos de um sistema de 2-níveis (NIELSEN e CHUANG, 2010). Matematicamente, os estados quânticos são descritos por um vetor bidimensional no espaço Hilbert complexo de dimensão finita. Para um dado sistema quântico escolhemos uma base vetorial padrão que são dois estados distinguíveis: $|0\rangle$ e $|1\rangle$ também conhecida como base padrão da computação quântica. Assim, um qubit pode ser escrito como uma superposição de estados da base computacional e formam uma base ortonormal nesse espaço vetorial. Isto é, matematicamente podemos escrever: $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, onde α e β são as amplitudes de probabilidades dos respectivos estados da base, com $|\alpha|^2 + |\beta|^2 = 1$. Uma visão geométrica do qubit é mapear todos os estados quânticos possíveis sobre a superfície de uma esfera de raio unitário. Isto é, mapeamos todas as amplitudes de probabilidades α e β em função dos ângulos θ e φ, como: $|\psi\rangle = \cos(\theta/2)\,|0\rangle + e^{i\varphi}\sin(\theta/2)\,|1\rangle$, veja a Figura 3.

Figura 3 – Representação geométrica dos estados de 1-qubit na Esfera de Bloch.



Fonte: Figura retirada de RABELO e COSTA, 2018.

3. PORTAS E CIRCUITOS QUÂNTICOS

3.1. Portas quânticas de 1 e 2-qubits

Os processadores quânticos da plataforma *IBM Q* podem executar portas quânticas de 1, 2 e múltiplos qubits. Como exemplo podemos citar: a porta H, S, X, Y, Z que são portas de 1-qubit e a porta CNOT (Não-Controlado), como exemplo de porta padrão de 2-qubits. Na parte superior da Figura 1, temos um conjunto portas quânticas que podem ser implementadas nos processadores quânticos da IBM usando a GUI da plataforma. Em nosso trabalho vamos explorar duas portas bem conhecidas da literatura da computação quântica, a Hadamard e a CNOT. Para melhor compreensão do processamento das portas nos qubits, vamos detalhar a atuação de cada uma delas.

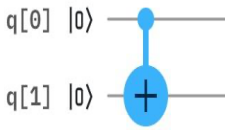
A primeira porta quântica é a Hadamard. Essa porta atua em um único qubit. A transformação matricial e mudança na base computacional $|0\rangle$ e $|1\rangle$ é dada, respectivamente, por:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} ; |0\rangle \mapsto H \mapsto \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \; ; \; |1\rangle \mapsto H \mapsto \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \, .$$

Observe que essa transformação converte o qubit que estar no estado $|0\rangle$ ou $|1\rangle$ em uma superposição (combinação) de estados quânticos da base computacional, $(|0\rangle \pm |1\rangle)/\sqrt{2}$. Observe que não podemos realizar essa tarefa com bits clássicos, visto que, os mesmos podem somente assumir um único valor possível, 0 ou 1.

A outra porta de interesse é a CNOT, veja a Figura 4. Essa porta atua em dois qubits, sendo o primeiro qubit, o de controle e o segundo qubit, o alvo. Essa porta atua nos qubits com a seguinte transformação matricial na base computacional $|0\rangle_c$, $|1\rangle_a$, onde os índices representam os qubits de controle e o alvo:

Figura 4 – Representação de circuito, matricial e de estados da base computacional da porta CNOT.



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

$$|0\rangle|0\rangle \mapsto CNOT \mapsto |0\rangle|0\rangle$$
$$|0\rangle|1\rangle \mapsto CNOT \mapsto |0\rangle|1\rangle$$
$$|1\rangle|0\rangle \mapsto CNOT \mapsto |1\rangle|1\rangle$$
$$|1\rangle|1\rangle \mapsto CNOT \mapsto |1\rangle|0\rangle \; .$$

Fonte: figura gerada pelos próprios autores.

