UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE FÍSICA
BACHARELADO EM ENGENHARIA FÍSICA

Natália Capra Ferrazzo

Algoritmo de Busca de Grover:

Estudo comparativo de desempenho dos *hardwares* quânticos disponibilizados pela IBM, Microsoft e Amazon.

Porto Alegre
2022

Natália Capra Ferrazzo

Algoritmo de Busca de Grover:

Estudo comparativo de desempenho dos *hardwares* quânticos disponibilizados pela IBM, Microsoft e Amazon.

Trabalho de Diplomação em Engenharia Física do Instituto de Física, da Universidade Federal do Rio Grande do Sul, como requisito para a obtenção do título de Bacharel em Engenharia Física.

Orientadora: Prof.ª Dr.ª Sandra Denise Prado
Coorientador: Prof. Dr. Wilson Ricardo Matos Rabelo

Porto Alegre
2022

AGRADECIMENTOS

Acredito ser justo começar meus agradecimentos com a pessoa responsável por me ajudar em meus primeiros passos na área de exatas: minha avó, Zaida. Foi ela que me ensinou a fazer meus primeiros cálculos matemáticos. Ela costumava preparar contas de adição e subtração em um caderno que ficava ao lado do telefone, na sua sala de estar. E eu adorava passar as tardes resolvendo-as. Logo que acabava, corria para seu colo pedindo por mais, que prontamente atendia ao pedido. Sem dúvidas essa foi a primeira atividade a despertar meu interesse na área, cujas memórias levarei comigo para sempre. Obrigada, vó. Eu te amo.

Nessa mesma época, também houve outra pessoa essencial para minha educação: meu falecido avô, Duílio. Era ele que levava e buscava eu e meu primo, Vinícius, diariamente da escolinha. Me lembro até hoje que eu e o Vini sempre discutíamos para escolher o caminho de volta para casa. Sobrava para o meu avô resolver esse impasse. Obrigada, vô. Descansa em paz. Sempre te amarei.

Passado um tempo, quando eu já estava um pouco maior, minhas tardes eram passadas na escola de informática do meu pai, Vilceu. Foi ali que surgiu minha segunda paixão: a computação. Ali também foi onde meu interesse por exatas e lógica se intensificou. Para me manter entretida enquanto trabalhava, meu pai era criativo: sempre arranjava um teste de lógica ou problema matemático diferente para que eu resolvesse. O meu favorito era o Enigma do Einstein – que mais tarde soube ser de autoria incerta. Contudo, pouco importa se o nome é de fato atribuído à pessoa correta ou não: descobrir que o alemão era o dono do peixe me consumiu um bom tempo; mas, acima disso, me introduziu à conceitos de lógica fundamentais – mesmo que, na época, eu nem entendesse o que isso significava e só estava curiosa para resolver o desafio. Obrigada, pai. Eu te amo.

Na adolescência foi a vez dela – minha mãe, Claudete – contribuir com essa jornada. O papel de uma mãe na vida de um filho é algo tão sublime, que agradecê-la com palavras se torna uma tarefa praticamente impossível. Sem seu zelo e dedicação, que começaram antes mesmo do meu nascimento, tudo isso seria impossível. Mas não vou mentir: nossa relação nem sempre foi das melhores – principalmente na época da adolescência. Contudo, ela sempre fez tudo o que pode pelo meu futuro e bem-estar, mesmo que isso significasse renunciar ao seu próprio bem-estar. Ou me obrigar a fazer inglês – coisa que hoje, com o olhar mais maduro, agradeço imensamente. Foi ela também que esteve ao meu lado no momento mais difícil da minha graduação, o último semestre. Se não fosse ela me ajudar nesse momento tão difícil, onde tanta coisa estava acontecendo, não sei se conseguiria finalizar. Obrigada mãe. Eu te amo.

Falando em tempos difíceis, quem sempre está lá para me ouvir e acolher quando preciso é minha irmã, Vanessa. Somos tão conectadas que às vezes parece que somos uma só, ela me entende como mais ninguém no mundo é capaz de entender. Obrigada, Vani. Eu te amo.

Em minha graduação pude contar com professores e mentores excepcionais para me guiar ao longo do percurso. Vale destacar aqui aquele que me introduziu à minha grande paixão, e assunto desse trabalho, a computação quântica. Foi na disciplina de Fundamentos da Mecânica Quântica que o professor Pedro Grande me fez ver a beleza e estranheza dessa área singular que está recém dando seus primeiros passos. Obrigada, prof. Pedro.

Para definir o assunto e nortear minhas pesquisas no desenvolvimento desse projeto, pude contar com o meu coorientador, professor Wilson Rabelo, que sempre se mostrou muito solícito e acessível no decorrer desse trabalho. Obrigada, prof. Wilson.

Agradeço também aos professores que aceitaram o convite para participar da banca de avaliação e me orientaram no processo de criação desse trabalho. Agradeço-lhes pelo comprometimento e apontamentos valiosos. Obrigada, prof. David. Obrigada, prof. Leonardo.

Por fim, é claro, agradeço àquela que sem dúvida alguma é o maior presente dessa jornada, cuja honra de conhecer só obtive na parte final do curso: minha orientadora, professora Sandra Prado. Além de ser uma mentora excepcional e professora extremamente dedicada, é uma pessoa com um coração enorme, cujo contato desejo manter para além da graduação. Obrigada, prof. Sandra.

RESUMO

Este trabalho visa detalhar o processo de implementação do algoritmo de busca de Grover, com foco na codificação de dados de base clássica em base quântica. Nele são detalhados os fundamentos quânticos que regem o funcionamento do algoritmo. Além disso, são apresentadas as plataformas disponíveis de forma gratuita para implementação de algoritmos quânticos da IBM, Microsoft e Amazon. Ao fim, é realizada uma comparação de desempenho entre seus os *hardwares* quânticos. Com isso, espera-se disponibilizar à academia um estudo completo e atualizado que seja acessível não só a estudantes de áreas da Física, mas também há estudantes de Engenharias, Ciências da Computação e áreas afins, visando fomentar pesquisas nos diferentes campos que podem se beneficiar com a computação quântica. Além disso, este projeto proverá à academia brasileira um primeiro passo para o desenvolvimento de metodologias de *benchmark* em *hardwares* quânticos.

Palavras-chave: Algoritmo de Busca de Grover; Computação Quântica; Codificação; *Benchmark*.

ABSTRACT

This work aims to detail the implementation process of Grover's search algorithm, focusing on the encoding of classical data into quantum data. It details the quantum fundamentals that govern the algorithm's operation. In addition, the platforms available for free for the implementation of quantum algorithms from IBM, Microsoft and Amazon are presented. Finally, a performance comparison is made between their quantum hardware. Through this, it is expected to provide the academy with a complete and updated study that is accessible not only to students from areas of Physics, but also to students of Engineering, Computer Science and related areas, aiming to promote research in different fields that can get benefit from quantum computing. In addition, it will provide Brazilian academia with a first step towards the development of benchmark methodologies in quantum hardware.

Palavras-chave: Grover's Search Algorithm; Quantum Computing; Encoding; Benchmark.

SUMÁRIO

1 INTRODUÇÃO

Como Feynman previu há 40 anos (R.P. FEYNMAN, 1982), atualmente são utilizadas diversas tecnologias baseadas na mecânica quântica. Desde componentes eletrônicos, circuitos integrados em chips semicondutores, laser, microscópios eletrônicos, LED, supercondutores até a tomografia por ressonância magnética, todos são baseados nas propriedades de sistemas de partículas quânticas e no seu controle. Exemplos concretos de fenômenos quânticos presentes em tecnologias são: o *efeito túnel* em transistores, a *coerência de fótons* em lasers, os *saltos quânticos* de elétrons em relógios atômicos. Físicos e engenheiros já se acostumaram com esses fenômenos quânticos estranhos. Eles constituem a base da primeira revolução quântica (JAEGER, 2018).

Segundo L. Jaeger (JAEGER, 2018, p. 21), em tradução livre,

> Tecnologias quânticas anteriores foram essencialmente baseadas no comportamento de sistemas quânticos de inúmeras partículas. A próxima geração de tecnologias quânticas tem sua base na manipulação dos estados de partículas únicas.

Corroborando o parecer de Jaeger, a emergente geração de tecnologias quânticas vem sendo desenvolvida com base em preparação direcionada, controle, manipulação e seleção de estados de *partículas quânticas individuais* e interações delas umas com as outras. Devido a isso, um dos fenômenos mais peculiares do mundo quântico é a estrela desse cenário: o *emaranhamento*. Através dele, é descrito como partículas quânticas podem estar em estados no qual se comportam como se estivessem "conectadas" umas às outras, mesmo que fisicamente distantes.

Para além do mundo "físico", uma das maiores benesses do entendimento da mecânica quântica está justamente nisso – entendê-la; explorar suas propriedades e aplicá-las a problemas antes abordados de maneira clássica. O algoritmo de Shor (SHOR, 1994), o algoritmo de Grover (GROVER, 1996) e o algoritmo de Deutsch-Jozsa (DEUTSCH; JOZSA, 1992) são bons exemplos disso. Ao aplicar conceitos e propriedades quânticas a algoritmos, um novo jeito de pensar e abordar problemas surge, muitas vezes trazendo ganhos de performance computacional excepcionais e abrindo possibilidade para uma nova era na computação.

1.1 CONTEXTUALIZAÇÃO

Nos últimos anos, muitos centros de pesquisa em tecnologia quântica surgiram no mundo. Empresas de alta tecnologia estão cientes das possibilidades e vislumbram as novas aplicações levantadas pela tecnologia quântica: IBM, Amazon, Google e Microsoft, por exemplo, estão reconhecendo o enorme potencial de receita e desenvolvimento e investindo fortemente em pesquisas sobre como explorar estados quânticos emaranhados e em superposição, seja por meio de parcerias com universidades (CDOTRENDS, 2022), lançamento de novos processadores com mais *qubits*[1] (IBM RESEARCH BLOG, 2021) ou promovendo competições e treinamentos internacionais (IBM RESEARCH BLOG, 2022).

Em maio de 2016, cientistas europeus assinaram o *Quantum Manisfesto*, uma iniciativa para promover a coordenação entre a academia e a indústria para pesquisar e desenvolver tecnologias quânticas na Europa ("Quantum Manifesto A New Era of Technology", 2016). O objetivo era chamar a atenção de políticos da região para o fato de que a Europa corre o risco de ficar para trás na pesquisa e desenvolvimento de tecnologias quânticas: a China atualmente domina o campo da comunicação quântica e empresas dos EUA lideram o desenvolvimento de *hardware* quântico. O objetivo do manifesto foi alcançado e logo a comissão da União Europeia decidiu promover um projeto de pesquisa em tecnologias quânticas de um bilhão de euros por ano pelos próximos 10 anos com foco em quatro áreas: comunicação, computação, sensores e simulações, com objetivo final de desenvolver um computador quântico europeu (JAEGER, 2018).

Apenas alguns meses depois, em agosto de 2016, a China lançou o primeiro satélite quântico do mundo, desenvolvido para estabelecer comunicações quânticas seguras transmitindo chaves quânticas do espaço para a Terra (YAN, 2016). Desde então, o país tem direcionado seus esforços para o aprimoramento da segurança na transmissão de informações. Em 06 de abril de 2022, foi publicado um artigo onde cientistas chineses quebraram o recorde

[1] O *qubit* é a unidade elementar da computação quântica, análogo do *bit* da computação clássica. Por meio dele, as operações computacionais básicas são desempenhadas, como será discutido mais a frente.

de distância de comunicação direta com segurança quântica (QSDC), ultrapassando 100 km de distância entre a fonte emissora e a receptora (ZHANG et al., 2022).

Nos EUA, as pesquisas tem se concentrado em aprimoramento de *hardware*. Uma das novas descobertas da área é utilizar *qubits* feitos de silício. Em um estudo recente (MILLS et al., 2021), alcançou-se um nível de *fidelidade*[2] sem precedentes usando uma porta lógica com dois desses *qubits* semicondutores. Superando os 99%, esta é a maior fidelidade alcançada até agora para uma porta de dois *qubits* em um semicondutor, e está no mesmo nível dos melhores resultados alcançados por tecnologias concorrentes, como a dos *qubits* supercondutores. Devido à grande promessa, recentemente *qubits* de silício foram fabricados em escala industrial pela primeira vez. (ZWERVER et al., 2022). Esta demonstração promete acelerar o uso da tecnologia de silício como uma alternativa viável a outras tecnologias de computação quântica.

Uma maneira de avaliar o investimento em determinado setor é através do número de requerimentos de patentes. Na **Figura 1** é possível notar que em 2018 a China teve quase o dobro de requerimentos de registro de patentes que os Estados Unidos para a tecnologia quântica em geral, uma categoria que inclui dispositivos de comunicação e criptografia. Os Estados Unidos, no entanto, lideram o mundo em requerimentos relacionados ao segmento mais valorizado do campo – computadores quânticos – graças ao pesado investimento de empresas como IBM, Amazon, Microsoft etc., (WHALEN, 2019), conforme ilustrado na **Figura 2**.

[2] A *fidelidade* é uma medida da capacidade de um *qubit* de realizar operações sem erros.

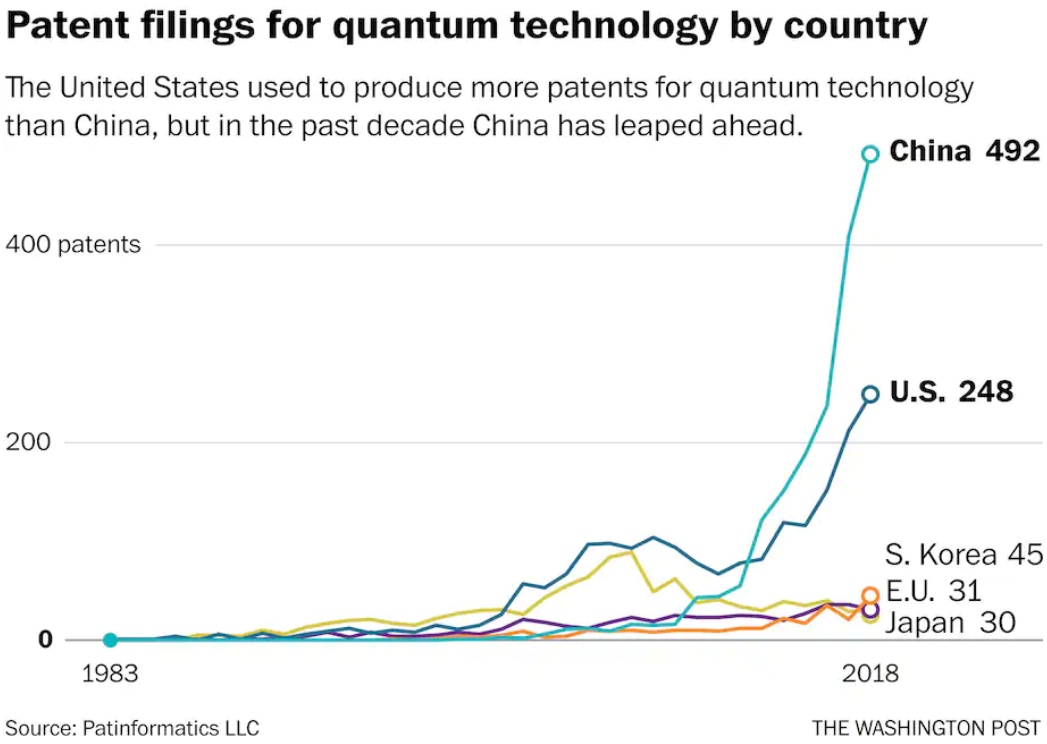

Figura 1. Requerimentos de Patentes para "*Tecnologias Quânticas*" por país.
Fonte: (PATINFORMATICS LLC, 2017).