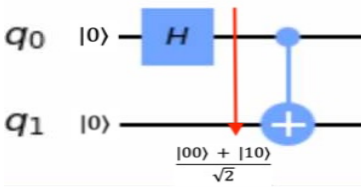
Observe que, na Figura 4 temos a representação da porta CNOT no circuito quântico, a forma matricial da CNOT e a atuação da mesma sobre os estados da base computacional, respectivamente. No circuito, temos que a primeira linha representa o qubit de controle e a segunda linha o qubit alvo. Quando o qubit de controle é $|0\rangle_c$, a porta CNOT não é habilitada. Portanto, não temos alteração do qubit alvo como podemos notar nas duas primeiras linhas acima. Entretanto, quando o qubit de controle é $|1\rangle_c$, a porta CNOT é habilitada. Agora a atuação será de inverter (ou negar) o qubit alvo. Podemos ver essa execução da CNOT nas duas últimas linhas da Figura 4.

3.2. Um circuito quântico emaranhador de 2-qubits

Vários protocolos de informação e computação quântica utilizam estados quânticos emaranhados, ou na linguagem da computação quântica, múltiplos qubits emaranhados. É consenso na literatura que vários protocolos possuem caraterísticas singulares devido ao uso do

emaranhamento quântico, entre eles podemos citar: a codificação superdensa (NIELSEN e CHUANG, 2010), o teleporte quântico (NIELSEN e CHUANG, 2010), o algoritmo quântico de fatoração (NIELSEN e CHUANG, 2010), entre outros. Estados quânticos puros, emaranhados (ou correlacionados quanticamente) e bipartidos, são aqueles que o sistema global AB não podem ser fatorados nos seus subsistemas A e B, ou seja, não podem ser escritos matematicamente da forma: $|\psi\rangle_{AB} = |\phi\rangle_A \otimes |\chi\rangle_B$. Fazendo uma ligação com a nossa notação, $|\phi\rangle_A$ é o nosso qubit q_0 e $|\chi\rangle_B$ é o nosso qubit q_1. Um exemplo de circuito que fornece um estado quântico emaranhado de 2-qubits é representado na Figura 5 com seu respectivo estado de saída. Observe os passos intermediários após atuação de cada porta quântica. A primeira porta H coloca o qubit q_0 em uma superposição de estados, como podemos ver a seta vermelha na Figura 5. A última porta, a CNOT permite a correlação quântica, ou emaranhamento (NIELSEN e CHUANG, 2010), entre os dois qubits q_0 e q_1, veja a Figura 5. Observe que agora não podemos fatorar esse sistema resultante (ou 2-qubits) em subsistemas.

Figura 5 – Circuito que gera um estado quântico emaranhado de 2-qubits e seu estado resultante.



$$|q_0 q_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle_0|0\rangle_1 + |1\rangle_0|1\rangle_1)$$

Fonte: Figura gerada pelos próprios autores.

4. PORTAS E CIRCUITO QUÂNTICO VIA FRAMEWORK QISKIT

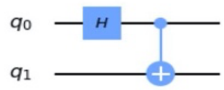
Neste tópico vamos demonstrar como usar o Qiskit como um recurso pedagógico na construção de blocos de portas e circuitos quânticos. Como já observamos na Figura 1, a plataforma *IBM Q Experience* disponibiliza uma interface gráfica do usuário, do tipo arraste e solte, para a criação do seu circuito ou algoritmo de interesse, a partir de um conjunto de portas de 1-qubit e/ou múltiplos qubits. Entretanto, para o nosso trabalho vamos apenas analisar as duas portas mencionadas na subseção anterior, as portas H e CNOT e, como podemos implementá-las via Qiskit. Posteriormente, vamos complementar usando essas portas para construir o circuito emaranhador de 2-qubits usando o Qiskit.