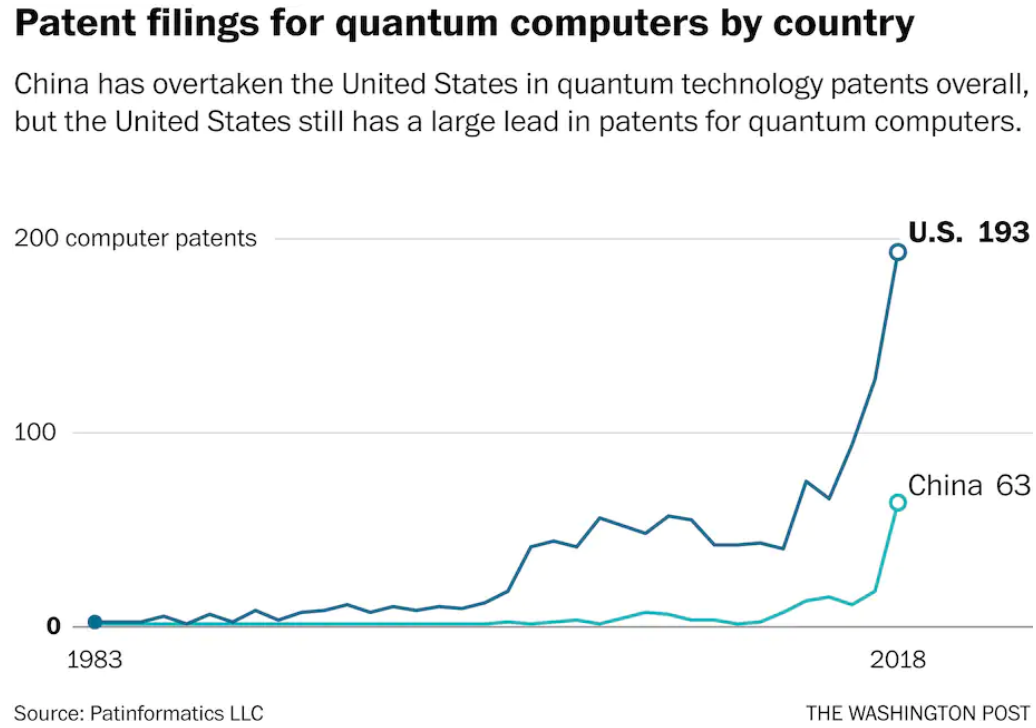

Figura 2. Requerimento de patentes para "*Computadores Quânticos*" por país.
Fonte: (PATINFORMATICS LLC, 2017).

Apesar do destaque desses países, o investimento no setor não é algo pontual. Diversos países estão adotando iniciativas de incentivo a essas tecnologias, como é possível notar no levantamento feito pela QuRECA (QURECA, 2022). Nele são elencados os investimentos públicos realizados por cada país, que totalizam quase US$ 30 bilhões, conforme ilustrado na **Figura 3**.

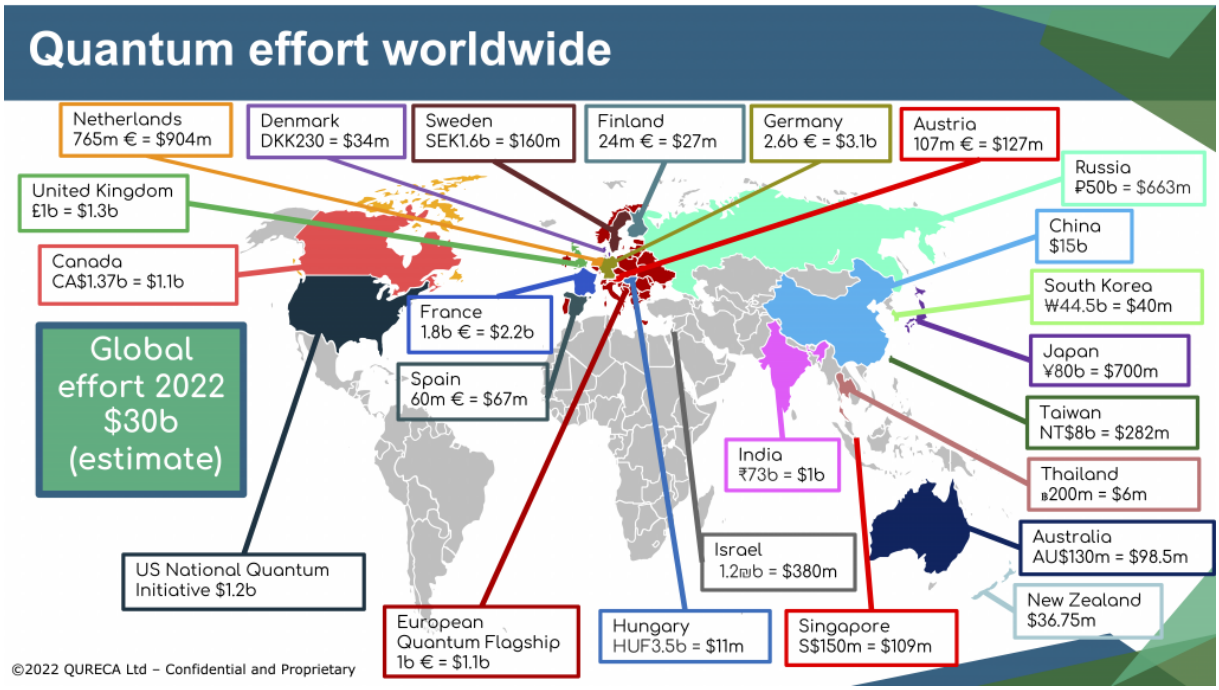

Figura 3. Panorama do financiamento público em tecnologias quânticas.
Fonte: (QURECA, 2022)

Como fica evidente pelo infográfico, no Brasil ainda não há esforço das autoridades públicas federais para aderir a essa tendência. Apesar disso, algumas universidades do país vêm estabelecendo parcerias e desenvolvendo iniciativas para tentar cobrir essa carência.

1.2 PROBLEMA TÉCNICO

Muitos dos problemas que os computadores resolvem são tipos de problemas de busca. Uma busca na web usando um mecanismo de pesquisa (como o buscador *www.google.com*) nada mais é do que um programa que cria um banco de dados a partir de sites e permite que o usuário pesquise nele. Um banco de dados pode ser interpretado como um programa que recebe um endereço como entrada e retorna os dados contidos nesse endereço. Uma lista telefônica é um exemplo de banco de dados: cada entrada no livro contém um nome e um número. Por exemplo, pode-se requisitar ao banco de dados que nos forneça os dados no 410º endereço e ele retornará o 410º nome e número no livro.

Para efetuar uma busca em um banco de dados na computação clássica, é necessário checar em média $^{N}/_{2}$ entradas até encontrar o endereço requisitado, onde N corresponde ao número de entradas do banco de dados. Ou seja, a complexidade temporal de algoritmos de busca clássicos é $O(N)$. Com a computação quântica, essa complexidade sofre uma redução quadrática: um hardware quântico, através do algoritmo de busca de Grover, efetua uma busca em um banco de dados em $O(\sqrt{N})$.

Para implementar o algoritmo de Grover, é necessário codificar a informação do banco de dados em uma base quântica. Embora seja possível encontrar trabalhos sobre implementações do algoritmo de Grover em língua portuguesa (MICROSOFT, 2022; PRADO; DILLENBURG, 2014; QISKIT, 2021a) há uma defasagem no estudo de estratégias de codificação da informação – etapa necessária para transformar uma base clássica em uma base quântica. Por essa razão, esse estudo se concentrará principalmente nessa parte.

Embora a computação quântica tenha um grande potencial para mudar a maneira como diversos problemas são abordados, o grande desafio dos computadores quânticos atuais, sem dúvida, é referente a *estabilidade*. Os *hardwares* quânticos disponíveis atualmente são ruidosos – apresentam elevada taxa de *ruido quântico*, o que gera perda de performance no sistema. Além disso, existe a *decoerência*, que faz com que os *qubits* percam suas propriedades de superposição e emaranhamento, limitando a complexidade dos cálculos quânticos. Isso gera erros de medida em comparação com a medida teórica esperada.

1.3 OBJETIVOS

1.3.1 Objetivo Geral

O objetivo geral deste trabalho é prover um estudo comparativo entre os diferentes *hardwares* quânticos disponíveis atualmente, analisando sua *performance*, *nível de ruído* e *tempo de processamento*, além de disponibilizar uma pesquisa detalhada e confiável da implementação do algoritmo de busca de Grover para a comunidade acadêmica brasileira buscando, com isso, fomentar o estudo de algoritmos quânticos neste meio e contribuindo para a disseminação do conhecimento sobre o assunto.

1.3.2 Objetivos Específicos

Os objetivos específicos desta pesquisa são:

a) Apresentar um estudo detalhado da implementação do algoritmo de Grover, acessível ao público de qualquer área das Ciências Exatas, cobrindo a fundamentação teórica, mecanismos de funcionamento e portas lógicas necessárias para sua implementação.
b) Descrever as diferentes estratégias de codificação de informação em sistemas de *n-qubits* que descrevem um estado.
c) Implementar o algoritmo de Grover nos ambientes de computação quântica disponíveis da IBM, Microsoft e Amazon.
d) Analisar a performance dos *hardwares* quânticos disponibilizados pelas empresas, comparando seus desempenhos na implementação do algoritmo de busca.
e) Detalhar todo o processo para que esse estudo seja uma base de confiança para pesquisas acadêmicas brasileiras futuras nos campos das Ciências Exatas.

2 CONCEITOS TEÓRICOS

A mecânica quântica é a base para o entendimento da computação e informação quântica. Para entender seus postulados e implicações, certa familiaridade com conceitos de álgebra linear se fazem necessários.

2.1 ÁLGEBRA LINEAR E OS FUNDAMENTOS DA COMPUTAÇÃO QUÂNTICA

Álgebra linear é o estudo de espaços vetoriais e de operações lineares nesses espaços. Uma boa compreensão da mecânica quântica é baseada em um sólido entendimento de álgebra linear elementar (NIELSEN; CHUANG, 2010). Como aqui a mecânica quântica é a motivação para o estudo de álgebra linear, é interessante fazer uso da *Notação de Dirac* (DIRAC, 1939). As notações mais usuais, em conjunto com suas descrições, estão indicadas na **Tabela 2.1**.

O objeto de estudo da álgebra linear são os *espaços vetoriais*. Dentre eles, o de maior interesse neste trabalho é $\mathbb{C}^n$, espaço contendo todos os n-vetores de números complexos, $(z_1, \ldots, z_n)$.

Tabela 2.1 Principais notações utilizadas em mecânica quântica, conhecidas como Notações de Dirac.

Notação	Descrição
z	Número complexo formado por $z \equiv x + iy$ onde $x, y \in \mathbb{R}$ e $i \equiv \sqrt{-1}$.
z^*	Complexo conjugado do número complexo z. $(x + iy)^* = x - iy$
$\|\psi\rangle$	Vetor *ket*.
$\langle\psi\|$	Vetor *bra*. É o vetor dual a $\|\psi\rangle$.
$\langle\varphi\|\psi\rangle$	Produto interno entre os vetores $\|\varphi\rangle$ e $\|\psi\rangle$.
$\|\varphi\rangle\otimes\|\psi\rangle$	Produto tensorial de $\|\varphi\rangle$ e $\|\psi\rangle$.
$\|\varphi\rangle\ \|\psi\rangle$	Notação abreviada para o produto tensorial de $\|\varphi\rangle$ e $\|\psi\rangle$.
A^*	Matriz complexa conjugada da matriz *A*.
A^T	Matriz transposta da matriz *A*.
$A^\dagger$	Conjugado Hermitiano ou Adjunto da matriz *A*. Onde $A^\dagger = (A^T)^*$.

2.1.1 Bases e *Qubit*

Uma *bas*e de um espaço vetorial é um conjunto de vetores $|v_1\rangle, \dots, |v_n\rangle$ com os quais qualquer vetor $|v\rangle$ deste espaço pode ser escrito como uma combinação linear dos vetores do conjunto.

$$|v\rangle = \sum_i a_i |v_i\rangle. \tag{2.1}$$

Para o espaço vetorial $\mathbb{C}^2$, necessário para descrever um *qubit*, uma base é dada pelo conjunto

$$|v_1\rangle \equiv \begin{bmatrix} 1 \\ 0 \end{bmatrix};$$

$$|v_2\rangle \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

de modo que qualquer vetor do espaço $\mathbb{C}^2$ pode ser escrito a partir de combinações lineares de $|v_1\rangle e\ |v_2\rangle$, por meio da expressão

$$|v\rangle = \ a_1|v_1\rangle + \ a_2|v_2\rangle. \tag{2.2}$$

É importante destacar que, de modo geral, um espaço vetorial possui inúmeras bases diferentes. Contudo, para o objeto de estudo deste trabalho, a base supracitada é a mais usual. Esta base é tão importante na computação quântica que recebe rótulos especiais, que fazem alusão à computação clássica:

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

$$|1\rangle \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Estes estados são a *base computacional* da computação quântica e formam uma base ortonormal para este espaço vetorial.

Ao contrário da computação clássica, onde *bits* só assumem os valores *0* ou *1*, *qubits* podem estar em *combinações lineares* destes estados, chamados de *estados de superposição* e expressos pela equação:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle. \tag{2.3}$$

Deste modo, um espaço vetorial quântico, aqui um *qubit*, é representado por $|\psi\rangle$, onde α e β são números complexos que representam a *amplitude* associada a cada estado, $|0\rangle$ e $|1\rangle$.

É possível notar que, na verdade, um *qubit* pode estar em um espectro *contínuo* de estados entre $|0\rangle e\ |1\rangle$. Além disso, devido aos postulados da teoria quântica, é impossível saber o estado de um *qubit* antes de medi-lo e, após medi-lo, o estado *colapsa* para $|0\rangle$ ou $|1\rangle$.

Dois aspectos importantes estão relacionados à amplitude. Para isso, α e β devem ser normalizados:

$$|\alpha|^2 + |\beta|^2 = 1. \tag{2.4}$$

Um deles é a *magnitude*, dada por $|\alpha|^2$ e $|\beta|^2$, e que representa, respectivamente, a *probabilidade* de, após a medida, o estado obtido seja $|0\rangle$ e $|1\rangle$.

O outro é a *fase relativa*, determinada a partir das partes imaginárias de α e β, que representa o grau em que diferentes caminhos computacionais interferem construtiva ou destrutivamente.

Dessa forma, quando um *qubit* no estado

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

é medido, irá colapsar para o estado $|0\rangle$ em 50% das vezes e para $|1\rangle$ nas outras vezes.

Apesar de toda estranheza, *qubits* são reais e sua existência e comportamento já foram extensivamente validados por experimentos e sistemas físicos diversos (NIELSEN; CHUANG, 2010, cap. 7). A maneira mais usual atualmente de se obter um *qubit* é a partir de diferentes polarizações em fótons. Outras maneiras são, por exemplo, através do alinhamento do *spin* nuclear em campos magnéticos uniformes e incidindo luz sobre um átomo de maneira a alterar seu estado – por exemplo, do *estado fundamental* para o *estado excitado*, que podem ser definidos como $|0\rangle$ e $|1\rangle$. O interessante é que, ao incidir luz por tempo adequado, o elétron

pode ficar em um estado de superposição ou “no meio do caminho” entre $|0\rangle$ e $|1\rangle$, o que gera a característica probabilística de estados do *qubit* (NIELSEN; CHUANG, 2010).

2.1.1.1 A esfera de Bloch

Utilizando o fato de $|\alpha|^2 + |\beta|^2 = 1$, é possível reescrever a Equação (2.3) para representar um *qubit* através da forma polar de números complexos:

$$|\psi\rangle = cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\varphi}\, sin\left(\frac{\theta}{2}\right)|1\rangle. \tag{2.5}$$

onde θ e φ são números reais. O termo $e^{i\gamma}$ que surge na transformação pode ser ignorado devido ao fato de não ter efeitos observáveis (NIELSEN; CHUANG, 2010, cap. 1, p. 15).
Isso permite sua visualização em um sistema de referência tridimensional, denominada *Esfera de Bloch*, como ilustrado na **Figura 4**.

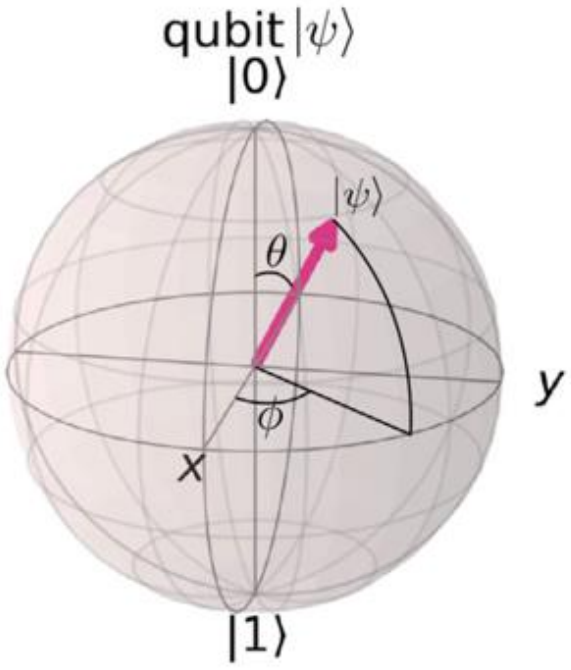

Figura 4. Representação de um estado genérico na esfera de Bloch

2.1.2 Operadores Lineares e Portas Quânticas

Um *operador linear* é uma função que aplica uma transformação linear de um espaço vetorial nele mesmo, transformando cada um de seus vetores linearmente (MILLER, 2008), segundo

$$A\left(\sum_i a_i\,|v_i\rangle\right) = \sum_i a_i\,A\,(|v_i\rangle). \tag{2.6}$$

A maneira mais conveniente de trabalhar com operadores é através de sua representação matricial. Nem todo operador linear, contudo, pode ser uma porta quântica: devido à normalização, apenas operadores *unitários* podem ser aplicados à *qubits*, já que estes são *reversíveis*. Eles operam da seguinte forma:

$$U|\psi\rangle = U[\alpha|0\rangle + \beta|1\rangle] = \alpha U|0\rangle + \beta U|1\rangle. \tag{2.7}$$

Existem quatro matrizes que são operadores quânticos – e, portanto, portas quânticas – extremamente úteis: as *Matrizes de Pauli*. Elas são matrizes 2×2 e recebem notações especiais conforme sua aplicabilidade, ilustradas abaixo juntamente com sua forma na notação de Dirac:

[1] $I \equiv \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = |0\rangle\langle 0| + |1\rangle\langle 1|;$

[2] $\sigma_x \equiv X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = |0\rangle\langle 1| + |1\rangle\langle 0|;$

[3] $\sigma_y \equiv Y \equiv \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = -i|0\rangle\langle 1| + i|1\rangle\langle 0|;$

[4] $\sigma_z \equiv Z \equiv \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = |0\rangle\langle 0| - |1\rangle\langle 1|.$

Suas representações na mecânica quântica usualmente recebem letras gregas com índices (σ_x, σ_y, σ_z). Já para circuitos quânticos, letras arábicas são preferidas (*X*, *Y* e *Z)*.

A operação **[1]** corresponde à *Identidade*. Ela é útil em cálculos de equações matriciais. Porém, seu resultado prático é nulo: ao aplicar a matriz em um vetor/estado, o resultado é o próprio vetor/estado.

A operação **[2]**, *Pauli-X*, por outro lado, possui um papel primordial para o funcionamento da computação quântica: corresponde à porta lógica clássica NOT, invertendo o estado do *qubit* de $|0\rangle$ para $|1\rangle$ e vice-versa. Graficamente, é possível interpretar a atuação desse operador como uma rotação de π radianos em torno do eixo *x*, como mostra a **Figura 5**.

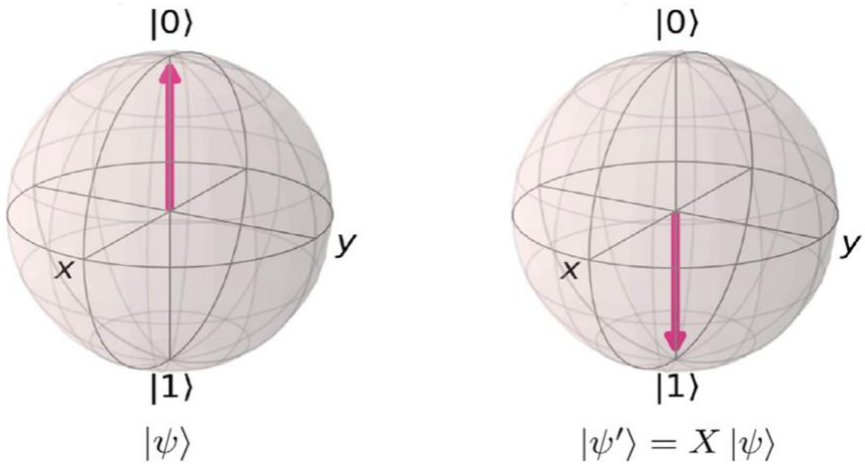

Figura 5. Porta X aplicada a um *qubit* no estado $|0\rangle$: $X|0\rangle = |1\rangle$

Já a matriz **[3]**, *Pauli-Y*, atua alterando tanto o estado quanto a fase do *qubit*. De maneira análoga ao operador anterior, atua rotacionando o estado do *qubit* em π radianos em torno do eixo *y*.

A última, *Pauli-Z*[3], deixa inalterado o estado $|0\rangle$, mudando apenas a fase de $|1\rangle$. Assim como as demais, também rotaciona em π radianos o estado do *qubit* em torno do eixo *z*.

Outra porta quântica útil é a *Hadamard*. É ela que cria o estado de *superposição* e, consequentemente, possibilita o *emaranhamento*. Seu operador unitário é:

---

[3] É interessante notar que os elementos $\{|0\rangle,|1\rangle\}$ são *autoestados* desse operador. Por essa razão, *medidas* de circuitos quânticos usualmente recebem o apelido de "*z-measurement*".

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

Ao aplicá-la, por exemplo, em um *qubit* no estado $|\psi\rangle = |0\rangle$, obtém-se:

$$|\psi'\rangle = \frac{1}{\sqrt{2}}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|)(|0\rangle);$$

$$|\psi'\rangle = \frac{1}{\sqrt{2}}(|0\rangle\langle 0|0\rangle + |0\rangle\langle 1|0\rangle + |1\rangle\langle 0|0\rangle - |1\rangle\langle 1|0\rangle);$$

$$|\psi'\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\ 1\end{bmatrix}.$$

Ou seja, após aplicar o operador quântico *H*, o *qubit* estará em estado de *superposição*, expresso pela equação abaixo e que pode ser visualizado na **Figura 6**.

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

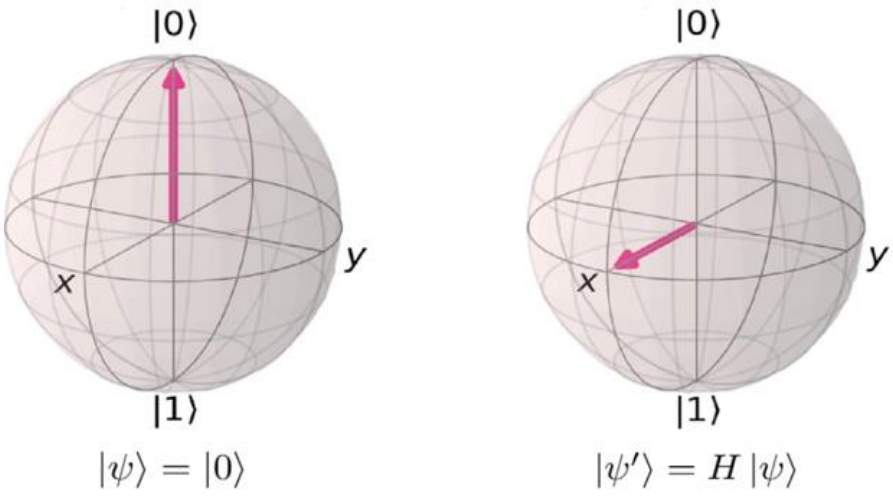

Figura 6. Ilustração da porta Hadamard aplicada a um *qubit* inicialmente no estado |0⟩: H|0⟩.

Como esse estado, juntamente com seu análogo, são autoestados de σ_x, recebem rótulos especiais:

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle); \qquad |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

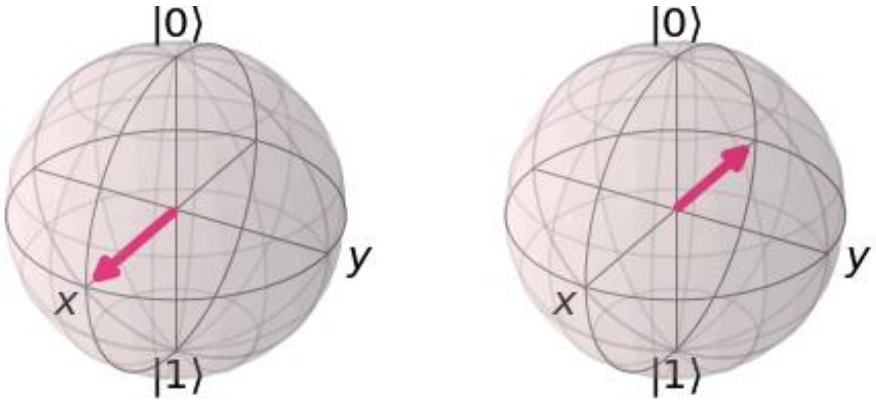

Figura 7. Ilustração dos estados $|+\rangle$ e $|-\rangle$.

Assim como os autoestados de σ_x e σ_z, os autoestados de σ_y também recebem notações específicas:

$$|+i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle); \qquad |-i\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle).$$

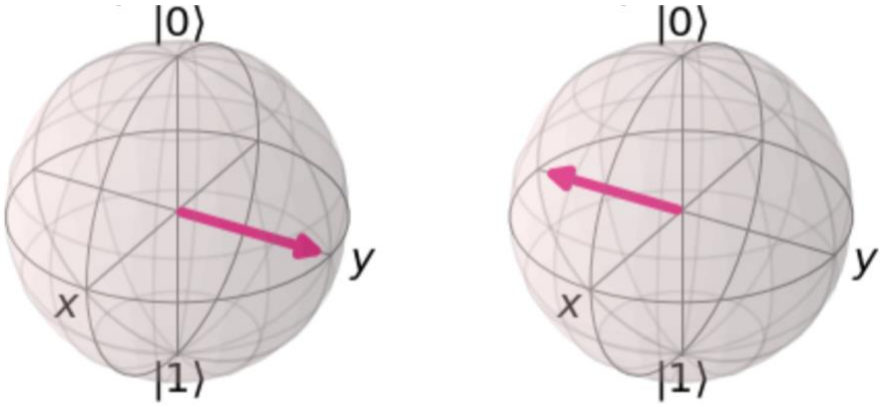

Figura 8. Ilustração dos estados $|+i\rangle$ e $|-i\rangle$.

2.1.3 Múltiplos *Qubits* e os Estados de Bell

Apesar de um único *qubit* já possuir um comportamento excepcional, ao lidar com mais de um *qubit* novas possibilidades surgem. A mais notável de todas, sem dúvida, é o *emaranhamento*. Para isso, é preciso discutir o conceito de *registrador quântico*.

Um registrador quântico de tamanho *n* compreende um sistema quântico com *n qubits*, onde cada *qubit* q_i com $i \in \{0, \dots, n-1\}$ é representado por um vetor unitário do espaço de Hilbert[4] $\mathcal{H}_i$ com $i \in \{0, \dots, n-1\}$. Com isso, o registrador quântico resultante é representado por um vetor unitário *n*-dimensional do espaço de Hilbert, calculado através do produto tensorial dos vetores primários:

$$\mathcal{H} = \mathcal{H}_{n-1} \otimes \mathcal{H}_{n-2} \otimes \cdots \otimes \mathcal{H}_0. \tag{2.8}$$

Portanto, da mesma forma que *n bits* tem 2^n estados possíveis, a *base computacional* para o espaço de Hilbert de *n qubits* terá dimensão 2^n.

De forma geral, o estado de um sistema quântico é expresso pela equação:

$$|\psi\rangle = \sum_{i=1}^{2^n} \alpha_i |i\rangle. \tag{2.9}$$

Assim, dado um circuito com 2 *qubits*, o estado do sistema é representado por:

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

Por exemplo, se um *qubit* q_A está no estado $|1\rangle_A$ e um *qubit* q_B está no estado $|0\rangle_B$ o estado do sistema é:

$$|\psi\rangle = |1\rangle_A \otimes |0\rangle_B = |10\rangle_{AB} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}.$$

[4] O *Espaço de Hilbert* é uma variação do espaço complexo que possui propriedades especiais abordadas por Griffiths et. al, (GRIFFITHS; SCHROETER, 2001, p. 118-121).

Para se obter um estado de *emaranhamento*, além da porta Hadamard, a porta *CNOT* (também chamada de *C–X*) é necessária. A porta CNOT (*Controlled NOT*) opera em dois *qubits*: um *qubit* de controle, *control qubit*, e um *qubit* alvo, *target qubit*. Essa porta aplica a operação lógica NOT ao *target qubit* somente se o *control qubit* tiver o valor 1. O operador unitário dessa porta é o seguinte:

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} = |00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 11| + |11\rangle\langle 10|.$$

Ao aplicar esse operador em um circuito no estado $|00\rangle$, por exemplo, é possível notar que nada acontece. Já ao aplicá-lo ao estado $|10\rangle$, o estado $|11\rangle$ é retornado.