Antes de começarmos a usar o Qiskit, devemos inicializar a distribuição Anaconda, já discutida na introdução deste artigo. Posteriormente, dentro do ambiente Anaconda, iniciamos a plataforma Jupyter Notebook. Segundo o Projeto Jupyter (JUPYTER, 2020), essa plataforma é: "*um aplicativo web, de código aberto, que permite criar e compartilhar documentos que contenham códigos, equações, visualizações e texto*". Ou seja, esse aplicativo trabalha elementos web e a linguagem de programação Python conjuntamente. Entretanto, existem outras linguagens de programação que o Jupyter Notebook trabalha.

Na Figura 6, temos uma coleção de instruções do framework Qiskit declarados no Jupyter Notebook (JUPYTER, 2020). Essa figura mostra como podemos declarar as portas H e CNOT no Qiskit e visualizar essas portas.

Figura 6 – Declaração das portas H e CNOT usando o framework Qiskit.

```
In [1]: # Importar os pacotes necessários:
        import numpy as np
        from qiskit import(
          QuantumCircuit,
          execute,
          Aer)

In [2]: # Definir um circuito quântico de 2-qubits (q_0 e q_1):
        circuit = QuantumCircuit(2)

In [3]: #Aplicar a porta H no qubit q_0:
        circuit.h(0)

<qiskit.circuit.instructionset.InstructionSet at 0x7f9c778829d0>

In [4]: #Aplicar a porta Cx(CNOT) nos qubits q_0, q_1:
        circuit.cx(0, 1)

<qiskit.circuit.instructionset.InstructionSet at 0x7f9c77882e50>

In [5]: # Visualizando as portas declaradas acima:
        circuit.draw('mpl')
```



Fonte: figura gerada pelos próprios autores.

Observe na Figura 6, que começamos importando todas as bibliotecas necessárias para o projeto. Além do próprio Qiskit, outras bibliotecas do Python como Numpy. Posteriormente, criamos o circuito e as portas H e CNOT e, finalizamos com a visualização dessas portas no diagrama padrão de circuito da computação quântica. Note que já começamos implementar em parte o circuito de interesse. Entretanto, faltam ainda itens importantes no circuito da Figura 6.

Na Figura 7, observamos os itens que ainda faltam no circuito que são: adicionar os 2-bits clássicos ao circuito e, adicionar as medidas sobre os 2-qubits, q_0 e q_1. As duas últimas linhas com o símbolo C representam os bits clássicos e são fixas. Note que, a quantidade desses bits necessários para o circuito é representada por número (neste caso, o índice é 2) e uma barra após a letra C. Também temos as medidas realizadas sobre os qubits, q_0 e q_1, onde os resultados serão armazenados nos bits clássicos. No caso ideal, após uma grande quantidade eventos, o resultado das medidas sobre os dois qubits terão duas possibilidades: com probabilidade ½, os qubits irão colapsar para os bits clássicos 00 e, com probabilidade ½, irão colapsar para os bits clássicos 11, conforme podemos observar o estado quântico da Figura 5.

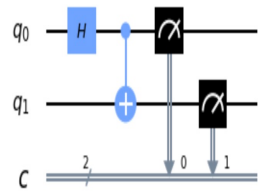
Após cada medida nos qubits, o sistema inicializa um novo circuito com os qubits nos estados padrão, $q_0: |0\rangle, q_1: |0\rangle$. Para cada inicialização chamamos de *shots*, veja a Figura 7 (a linha job). Observamos também que outras bibliotecas foram importadas para simular os resultados e visualização dos mesmos, entre as quais o Matplotlib e Qiskit Aer responsável pelo *backend* de simulador. Note que além da definição do processo de simulação, temos a implementação das medidas através da declaração *"circuit.measure([0,1],[0,1])"*. Posteriormente, executamos o circuito através do "*execute(circuit,simulador,shots=1000)"* e coletamos os resultados via "*result.get_counts(circuit)"*. Por fim, realizamos a visualização dos

mesmos através da declaração "*plot_histogram(counts)*", isto é, um gráfico de barras ou histograma.