Porém o resultado mais notável deste operador é alcançado ao aplicá-lo a sistemas com *qubits* em estados de superposição, como por exemplo, o obtido anteriormente, ao aplicar a porta Hadamard no estado $|0\rangle$: $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$.

Se esse *qubit* está em um sistema de dois *qubits*, onde o outro foi deixado no estado inicial $|0\rangle$, estado do sistema é descrito por:

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle;$$

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle + |1\rangle \otimes |0\rangle);$$

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle).$$

Com isso, é possível calcular o resultado do operador *CNOT* aplicado ao sistema:

$$|\psi'\rangle = CNOT|\psi\rangle;$$

$$|\psi'\rangle = \frac{1}{\sqrt{2}}(|00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 11| + |11\rangle\langle 10|)(|00\rangle + |10\rangle);$$

$$|\psi'\rangle = \frac{1}{\sqrt{2}}(|00\rangle\langle 00|00\rangle + |01\rangle\langle 01|00\rangle + |10\rangle\langle 11|00\rangle + |11\rangle\langle 10|00\rangle)$$
$$+ \frac{1}{\sqrt{2}}(|00\rangle\langle 00|10\rangle + |01\rangle\langle 01|10\rangle + |10\rangle\langle 11|10\rangle + |11\rangle\langle 10|10\rangle);$$

$$|\psi'\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

Esse estado, juntamente com suas três variações formam uma base ortonormal e são conhecidos como os *Estados de Bell*[5]. Por representarem a forma de *emaranhamento máximo* entre dois *qubits*, recebem notações especiais:

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

$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle);$$

$$|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle);$$

$$|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle).$$

[5] As vezes também chamados de *estados EPR*, em homenagem a Einstein, Podolsky e Rosen – que, juntamente com Bell, foram os primeiros a notar as propriedades peculiares desses estados.

2.1.4 Fase Relativa e Retorno de Fase

Não é apenas a porta X que tem uma versão controlada – é possível acoplar o controle em qualquer porta. Dado um operador genérico U, cuja matriz é:

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

o operador *Controlled-U* será:

$$CU = \begin{bmatrix} I & 0 \\ 0 & U \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{bmatrix}.$$

Um comportamento peculiar ocorre na porta *Controlled*-Z – e o algoritmo de Grover se vale justamente desse mecanismo. Para entendê-lo, é útil saber que a porta Z é derivada do operador *Fase*, substituindo $\lambda = \pi$ na matriz abaixo:

$$Fase = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{bmatrix}.$$

Ao substituir $\lambda = {}^{\pi}/_{4}$ tem-se a porta T, que equivale a uma rotação de ${}^{\pi}/_{4}$ em torno do eixo Z. Ou seja, ao aplicar a porta T a um *qubit* no estado $|1\rangle$, uma *fase* de ${}^{\pi}/_{4}$ é adicionada a esse *qubit*:

$$T|1\rangle = e^{i\pi/4}|1\rangle.$$

Esta é uma *fase global* e não é observável. Mas ao controlar esta operação usando outro *qubit* em um estado de *superposição*, como por exemplo no estado $|+\rangle$, a fase não é mais *global*, e sim *relativa*, o que altera a *fase relativa* do *qubit* de controle:

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

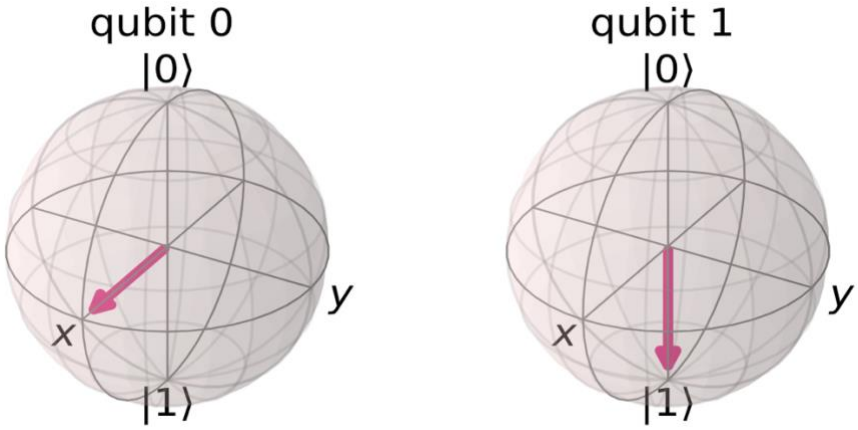

Figura 9. Ilustração do sistema no estado $|1+\rangle$.

$$CT|1+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + e^{i\pi/4}|11\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}|1\rangle) \otimes |1\rangle;$$

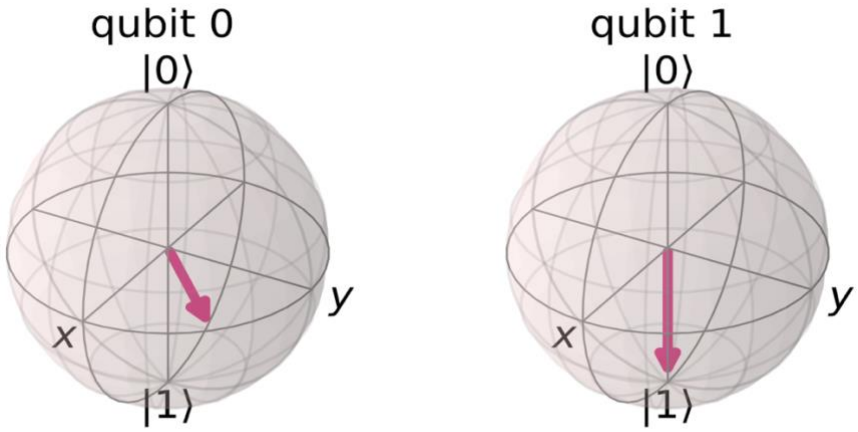

Figura 10. Ilustração do sistema após aplicação da porta C-T: $C-T|1+\rangle$.

Ou seja: aplicar essa porta controlada produz o efeito de girar o *qubit* de controle em torno do eixo Z da esfera Bloch, deixando o *qubit* alvo inalterado! Esse efeito é chamado de *Retorno de Fase* e é fundamental para o algoritmo de Grover.

2.2 ALGORITMO DE BUSCA DE GROVER

Um dos algoritmos quânticos mais famosos, o algoritmo de Grover, foi proposto por Lov Grover em 1996 (GROVER, 1996). Ele pertence à classe de algoritmos de busca quântica, que tem como finalidade resolver o seguinte problema: Dado um banco de dados de N elementos e nenhuma informação prévia sobre a estrutura de sua informação, deseja-se encontrar um elemento desse banco que satisfaça determinada condição, um elemento "marcado". Por exemplo, suponha uma lista com N caixas. Entre essas caixas, existe a caixa ω com uma propriedade única que se deseja localizar: enquanto todas as demais caixas são cinzas, ela é roxa, conforme ilustrado na **Figura 11**.

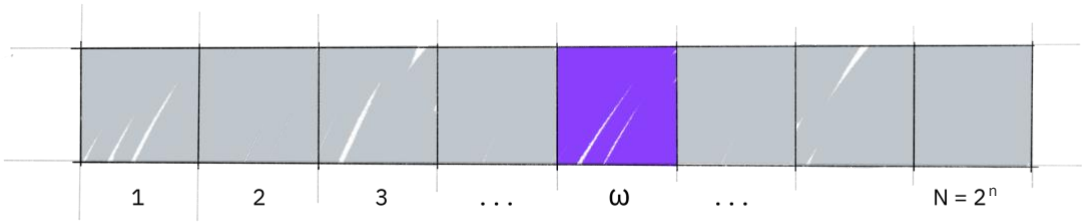

Figura 11. Ilustração de N caixas com a caixa ω marcada.
Fonte: (QISKIT, 2021b)

Este problema é chamado de *Pesquisa Não Estruturada*. Para encontrar a caixa roxa – o item marcado – usando computação clássica, é necessário verificar em média $N/2$ caixas e, na pior das hipóteses, todas elas – o que resulta em uma complexidade temporal de $O(N)$. Já o algoritmo de busca quântica requer apenas aproximadamente $\sqrt{N}$ operações para resolvê-lo, mediante o método de *amplificação de amplitude* de Grover. Uma aceleração quadrática, que representa uma economia de tempo substancial para encontrar itens marcados em listas longas. Além disso, o algoritmo não utiliza a estrutura interna da lista, o que o torna genérico.

2.2.1 Codificação de Informações

Antes de partir para o algoritmo, é necessário realizar a codificação de informações em um sistema de *n-qubits* descrito por um estado descrito pela expressão (2.9). Existem diversas estratégias disponíveis para essa tarefa que podem ser divididas em 4 tipos, de acordo com (SCHULD; PETRUCCIONE, 2018a): *codificação de base*, *codificação de amplitude*, *codificação qsample* e *codificação dinâmica*. A **Tabela 2.2** sumariza estes quatro tipos de codificação.

Tabela 2.2 Resumo dos diferentes tipos de codificação de dados.

Dados clássicos	**Propriedades**	**Estados quântico**
Codificação de base		
$(\boldsymbol{b_1}, \ldots, \boldsymbol{b_N}),\ \boldsymbol{b_i} \in \{\boldsymbol{0}, \boldsymbol{1}\}$	b codifica $x \in \mathbb{R}^N$ em binário	$\lvert x\rangle = \lvert b_i, \ldots, b_{N-1}\rangle$
Codificação de amplitude		
$\boldsymbol{x} \in \mathbb{R}^{\boldsymbol{2^n}}$	$\sum_{i=1}^{2^n} \lvert x_i\rvert^2 = 1$	$\lvert\psi_x\rangle = \sum_{i=1}^{2^n} x_i \lvert i\rangle$
$\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{2^n \times 2^m}}$	$\sum_{i=1}^{2^n} \sum_{j=1}^{2^m} \lvert a_{ij}\rvert^2 = 1$	$\lvert\psi_A\rangle = \sum_{i=1}^{2^n} \sum_{j=1}^{2^m} a_{ij} \lvert i\rangle \lvert j\rangle$
$\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{2^n \times 2^n}}$	$\sum_{i=1}^{2^n} a_{ii} = 1, a_{ij} = a_{ji}^*$	$\rho_A = \sum_{ij} a_{ij} \lvert i\rangle\langle j\rvert$
Codificação qsample		
$\boldsymbol{p(x)}, \boldsymbol{x} \in \{\boldsymbol{0}, \boldsymbol{1}\}^{\otimes \boldsymbol{n}}$	$\sum_x p(x) = 1$	$\sum_x \sqrt{p(x)} \lvert x\rangle$
Codificação dinâmica		
$\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{2^n \times 2^n}}$	Matriz unitária A	U_A com $U_A = A$
$\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{2^n \times 2^n}}$	Hermitiano A	H_A com $H_A = A$
$\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{2^n \times 2^n}}$	-	$H_{\tilde{\mathrm{A}}}$ com $\tilde{\mathrm{A}} = \begin{pmatrix} 0 & A \\ A^\dagger & 0 \end{pmatrix}$

2.2.1.1 Codificação de Base

A codificação da base é a mais simples e usual de todas. Ela associa uma base computacional binária clássica ao estado quântico de *n-qubits*. Dessa forma, cada *bit* é diretamente substituído por um *qubit*. Ou seja, a codificação de base utiliza uma representação binária para representar números reais, tal qual ocorre na computação clássica.

Com esse tipo de codificação, o valor absoluto ao quadrado das amplitudes dos estados da base representa a probabilidade de mensuração daquele estado. Dessa forma, para implementações que utilizam a codificação de base, o objetivo do algoritmo quântico é aumentar a amplitude do estado da base que corresponde à solução codificada.

Existem diversas maneiras de representar um número real na forma binária, que podem ser consultadas em (DA CUNHA, 2017). Uma delas é a *representação binária de fração*, onde cada número real do intervalo [0,1) é representado por uma sequência de τ-*bits* tal que:

$$r = \sum_{k=1}^{\tau} b_k \frac{1}{2^k}, \tag{2.10}$$

onde τ representa a precisão, b o coeficiente da codificação. Então, por exemplo, para codificar um vetor $x = (0.1, -0.4, -1.0)$ em representação binária, com precisão $\tau = 4$ e o primeiro bit da sequência representando o sinal, tem-se:

$$0.1 \rightarrow 0\ 0001;$$

$$-0.4 \rightarrow 1\ 0110;$$

$$-1.0 \rightarrow 1\ 1111.$$

Dessa maneira, na computação quântica o vetor x seria representado pelo estado:

$$b = |00001\ 10110\ 11111\rangle.$$

2.2.1.2 Codificação de Amplitude

A codificação de amplitude, menos comum na computação quântica, associa informações clássicas às amplitudes quânticas. Existem diversas formas de efetuar essa codificação. É possível, por exemplo, representar um vetor clássico normalizado $x \in \mathbb{C}^{2^n}$, $\sum_k |x_k|^2 = 1$, pelas amplitudes de um estado quântico $|\psi\rangle$:

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_{2^n} \end{bmatrix} \leftrightarrow |\psi_x\rangle = \sum_{i=1}^{2^n} x_i |i\rangle. \tag{2.11}$$

De maneira análoga, uma matriz $A \in \mathbb{C}^{2^n \times 2^m}$, com componentes a_{ij} que satisfaçam a normalização $\sum_{ij} a_{ij} = 1$, podem ser codificados através de:

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1j} \\ \vdots & \ddots & \vdots \\ a_{i1} & \cdots & a_{ij} \end{bmatrix} \leftrightarrow |\psi_A\rangle = \sum_{i=1}^{2^m} \sum_{j=1}^{2^n} a_{ij} |i\rangle|j\rangle. \tag{2.12}$$

Para matrizes hermitianas $A \in \mathbb{C}^{2^n \times 2^n}$, de traço[6] $tr(A) = 1$, existe outra opção: é possível associar seus elementos com elementos de uma matriz densidade ρ_A tal que $a_{ij} \leftrightarrow \rho_{ij}$.

Uma restrição desse método é que apenas vetores clássicos *normalizados* podem ser codificados. Isso implica que, ao codificar um vetor dessa forma, os estados quânticos representarão os dados em uma dimensão a menos – ou seja, com um grau de liberdade a menos. Um vetor bidimensional clássico (x_1, x_2), por exemplo, só pode ser associado a um vetor de amplitude (α_1, α_2) de um *qubit* que satisfaz $|\alpha_1|^2 + |\alpha_2|^2 = 1$, o que representa um círculo unitário (uma forma unidimensional em um espaço bidimensional), conforme ilustrado na **Figura 12**.

[6] O traço é definido como a soma dos elementos da diagonal de uma matriz quadrada. Assim, uma matriz A de tamanho $n \times n$ terá o traço definido por: $tr(A) = a_{11} + a_{22} + \ldots + a_{nn}$.

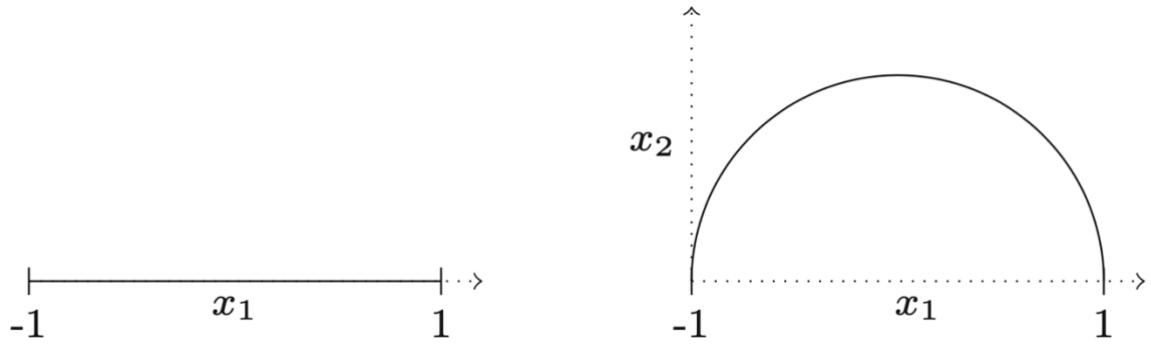

Figura 12. Os pontos de dados no intervalo unidimensional [−1, 1], à esquerda, podem ser projetados em vetores normalizados adicionando um valor constante em uma segunda dimensão x_2 e renormalizando.
Fonte: (SCHULD; PETRUCCIONE, 2018b)

Uma maneira de contornar a perda de grau de liberdade ocasionada por essa codificação é aumentando o espaço do vetor clássico em uma dimensão $x_{N+1} = 1$ e normalizar o vetor resultante. O espaço N–dimensional será, então, embutido em um espaço N+1–dimensional no qual os dados são normalizados sem perda de informação (SCHULD; PETRUCCIONE, 2018b).

Assim, para representar o mesmo vetor anterior $x = (0.1, -0.4, -1.0)$, em codificação de amplitude, primeiro é preciso normalizá-lo e depois preenchê-lo com zeros para a dimensão apropriada:

$$x' = (0.085, -0.342, -0.855, 0.000),$$

e então, representá-lo por um estado quântico de 2 *qubits*:

$$0.085|00\rangle - 0.342|01\rangle - 0.855|10\rangle + 0.000|11\rangle.$$

É interessante notar que esse mesmo estado também codifica a matriz A:

$$A = \begin{pmatrix} 0.085 & -0.342 \\ -0.855 & 0.000 \end{pmatrix}$$

para o caso em que o primeiro *qubit* representa o índice para a linha e o segundo *qubit* representa o índice para a coluna.

2.2.1.3 Codificação *Qsample*

A codificação *qsample* associa um vetor de amplitude real $\left(\sqrt{p_1}, \ldots, \sqrt{p_N}\right)^T$ com uma distribuição de probabilidade discreta clássica $(p_1, \ldots, p_N)$. Este tipo de codificação pode ser interpretado como um caso híbrido de codificação de base e amplitude, já que a informação de interesse é representada por amplitudes, mas os N elementos são codificados nos *qubits*:

$$|\psi\rangle = \sum_{i=1}^{2^n} \sqrt{p_i}|i\rangle. \tag{2.13}$$

2.2.1.4 Codificação Dinâmica

Para algumas aplicações pode ser útil codificar matrizes na forma dinâmica, por exemplo, em operadores unitários. Uma maneira é associar um Hamiltoniano H com uma matriz quadrada A. Caso A não seja uma matriz hermitiana, pode-se utilizar a seguinte transformação:

$$\tilde{\mathrm{A}} = \begin{pmatrix} 0 & A \\ A^\dagger & 0 \end{pmatrix}. \tag{2.14}$$

Desta forma, os autovalores de A podem ser processados em uma rotina quântica, por exemplo, para multiplicar A ou A^{-1} com um vetor codificado em amplitude (SCHULD; PETRUCCIONE, 2018b).

2.2.2 O Algoritmo

O algoritmo de Grover consiste em três etapas principais: *preparação dos estados*, *oráculo* e *difusão*, conforme ilustrado na **Figura 13**.

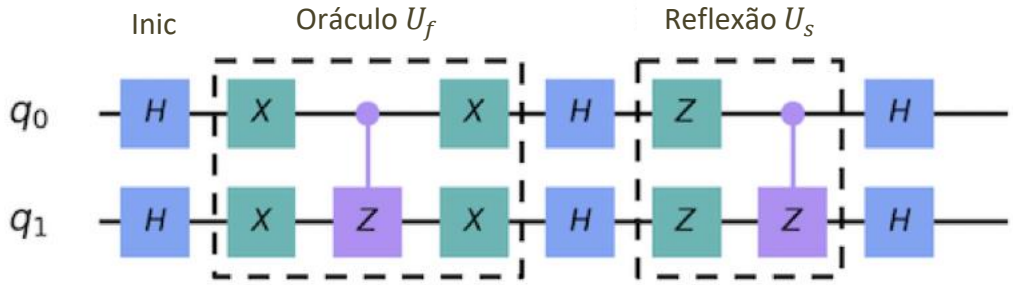

Figura 13. Ilustração do circuito do Algoritmo de Grover para 2 *qubits*.
Fonte: (YE, 2020)

A *preparação do estado* é onde o espaço de busca é criado, que são todos os casos possíveis de resposta. No exemplo de lista mencionada acima, o espaço de busca seria todos os itens dessa lista. O *oráculo* é o que marca a resposta correta, e o *operador de difusão* amplia essa resposta para que ela possa se destacar e ser medida no final do algoritmo. A resposta correta pode abranger mais de um item, e, nesse caso, todos eles são marcados e ampliados por meio desse algoritmo.

Esse procedimento é chamado de *amplificação de amplitude* e é a maneira como um computador quântico aumenta significativamente a probabilidade de medir a resposta marcada. Ao aumentar/amplificar a amplitude do item marcado, há uma diminuição da amplitude dos outros itens, de modo que a medição do estado final retornará o item correto com grande probabilidade.

O algoritmo de Grover tem uma interpretação geométrica interessante, pois produz duas reflexões que geram uma rotação no plano bidimensional. Os únicos dois estados que são necessários considerar é o vencedor $|\omega\rangle$ e a superposição uniforme $|s\rangle$. Esses dois vetores abrangem um plano bidimensional no espaço vetorial $\mathbb{C}^N$, como será mostrado a seguir.

2.2.2.1 Preparação do Estado

O procedimento de amplificação da amplitude inicia com a preparação do estado. A superposição uniforme $|s\rangle$ pode ser construída aplicando a porta Hadamard em todos os *qubits*, inicializados em $|0\rangle$:

$$|s\rangle = H^{\otimes n}|0\rangle^{n}.$$

Com isso, o estado do sistema $|s\rangle$ pode ser expresso por:

$$|s\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle. \tag{2.15}$$

Se fosse realizada uma medida na base padrão $|x\rangle$ essa superposição colapsaria em qualquer um dos estados da base com a mesma probabilidade de $\frac{1}{N} = \frac{1}{2^n}$.

A **Figura 14** mostra, à esquerda, o plano bidimensional gerado pelos vetores perpendiculares $|w\rangle$ e $|s'\rangle$ e que permite expressar o estado inicial $|s\rangle$ e, à direita, um gráfico de barras das amplitudes do estado $|s\rangle$.

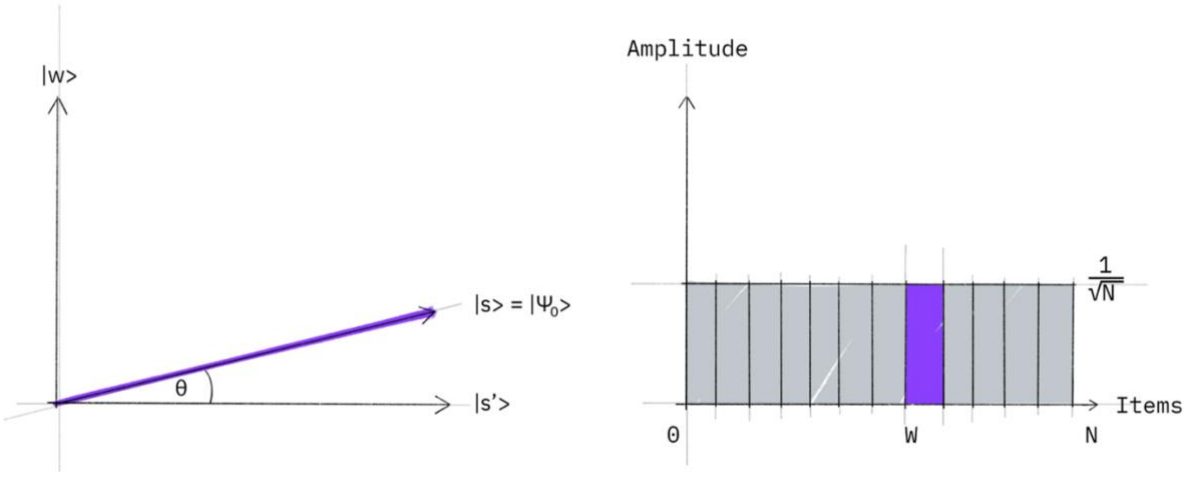

Figura 14. Ilustração do plano bidimensional à esquerda e gráfico de barras das amplitudes à direita: $|s\rangle$.
Fonte: (QISKIT, 2021b)

2.2.2.2 Oráculo

O próximo passo é aplicar o oráculo de reflexão U_f ao estado $|s\rangle$. Geometricamente, isso corresponde a uma reflexão do estado $|s\rangle$ sobre $|s'\rangle$. Esta transformação significa que a amplitude do estado se torna negativa, o que, por sua vez, implica que a amplitude média – indicada pela linha tracejada na imagem da direita da **Figura 15** – foi reduzida.

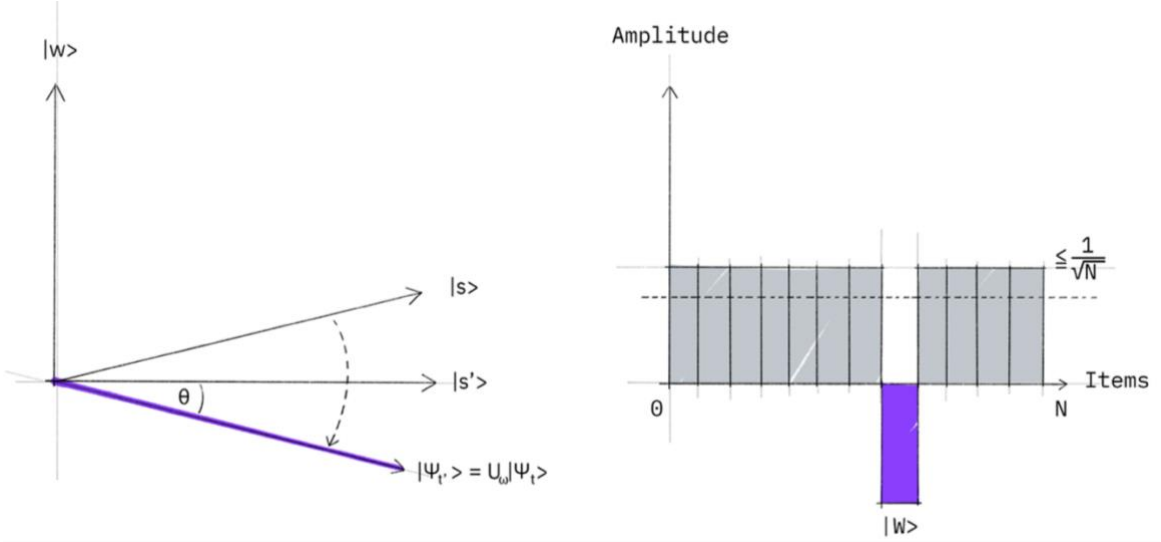

Figura 15. Ilustração do plano bidimensional à esquerda e gráfico de barras das amplitudes à direita: $U_f|s\rangle$.
Fonte: (QISKIT, 2021b)

Oráculos adicionam uma fase negativa aos estados da solução. Este oráculo será uma matriz diagonal, onde a entrada que corresponde ao item marcado terá uma fase negativa. Ou seja, para qualquer estado $|x\rangle$ na base computacional:

$$U_\omega|x\rangle = \begin{cases} |x\rangle & se\ x \neq \omega; \\ -|x\rangle & se\ x = \omega. \end{cases} \tag{2.16}$$

Ou seja, o operador oráculo é a porta CZ que pode ser representado por uma matriz $N \times N$, cujos elementos da diagonal principal são todos iguais a 1, com exceção do último elemento, que possui sinal negativo, para marcar o estado desejado:

$$U_f = \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & -1 \end{pmatrix}. \tag{2.17}$$

2.2.2.3 Difusor

Nessa etapa é aplicada uma reflexão adicional U_s ao estado $|s\rangle$, onde:

$$U_s = 2|s\rangle\langle s| - \mathbb{1}. \tag{2.18}$$

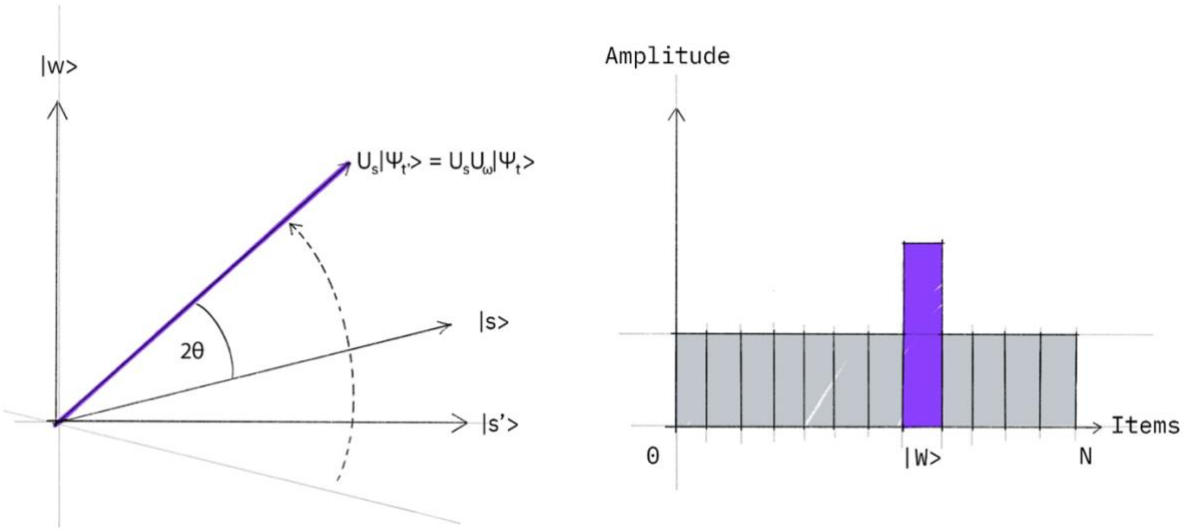

Figura 16. Ilustração do plano bidimensional à esquerda e gráfico de barras das amplitudes à direita: $U_s U_f|s\rangle$.
Fonte: (QISKIT, 2021b)

Essa transformação mapeia o estado $|s\rangle$ para $U_s U_f|s\rangle$ e completa a transformação.