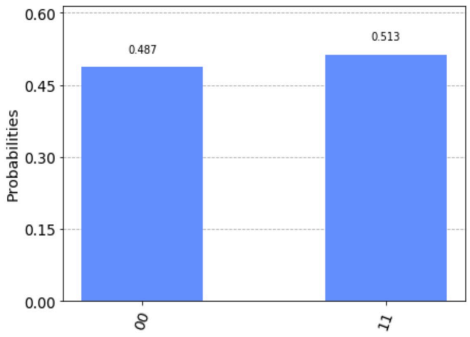
Figura 7 – Declaração do circuito quântico usando o framework Qiskit.

```
In [12]: import numpy as np
         from qiskit import(
           QuantumCircuit,
           execute,
           Aer)
         %matplotlib inline
         from qiskit.visualization import plot_histogram
         # ------------------------------------------------
            # Qasm_simulator do Qiskit Aer: Voltado
            # para simular o circuito:
         simulator = Aer.get_backend('qasm_simulator')

            # Definir um circuito quântico de 2-qubits,
            #(q_0 e q_1) e 2-bits classicos:
         circuit = QuantumCircuit(2, 2)

             #Aplicar a porta H no qubit q_0:
         circuit.h(0)

             #Aplicar a porta Cx(CNOT) nos qubits q_0(Controle),q_1 (Alvo):
         circuit.cx(0, 1)

            # Mapear os resultados das medidas quânticas para os 2-bits clássicos:
         circuit.measure([0,1], [0,1])

            # Executar o circuito nos servidores (clássicos):
            # Ou seja, executa o circuito no simulador 1000 vezes:
         job = execute(circuit, simulator, shots=1000)

            # Obtendo os resultados da variável job:
         result = job.result()

            # Contagem:
         counts = result.get_counts(circuit)
            # Imprimir na tela os resultados
         print("\nTotal da contagem de estados Quânticos para 00 e 11:",counts)
            # Esquema do Cicuito Simulado:
         circuit.draw('mpl')
```

```
Total da contagem de estados Quânticos para 00 e 11: {'11': 513, '00': 487}
```




```
plot_histogram(counts)
```



Fonte: figura gerada pelos próprios autores.

5. DISCUSSÃO DOS RESULTADOS

Neste trabalho utilizou-se um computador que possui as seguintes configurações: sistema operacional Linux, versão do kernel 5.3.0, distribuição Ubuntu 18.04.4 LTS (Long Term Support), arquitetura 64bits (x64), memória RAM de 2GB, processador Pentium Intel 3.30GHz, com HD de 500GB de armazenamento.

Na Figura 7, realizamos um circuito que produz um emaranhamento quântico de 2-qubits usando o framework Qiskit. Posteriormente, simulamos esse circuito localmente, ou seja, usando o nosso dispositivo para processamento. Inicializamos o circuito aplicando as portas e criando o estado emaranhado e, por fim, a realização das medidas nos qubits. Isso define uma rodada do circuito quântico. Repetimos esse processo 1.000 vezes para o levantamento da estatística do estado quântico. Na Figura 7, no canto superior a direita, temos a solução que é: 00: 487 e 11: 513. Esses resultados demonstram que das 1.000 rodadas, após as medidas, 487 vezes os qubits colapsaram para os bits 00 e, 513 vezes os qubits colapsaram para os bits 11. Podemos ver esses resultados através do histograma da Figura 7. Observe que nessa simulação os resultados foram muito próximos da previsão teórica, que neste caso seria: 50% dos qubits colapsando para os bits 00 e 50% para os bits 11. É necessário ressaltar que: (1) Após as medidas

quânticas, perdemos o estado de superposição quântica e os resultados que emergem das medidas são bits clássicos. (2) Observamos claramente que o framework Qiskit da plataforma IBM Q se torna um recurso pedagógico importante para a compreensão de conceitos básicos como portas e circuitos quânticos. (3) Na visualização gráfica, novamente o Qiskit pode ser usado como recurso na discussão dos resultados obtidos. Observe que esse trabalho vai colaborar com outros trabalhos já disponíveis na literatura, como em RABELO e COSTA, 2018.