2.2.2.4 Iterações

Duas reflexões resultam em uma rotação. Ou seja, a aplicação de $U_s U_f$ rotaciona o estado inicial $|s\rangle$ para mais perto do estado vencedor $|\omega\rangle$. Esse procedimento deve ser repetido t vezes e o estado $|\psi_t\rangle$ será obtido de forma que:

$$|\psi_t\rangle = \left(U_s U_f\right)^t |s\rangle. \tag{2.19}$$

De acordo com (SCHULD; PETRUCCIONE, 2018a), o número ideal de iterações necessárias é expresso pela equação:

$$t = \frac{\pi}{4}\sqrt{\frac{N}{m}}. \tag{2.20}$$

Onde $N = 2^n$ é o tamanho do espaço de busca (com $n = n^{\underline{o}}$ de *qubits*) e m é o número de respostas que se busca. A **Tabela 2.3** mostra a quantidade ideal de iterações para um espaço de busca construído a partir 2, 3, 4 e 5 *qubits* e com $m = 1$, ou seja, um único estado vencedor. Casos de arredondamento são ajustados para o menor número inteiro.

Tabela 2.3 Cálculo do número de iterações ideal para 2, 3, 4 e 5 *qubits* com um único estado vencedor.

Número de *qubits*	**Número de iterações *t***
2	$t = \frac{\pi}{4}\sqrt{2^2} = 1{,}57 \approx 1$
3	$t = \frac{\pi}{4}\sqrt{2^3} = 2{,}22 \approx 2$
4	$t = \frac{\pi}{4}\sqrt{2^4} = 3{,}14 \approx 3$
5	$t = \frac{\pi}{4}\sqrt{2^5} = 4{,}44 \approx 4$

2.3 ANÁLISE DE DESEMPENHO

Benchmarks para computadores convencionais são métodos padronizados que testam e avaliam *hardware*, *software* e sistemas para computação. Os resultados desses testes são expressos por meio de métricas que medem recursos e comportamentos do sistema, como velocidade e precisão. Para computadores quânticos, novos *benchmarks* são necessários para abordar essas mesmas métricas e, ao mesmo tempo, levar em conta as diferenças nas tecnologias subjacentes e nos modelos computacionais (IEEE QUANTUM, 2019).

Os *benchmarks de sistemas* medem as características fundamentais de um computador quântico. Essas medidas permitem avaliações do desempenho da máquina sem considerar casos de uso específicos e fornecem um registro claro do progresso. Já os *benchmarks de aplicações* fornecem uma visão mais abrangente do desempenho de um computador quântico em tarefas específicas. Eles podem ajudar os usuários finais e investidores a avaliar o desempenho de um sistema inteiro e medir o desempenho com base em casos de uso do mundo real, fornecendo valor significativo além de apenas avaliar a vantagem quântica (LANGIONE et al., 2022).

Atualmente, alguns testes de *benchmark* para *hardwares* quânticos já estão começando a ser desenvolvidos para testar a *qualidade* dos *qubits* (PIRES, 2021). Existem, inclusive, empresas especializadas nesse mercado[7]. De acordo com (LANGIONE et al., 2022):

> O uso efetivo de *benchmarks* pode ser uma importante fonte de vantagem competitiva para investidores e usuários finais.

Ainda de acordo com os autores:

> O *benchmarking* de desempenho – avaliando os recursos absolutos e relativos de diferentes plataformas ou sistemas – provou ser útil na avaliação de outras tecnologias profundas. Acreditamos que o *benchmarking* de desempenho pode acelerar o progresso na computação quântica. A chave é projetar *benchmarks* que sejam úteis (eles dizem aos usuários o que eles precisam saber), escaláveis (eles podem se expandir e se adaptar às tecnologias em evolução) e abrangentes (eles cobrem todos os atributos relevantes) – negócio complicado para uma empresa tão radical e complexa. Dito isso, várias organizações já estão avançando e provavelmente

[7] Como é o caso da QuantumBenchmark®: https://quantumbenchmark.com/. Acesso em: 2 jul. 2022.

> veremos mais recursos de *benchmarking* surgirem à medida que a tecnologia se aproximar do mercado.

Ou seja, um bom *hardware* quântico não é feito apenas da quantidade de *qubits* que o compõe. Também é importante que haja *estabilidade* no sistema, por exemplo. Pensando nisso, David P. DiVicenzo publicou o que ficou conhecido como *Critério de DiVicenzo* (DIVINCENZO, 2000), onde os requisitos para a implementação física de um computador quântico de qualidade são elencados:

1. Escalabilidade: deve ser um sistema físico escalável com suas partes bem caracterizadas, usualmente *qubits*.
2. Inicialização: ter a habilidade de inicializar o sistema em um estado de *qubits* simples e fidedigno.
3. Controle: ter a habilidade de controlar o estado do computador através do uso de portas lógicas elementares universais.
4. Estabilidade: possuir grandes tempos de decoerência e a habilidade de suprimir essa decoerência através de correção de erros e da aplicação de computação tolerante a falhas.
5. Mensuração: ter a habilidade de medir o estado do sistema em uma base conveniente.

De acordo com (GEORGOPOULOS; EMARY; ZULIANI, 2021), dentro da era *Noisy Intermediate-Scale Quantum* (NISQ)[8] (PRESKILL, 2018), o *benchmarking* das capacidades e desempenho de computadores quânticos ao executar algoritmos quânticos é de suma importância, especialmente para avaliar sua *escalabilidade*:

[8] Corresponde à era atual da computação quântica, com computadores de 50-100 *qubits* que são limitados pelo ruído em portas quânticas.

> Uma das maneiras de realizar *benchmarking* em computadores quânticos é estabelecendo um conjunto de algoritmos e medindo seus desempenhos ao executar cada um deles. Esse processo ganha mais importância à medida que computadores quânticos maiores (com mais *qubits*) são construídos. (...)
> As diferentes tecnologias quânticas concorrentes representam um desafio. Os *hardwares* possuem diferentes topologias e, portanto, têm pontos fortes e fracos únicos. Por exemplo: a conectividade de um processador de *armadilha de íons* fornece uma grande vantagem em alguns *benchmarks* sobre um processador de *qubits supercondutores* (LINKE et al., 2017). (...)
> Contudo, problemas inteiramente novos podem ser introduzidos ao expandir e é difícil dizer se os desempenhos medidos hoje são bons indicadores de desempenho futuro. Por exemplo, o processador da IonQ (WRIGHT et al., 2019) tem todos os 11 *qubits* totalmente conectados. Essa configuração é possível nessa escala, mas isso pode não ser verdade para um sistema com cem ou mil qubits.

Na **Figura 17** é possível ver como é a construção do processador de 11 *qubits* da IonQ, mencionada pelo autor. E a **Figura 18** ilustra a arquitetura de alguns dos processadores de da IBM para comparação.

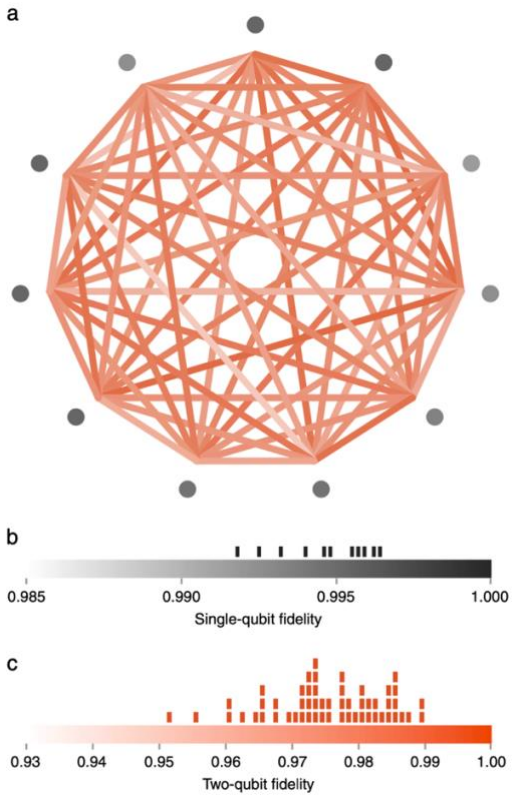

Figura 17. Ilustração da arquitetura de construção do processador da IonQ.
Fonte: (WRIGHT et al., 2019)

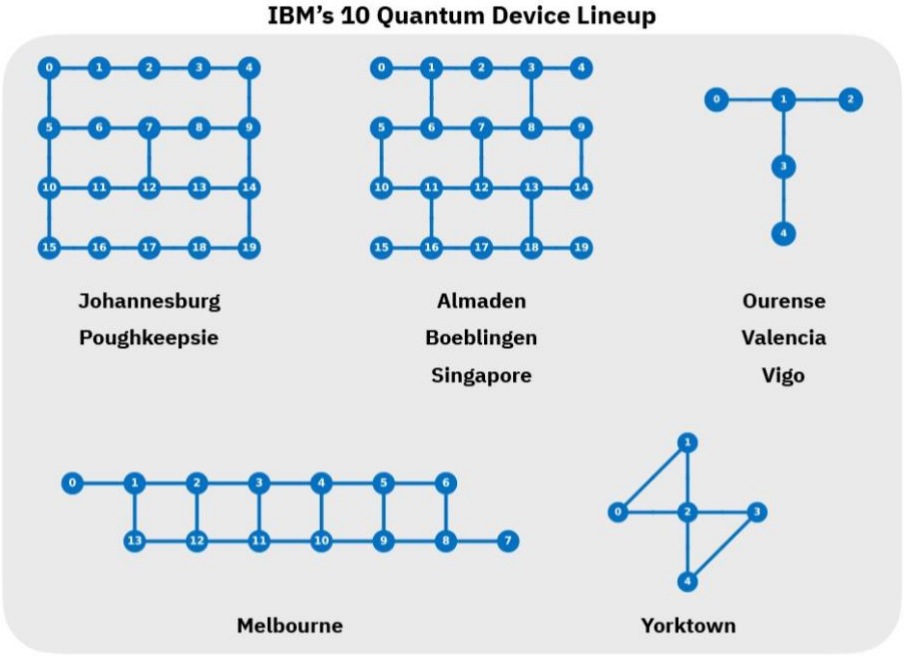

Figura 18. Ilustração da arquitetura de construção de alguns dos processadores da IBM.
Fonte: (IBM RESEARCH, 2019)

2.3.1 Algoritmos para *Benchmarking*

A escolha de algoritmos quânticos para *benchmarking* é crucial. Esses algoritmos devem ser escolhidos de acordo com três características principais que são interessantes para *benchmarking*:

1. Escalabilidade: o algoritmo deve ser capaz de aumentar de complexidade (e diminuir) para que seja possível executá-lo em sistemas quânticos cada vez maiores.
2. Previsibilidade: o algoritmo deve produzir um resultado que seja facilmente previsível. Uma adição importante à previsibilidade é a suscetibilidade ao ruído: o algoritmo deve fornecer um resultado cuja distorção sob os efeitos do ruído seja facilmente distinguível da evolução ideal.

3. Vantagem quântica: O algoritmo deve fornecer uma aceleração computacional sobre sua contraparte clássica ou, em outras palavras, representar uma aplicação do mundo real possivelmente relevante.

Segundo (GEORGOPOULOS; EMARY; ZULIANI, 2021), a *busca quântica* (ou QS, para *quantum search*) representa um algoritmo ideal para *benchmarking* de computadores quânticos pois:

1. O algoritmo pode escalar para procurar um item em um banco de dados maior simplesmente adicionando *qubits* ao registro quântico relevante.
2. O resultado é facilmente previsível, pois é simplesmente o item procurado, além de altamente suscetível a ruídos (por exemplo: um resultado errado pode aparecer devido ao ruído).
3. Além disso, a QS pode acelerar um problema de busca não estruturada de forma quadrática, tornando-se uma aplicação muito atraente para computadores quânticos.
4. E ainda, o algoritmo de Grover pode servir como sub-rotina para obter melhorias quadráticas de tempo de execução para uma variedade de outros algoritmos através da amplificação de amplitude (SCHULD; PETRUCCIONE, 2021).

Ainda segundo os autores, existem quatro características de um circuito quântico que são de interesse para *benchmarking*, os quais serão explorados nesse estudo:

i. O número de portas no circuito,
ii. O número de *qubits* ativos, ou *qubits* que são utilizados pelo circuito quântico (também chamado de espaço de trabalho),
iii. A profundidade do circuito, ou seja, o caminho mais longo entre o início do circuito e uma porta de medição,
iv. O tempo de execução do circuito no processador quântico.

3 METODOLOGIA

Para a implementação do algoritmo proposto serão utilizados cinco *hardwares* quânticos distintos, disponibilizados pela IBM, pela Microsoft e pela Amazon e os resultados obtidos serão comparados ao final do estudo. Também serão gerados resultados utilizando o Qiskit Aer, um simulador de alto desempenho para estudar algoritmos e aplicações de computação quântica (QISKIT, 2018).

Embora cada empresa possua linguagens/bibliotecas próprias para desenvolvimento de algoritmos quânticos, todas aceitam a implementação através de bibliotecas Qiskit – um SDK[9] de código aberto para trabalhar com computadores quânticos no nível de pulsos, circuitos e módulos de aplicativos em linguagem Python, e que será utilizado nesse projeto.

3.1 PLATAFORMAS E HARDWARES

3.1.1 IBM Quantum®

A plataforma IBM Quantum[10] disponibiliza diversas ferramentas para implementação de algoritmos quânticos, além de acesso aos *hardwares* quânticos da própria empresa. Para acessá-la basta criar uma conta gratuita, o IBMid. Dentro da plataforma, há a opção de utilizar o IBM Quantum Composer® para implementar circuitos quânticos de maneira gráfica ou o IBM Quantum Lab®, que fornece acesso a *Jupyter Notebooks* para a implementação de circuitos por meio de algoritmos.

Atualmente, a empresa oferece acesso gratuito a oito de seus processadores, seis deles com 5 *qubits* cada (*ibmq_lima*, *ibmq_belem*, *ibmq_quito*, *ibmq_manila*) e dois com 7 *qubits* (*ibm_nairobi*, *ibm_oslo*), além de cinco diferentes simuladores. O uso desses processadores é

[9] A sigla SDK significa *Software Development Kit*, que, em tradução livre, significa *Kit para Desenvolvimento de Software*.

[10] Disponível em: https://quantum-computing.ibm.com/. Acesso em: 24 jun. 2022.

totalmente gratuito, sem cota ou tempo máximo de uso. A única restrição é com relação ao número de *shots*[11] por tarefa, sendo 20.000 o máximo permitido. Existem processadores com maior número de *qubits*, porém de acesso pago. Além disso, todos os processadores da empresa são construídos utilizando a tecnologia de *qubits* supercondutores.

O *hardware* utilizado nesse estudo foi o *ibmq_belem*, devido ao menor tempo de espera em fila na data de execução do algoritmo. A **Figura 19** mostra sua arquitetura de construção.

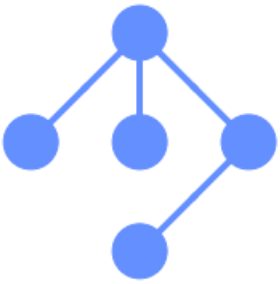

Figura 19. Ilustração da arquitetura de construção do processador *ibmq_belem* da IBM.

3.1.1.1 IBM Quantum Composer®

Pode-se facilmente construir circuitos quânticos graficamente com o IBM Quantum Composer. Ele permite que os usuários vejam dinamicamente algumas propriedades úteis do circuito, como as probabilidades de resultado de medição, vetores de estado, fase e *Q-sphere* (uma generalização da esfera de Bloch para sistemas com 2 ou mais *qubits*). Além disso, fornece a implementação do circuito em código Python.

3.1.1.2 IBM Quantum Lab®

O IBM Quantum Lab, fornece uma coleção de tutoriais em *Jupyter Notebooks* criados pela equipe do Qiskit®. Além disso, pode-se criar e rodar seu próprio *notebook* utilizando bibliotecas Qiskit. Devido a maior versatilidade desse ambiente, para a implementação do algoritmo no *hardware* quântico da empresa será utilizada essa opção.

---

[11] Um shot é uma única execução de um algoritmo em um processador quântico.

3.1.2 Microsoft Azure Quantum®

A plataforma de desenvolvimento de circuitos quânticos na nuvem da Microsoft é a Azure Quantum®. Nela é possível utilizar os *hardwares* quânticos disponibilizados pela empresa – atualmente estão disponíveis processadores da IonQ, Quantinuum e Rigetti – através de *Jupyter Notebooks*. A empresa possibilita a utilização de bibliotecas Qiskit em Python, embora tenha desenvolvido a linguagem Q# – criada pela empresa especificamente para se trabalhar com algoritmos quânticos.

Embora haja custos para se executar circuitos nos processadores disponibilizados pela Microsoft, ao criar uma conta na plataforma Azure, a empresa fornece um crédito de U$500 para cada um dos processadores quânticos.

3.1.2.1 IonQ

O IonQ Harmony é um computador quântico de íons presos (também chamado de *armadilha de íons*) e é configurável dinamicamente para usar até 11 *qubits*. Todos os *qubits* estão totalmente conectados conforme ilustrado na **Figura 17**, o que significa que é possível executar uma porta de dois *qubits* entre qualquer par (AZURE QUANTUM, 2022).

Também é disponibilizado IonQ Aria, a última geração de computador quântico de íons presos da empresa. Com um sistema também configurável dinamicamente, porém contando com 23 *qubits*. Contudo, devido a um *bug* de integração da plataforma Azure ao sistema da IonQ, até o fim desse estudo não houve sucesso ao executar o circuito nesse processador.

3.1.2.2 Quantinuum

A Quantinuum fornece acesso a sistemas de íons presos com alta fidelidade e *qubits* totalmente conectados. A geração de modelo de sistema H1 de computadores quânticos da empresa inclui dois computadores de destino: H1-1 e H1-2. Ambos os computadores quânticos

têm, fundamentalmente, o mesmo design e ambos atendem a um conjunto nominal de requisitos técnicos (QUANTINUUM, 2022). Enquanto o H1-1 possui 20 *qubits*, o H1-2 possui 12 *qubits* (AZURE QUANTUM, 2022b). Na data de execução dos circuitos desse estudo, apenas o H1-2 estava disponível para utilização.

Como é possível ver na ilustração da **Figura 20** sua arquitetura de construção segue o modelo linear.

Figura 20. Ilustração da arquitetura de construção do processador H1 da Quantinuum.

3.1.2.3 Rigetti

Embora a Microsoft disponibilize esse processador na sua gama de computadores quânticos, a sua integração à plataforma é recente. Na data de execução desse projeto ele não estava em funcionamento.

3.1.3 Amazon Braket®

Com inspiração na nomenclatura da notação de Dirac, essa plataforma de serviço é a que dá acesso ao maior número de processadores quânticos de diferentes empresas: estão disponíveis *hardwares* da D-Wave, IonQ, OQC (*Oxford Quantum Circuits*), Rigetti e Xanadu. Entretanto, não há nenhuma modalidade de acesso gratuito aos computadores para o público geral, sendo o modo *Pay As You Go* a escolha de oferta da empresa. A **Tabela 3.1** mostra os valores praticados pela Amazon para cada processador disponível, na época da execução desse estudo.

Embora seja a empresa com o maior leque de opções de *hardwares* quânticos de diferentes empresas, o serviço ainda não está totalmente integrado ao Qiskit SDK, sendo necessário a implementação do algoritmo através do *Amazon Braket Python SDK*® (AMAZON, 2022a) para a utilização da maioria dos processadores. Contudo, em junho de 2022 a empresa anunciou a integração de parte dos *hardwares* ao Qiskit, sendo eles: Rigetti, OQC e IonQ (AMAZON, 2022b). Devido ao algoritmo do projeto estar utilizando o Qiskit em todas as outras implementações, preferiu-se seguir com o mesmo SDK.

Além disso, como o processador da IonQ está disponível pela Microsoft Azure® – que oferece crédito para sua utilização – este *hardware* não foi utilizado pela plataforma da Amazon.

Tabela 3.1 Valores praticados pela Amazon para cada *hardware* oferecido.

Hardware Provider	QPU family	Per-task price	Per-shot price
D-Wave	2000Q	$0.30000	$0.00019
D-Wave	Advantage	$0.30000	$0.00019
IonQ	IonQ device	$0.30000	$0.01000
OQC	Lucy	$0.30000	$0.00035
Rigetti	Aspen-11	$0.30000	$0.00035
Rigetti	Aspen-M	$0.30000	$0.00035
Xanadu	Borealis	$0.30000	$0.0002

Fonte: (AMAZON BRAKET PRICING, 2022)

3.1.3.1 Rigetti

O Rigetti Aspen-M-2 é um processador de 80 *qubits* baseado em tecnologia *multi-chip* escalável e apresenta recursos de leitura aprimorados que contribuem para melhores fidelidades gerais do circuito, independentemente da profundidade e largura. A topologia do *chip* Aspen é octogonal com conectividade 3 (2 para bordas) e apresenta portas de emaranhamento *CPHASE* e *XY* que permitem aos desenvolvedores otimizar programas para desempenho e minimizar a profundidade do circuito (RIGETTI, 2022), conforme ilustrado na **Figura 21**.

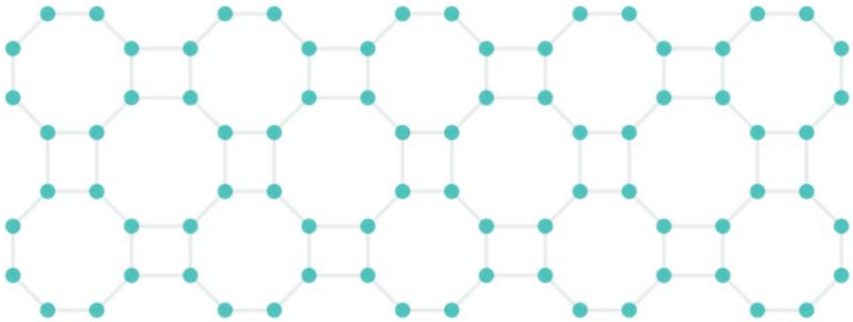

Figura 21. Ilustração da arquitetura de construção do processador Aspen-M da Rigetti.
Fonte: (RIGETTI, 2022)

3.1.3.2 OQC

A *Oxford Quantum Circuits* (OQC) é líder em computação quântica no Reino Unido e na Europa. Seu sistema mais recente, Lucy, é um processador quântico de 8 *qubits* supercondutores, nomeado em homenagem a Lucy Mensing, uma física alemã pioneira da mecânica quântica (AMAZON WEB SERVICES, 2022).

A principal inovação tecnológica dos processadores da empresa, o *Coaxmon*[12], é um tipo de *qubit* que possui arquitetura tridimensional e traz os principais componentes fora do chip para maior simplicidade, flexibilidade, facilidade de engenharia e – crucialmente – escalabilidade (OXFORD QUANTUM CIRCUITS, 2022).

[12] Um breve vídeo ilustrando essa tecnologia pode ser visto em: https://oxfordquantumcircuits.com/wp-content/uploads/2021/07/OQC-arch-vid-1.mp4. Acesso em: 25 set. 2022.

3.2 ALGORITMO

Devido às limitações atuais dos *hardwares* quânticos, a maior quantidade de *qubits* disponíveis de forma gratuita, comum há todos os processadores testados, é cinco unidades. Sendo assim, foram executados circuitos com 2, 3, 4 e 5 *qubits*. Em todos os casos o estado vencedor era único e correspondente a todos os *bits* iguais a 1 – a fim de facilitar a visualização gráfica – representados, respectivamente, por: $|11\rangle, |111\rangle, |1111\rangle, |11111\rangle$, utilizando a codificação de base. Dois tipos de circuitos foram construídos: com uma única aplicação da rotina de busca e com a quantidade ideal de iterações t, obtida pela equação (2.20). Além disso, foram executados três grupos de tarefas[13]: com 500 *shots*, com 1000 *shots* e com 2000 *shots*.

3.2.1 Probabilidade de Sucesso do Algoritmo

Uma fórmula para determinar a probabilidade de sucesso do algoritmo de Grover a partir de um estado de superposição inicial é fornecida por (BOYER et al., 1996). De acordo com os autores, a probabilidade de sucesso do algoritmo (ASP – *Algorithmic Success Probability*) de medir um único estado alvo ω (e os demais estados s') a partir do estado $|\psi_m\rangle$, após qualquer número de iterações $t = m + 1$ é dada por:

$$|\psi_m\rangle = |\psi(\omega_m, {s'}_m)\rangle,$$

onde:

$$ASP(\omega_{m+1}) = \left(\frac{N-2}{N}\omega_m + \frac{2(N-1)}{N}{s'}_m\right)^2; \tag{3.1}$$

[13] Uma tarefa é uma sequência de *shots* repetidos com base no mesmo projeto de circuito.

$$ASP(s'_{m+1}) = \left(\frac{N-2}{N} s'_m - \frac{2}{N}\omega_m\right)^2, \quad (3.2)$$

onde $N = 2^n$, com $n = n^{\underline{o}}$ de *qubits*, e $\omega_0 = s'_0 = \frac{1}{\sqrt{N}}$.

A **Tabela 3.2** apresenta as probabilidades de medida e amplitude teóricas do estado vencedor do algoritmo de busca de Grover para cada número de *qubits* executados. Como é possível ver, para 2 *qubits* basta uma iteração para atingir 100% de probabilidade de se medir o estado escolhido. Já para 3 *qubits*, após 2 iterações há 94,53% de probabilidade; para 4 *qubits*, são necessárias 3 iterações para atingir 96,13%; finalmente, para 5 *qubits*, é atingido 99,92% de probabilidade de se medir o estado escolhido após 4 iterações.

Tabela 3.2 Valores teóricos do algoritmo de Grover para diferentes quantidades de *qubits* e iterações, calculados a partir das Equações (3.1) e (3.2).

2 qubits	
n	2
N	4
Amplitude inicial	
ω_0	0,5000
s'_0	0,5000
Probabilidade após 1ª iteração	
m	0
ASP(ω_1)	1,0000
ASP(s'_1)	0,0000
Amplitude após 1ª iteração	
ω_1	1,0000
s'_1	0,0000
Probabilidade após 2ª iteração	
m	1
ASP(ω_2)	0,2500
ASP(s'_2)	0,2500
Amplitude Após 2ª Iteração	
ω_2	0,5000
s'_2	0,5000
Probabilidade após 3ª iteração	
m	2
ASP(ω_3)	1,0000
ASP(s'_3)	0,0000
Amplitude após 3ª iteração	
ω_3	1,0000
s'_3	0,0000
Probabilidade após 4ª iteração	
m	3
ASP(ω_4)	0,2500
ASP(s'_4)	0,2500
Amplitude após 4ª iteração	
ω_4	0,5000
s'_4	0,5000

3 qubits	
n	3
N	8
Amplitude inicial	
ω_0	0,3536
s'_0	0,3536
Probabilidade após 1ª iteração	
m	0
ASP(ω_1)	0,7813
ASP(s'_1)	0,0313
Amplitude após 1ª iteração	
ω_1	0,8839
s'_1	0,1768
Probabilidade após 2ª iteração	
m	1
ASP(ω_2)	0,9453
ASP(s'_2)	0,0078
Amplitude Após 2ª Iteração	
ω_2	0,9723
s'_2	0,0884
Probabilidade após 3ª iteração	
m	2
ASP(ω_3)	0,7812
ASP(s'_3)	0,0313
Amplitude após 3ª iteração	
ω_3	0,8839
s'_3	0,1768
Probabilidade após 4ª iteração	
m	3
ASP(ω_4)	0,9453
ASP(s'_4)	0,0078
Amplitude após 4ª iteração	
ω_4	0,9723
s'_4	0,0884

4 qubits	
n	4
N	16
Amplitude inicial	
ω_0	0,2500
s'_0	0,2500
Probabilidade após 1ª iteração	
m	0
ASP(ω_1)	0,4727
ASP(s'_1)	0,0352
Amplitude após 1ª iteração	
ω_1	0,6875
s'_1	0,1875
Probabilidade após 2ª iteração	
m	1
ASP(ω_2)	0,9084
ASP(s'_2)	0,0061
Amplitude Após 2ª Iteração	
ω_2	0,9531
s'_2	0,0781
Probabilidade após 3ª iteração	
m	2
ASP(ω_3)	0,9613
ASP(s'_3)	0,0026
Amplitude após 3ª iteração	
ω_3	0,9805
s'_3	0,0508
Probabilidade após 4ª iteração	
m	3
ASP(ω_4)	0,9084
ASP(s'_4)	0,0061
Amplitude após 4ª iteração	
ω_4	0,9531
s'_4	0,0781

5 qubits	
n	5
N	32
Amplitude inicial	
ω_0	0,1768
s'_0	0,1768
Probabilidade após 1ª iteração	
m	0
ASP(ω_1)	0,2583
ASP(s'_1)	0,0239
Amplitude após 1ª iteração	
ω_1	0,5082
s'_1	0,1547
Probabilidade após 2ª iteração	
m	1
ASP(ω_2)	0,6024
ASP(s'_2)	0,0128
Amplitude Após 2ª Iteração	
ω_2	0,7762
s'_2	0,1132
Probabilidade após 3ª iteração	
m	2
ASP(ω_3)	0,8969
ASP(s'_3)	0,0033
Amplitude após 3ª iteração	
ω_3	0,9471
s'_3	0,0577
Probabilidade após 4ª iteração	
m	3
ASP(ω_4)	0,9992
ASP(s'_4)	0,0000
Amplitude após 4ª iteração	
ω_4	0,9996
s'_4	0,0051

3.2.2 Preparação dos Circuitos

Os circuitos foram preparados conforme as figuras a seguir (**Figura 22** a **Figura 28**). O operador oráculo foi implementado conforme (2.17) e o difusor segue a fórmula (2.18).