6. CONCLUSÕES

Neste trabalho apresentou-se uma proposta de ensino da computação quântica por meio da aplicação de seus conceitos usando o framework Qiskit da plataforma *IBM Q Experience* (IBM Q, 2020). Primeiramente, apresentou-se a interface de usuário da plataforma *IBM Q*. Posteriormente, demonstramos como instalar o framework Qiskit em seu dispositivo de interesse, notebook ou computador pessoal. Revisamos também rapidamente conceitos como qubits, portas e circuito quântico, fornecendo um exemplo de um circuito quântico emaranhador. Após a revisão teórica, implementamos as portas e circuito quântico no Qiskit, fornecendo o código, resultado da simulação e visualização gráfica desses resultados. Concluímos que o framework Qiskit é um recurso pedagógico importante para explorar vários conceitos da computação quântica, tais como: portas, circuitos e medidas quânticas. Na visualização gráfica, o Qiskit pode auxiliar na discussão dos resultados obtidos de circuitos simples e complexos, voltados para pesquisa e ensino da computação quântica.

REFERÊNCIAS

NIELSEN, Michael Aaron.; CHUANG, Isaac. **Quantum Computation and Quantum Information.** Cambridge: Cambridge University Press. 2010.

D-Wave, Computador Quântico da Empresa D-Wave. Disponível em: https://www.dwavesys.com/our-company/meet-d-wave, Acesso em: 03 jul. 2020.

IBM Q, Computador Quântico da Empresa IBM. Disponível em: https://www.ibm.com/quantum-computing/. Acesso em: 07 jul. 2020.

Qiskit, conjunto de bibliotecas escrito em Python para a Computação Quântica. Disponível em: https://qiskit.org/. Acesso em: 07 jul. 2020.

ANACONDA, plataforma de distribuição da linguagem Python. Disponível em: http://www.anaconda.com/distribution. Acesso em: 13 jun. 2020.

JUPYTER, projeto Jupyter fornece um ambiente de desenvolvimento integrado para Python. Disponível em: https://jupyter.org/. Acesso em: 08 jul. 2020.

RABELO, Wilson R.; COSTA, Maria Lúcia M. Uma abordagem pedagógica no ensino da computação quântica com um processador quântico de 5-qubits**. Rev. Bras. Ensino de Física**, São Paulo, v. 40, n. 4, 2018.

QISKIT AND THE IBM Q PLATFORM: A CIRCUIT APPROACH FOR TEACHING OF QUANTUM COMPUTING

Abstract: *This work addresses how to use the quantum computing platform of the IBM company and its framework, known in the literature as IBM Q Experience and Qiskit, respectively. This platform supplies a set of quantum processors accessed via the cloud. These quantum processors can be accessed free of charge by researchers from different areas: physicists, mathematicians, engineers, educators. Qiskit is a development framework provided by the same company, which supplies, in addition to local simulation, the connection between the user and the quantum processor where one wishes to run quantum circuits or algorithms, aimed at teaching quantum computing. Our work aimed to provide a simple circuit that entangles 2-qubits, using Qiskit as a pedagogical resource to study basic concepts of quantum computing such as ports, circuits and, besides, we still have the process of measuring qubits available in this framework. After the theoretical review, we implemented the quantum gates and circuits in Qiskit, providing the code, the simulation result and, the graphic visualization of these results. We concluded that Qiskit proved to be an important pedagogical resource to explore various concepts of quantum mechanics and computation, such as doors, circuits, entanglement and, quantum measurements. In graphical visualization, Qiskit can assist in the discussion of the results obtained in simple and complex circuits, focused on research and teaching of quantum computing.*

Keywords: *Qiskit. IBM Q. Quantum Simulator. Quantum Computing.*