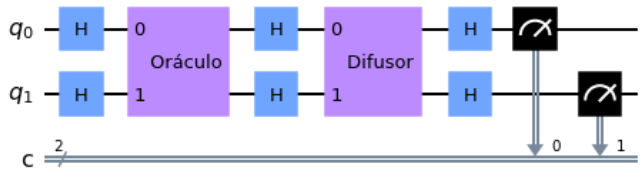

Figura 22. Ilustração do circuito para 2 *qubits* com uma iteração – o número ideal de iterações.

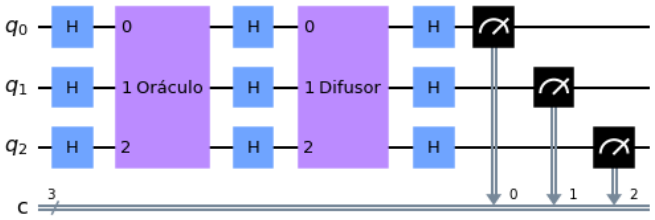

Figura 23. Ilustração do circuito com 3 *qubits* com uma única iteração.

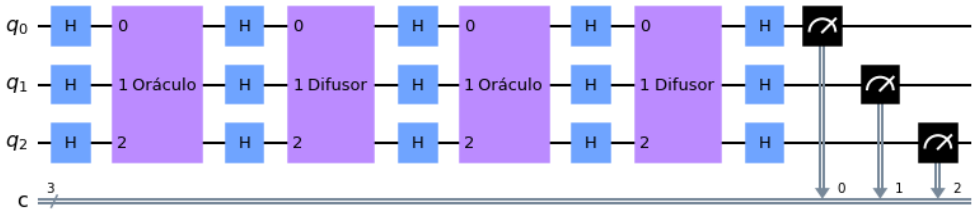

Figura 24. Ilustração do circuito com 3 *qubits* com o número ideal iterações.

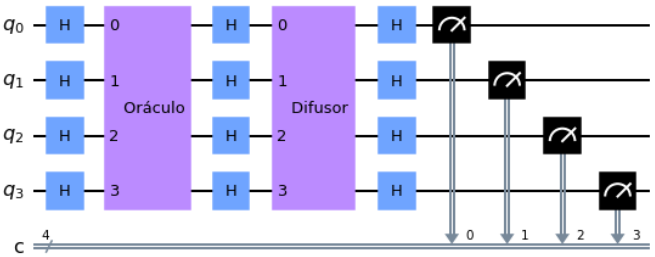

Figura 25. Ilustração do circuito com 4 *qubits* com uma única iteração.

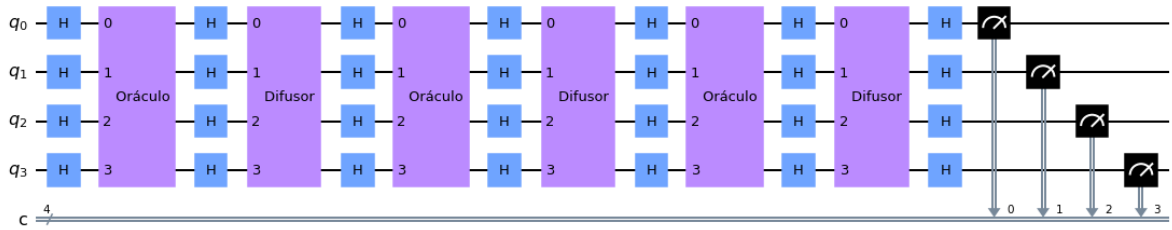

Figura 26. Ilustração do circuito com 4 *qubits* com o número ideal de iterações.

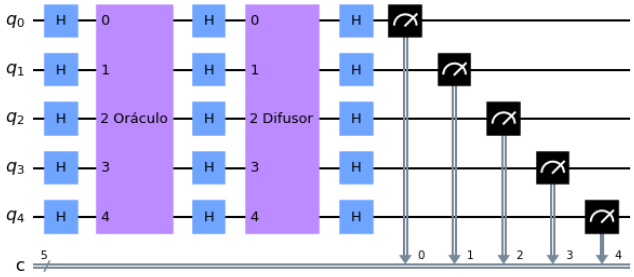

Figura 27. Ilustração do circuito com 5 *qubits* com uma única iteração.

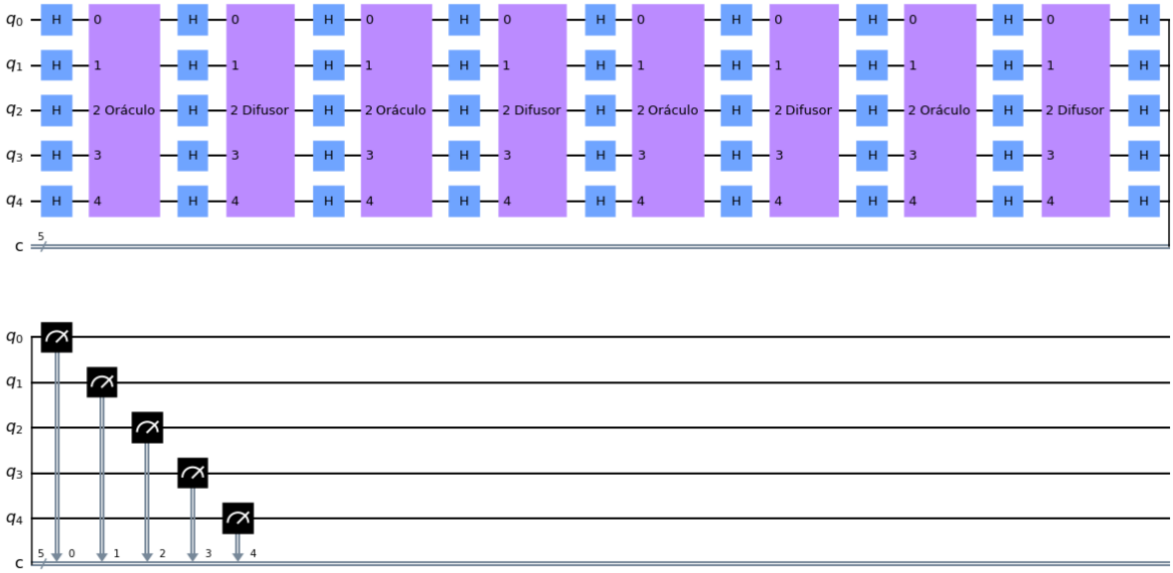

Figura 28. Ilustração do circuito com 5 *qubits* com o número ideal de iterações.

3.2.3 Características dos Circuitos

É importante salientar que ao executar um algoritmo em um *hardware* quântico, o aumento da quantidade de *qubits* causa um aumento na *quantidade de portas* necessárias para a implementação do circuito e na sua *profundidade* – o que influencia diretamente no ruído contido na resposta final.

3.2.3.1 Profundidade

A profundidade de um circuito é uma métrica que calcula o caminho mais longo entre a entrada de dados e a saída (GUNZI, 2020). Cada porta conta como uma unidade. Um ponto importante a ser considerado é que se um *qubit* depende de outro, um deles tem que esperar que o outro seja computado, por isso as portas aplicadas ao primeiro *qubit* contam para seu dependente. Assim, a profundidade do circuito de exemplo da **Figura 29** é 6.

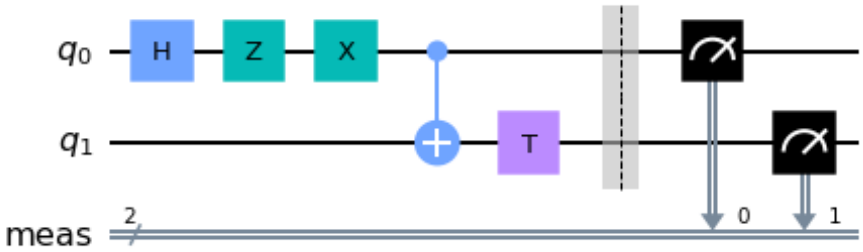

Figura 29. Ilustração de um circuito de profundidade 6.

3.2.4 Transpilação

A *transpilação* é o processo de reescrever um determinado circuito de entrada para corresponder à topologia de um dispositivo quântico específico e otimizar o circuito para execução nos atuais sistemas quânticos ruidosos. A maioria dos circuitos deve passar por uma série de transformações que os tornam compatíveis com um determinado dispositivo alvo e os otimizam para reduzir os efeitos do ruído nos resultados.

Reescrever circuitos quânticos para corresponder às restrições de *hardware* e otimizar o desempenho pode estar longe de ser trivial (QISKIT 0.38.0, 2022). Contudo, o Qiskit possui métodos de transpilação pré-construídos disponíveis, que foram utilizados nesse projeto.

Após a transpilação o número de portas cresce consideravelmente, a depender das portas nativas de cada processador, da quantidade de *qubits* e da arquitetura do *hardware*. A **Tabela 3.3** apresenta essas informações para cada circuito implementado.

A **Figura 30** ilustra o circuito com 2 *qubits*, transpilado para execução no *hardware* da IBM e, na **Figura 31**, o mesmo circuito transpilado para execução no processador da IonQ. Enquanto no da IBM são necessárias 19 portas e uma profundidade de circuito igual a 12, no processador da IonQ, devido aos seus *qubits* totalmente interligados, o circuito transpilado fica com 10 portas e profundidade igual a 6.

Na **Figura 32** é possível ver o efeito da adição de um único *qubit* no espaço de busca: a quantidade de portas necessárias para implementação no *hardware* da IBM cresce de maneira exponencial para cada *qubit* adicionado ao espaço de busca do algoritmo. Isso não ocorre apenas com o processador da IBM, como é possível ver no gráfico da **Figura 33**[14] – porém, o *hardware* da empresa é o que apresenta a maior taxa de crescimento de portas por número de *qubits*. Na **Figura 34** é possível ver que a profundidade segue a mesma tendência. Além disso, pode-se notar que os processadores da IonQ e da Quantinuum necessitam de bem menos portas para suas implementações. Isso se deve à arquitetura de construção desses *hardwares*, como discutido anteriormente.

[14] Para facilitar a identificação, todos os gráficos desse projeto seguirão o seguinte esquema de cores: roxo para processador da IBM, tons azuis para processadores utilizados através da Microsoft Azure e tons alaranjados para os utilizados através da Amazon.

Tabela 3.3 Principais características dos circuitos implementados.

Nº de *qubits*	Nº de iterações	*Hardware*	Nº de portas	Profundidade
2	1	Belem (IBM)	19	12
		Harmony (IonQ)	10	6
		H1-2 (Quantinuum)	10	6
		Aspen-M2 (Rigetti)	10	6
		Lucy (OQC)	17	10
3	1	Belem (IBM)	153	109
		Harmony (IonQ)	48	32
		H1-2 (Quantinuum)	47	32
		Aspen-M2 (Rigetti)	82	57
		Lucy (OQC)	111	74
3	2	Belem (IBM)	316	210
		Harmony (IonQ)	82	56
		H1-2 (Quantinuum)	88	61
		Aspen-M2 (Rigetti)	196	140
		Lucy (OQC)	208	141
4	1	Belem (IBM)	725	503
		Harmony (IonQ)	83	57
		H1-2 (Quantinuum)	82	55
		Aspen-M2 (Rigetti)	295	210
		Lucy (OQC)	431	296
4	3	Belem (IBM)	2157	1576
		Harmony (IonQ)	233	165
		H1-2 (Quantinuum)	230	159
		Aspen-M2 (Rigetti)	1252	929
		Lucy (OQC)	868	624
5	1	Belem (IBM)	2962	2090
		Harmony (IonQ)	154	114
		H1-2 (Quantinuum)	151	114
		Aspen-M2 (Rigetti)	1258	892
		Lucy (OQC)	1754	1214

5	4	Belem (IBM)	11977	8470
		Harmony (IonQ)	586	450
		H1-2 (Quantinuum)	574	447
		Aspen-M2 (Rigetti)	4996	3559
		Lucy (OQC)	5028	3567

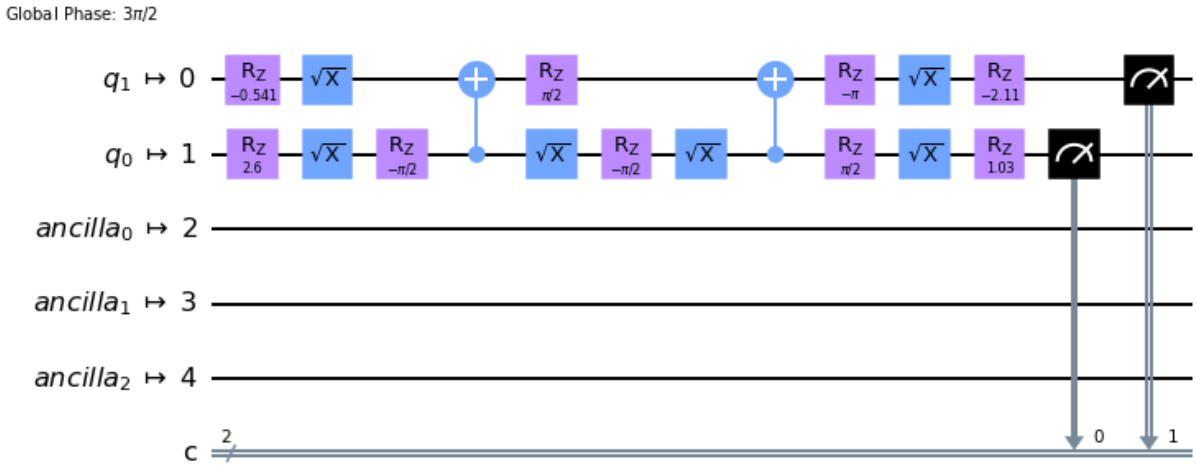

Figura 30. Ilustração do circuito de 2 *qubits* transpilado para execução no *hardware* Belem da IBM[15].

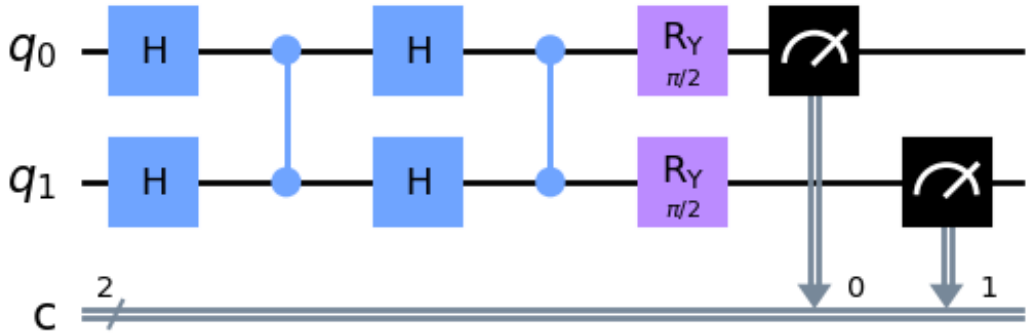

Figura 31. Ilustração do circuito de 2 *qubits* transpilado para execução no *hardware* Harmony da IonQ.

[15] Devido a esse *hardware* não ser configurável dinamicamente, todos os seus *quibits* são "chamados" para as tarefas. Os que não são utilizados no circuito são alocados como *ancilla* (*auxiliar*, em tradução livre). Contudo, é possível ver que eles não estão sendo utilizados no circuito – não há nenhuma porta aplicada a eles.

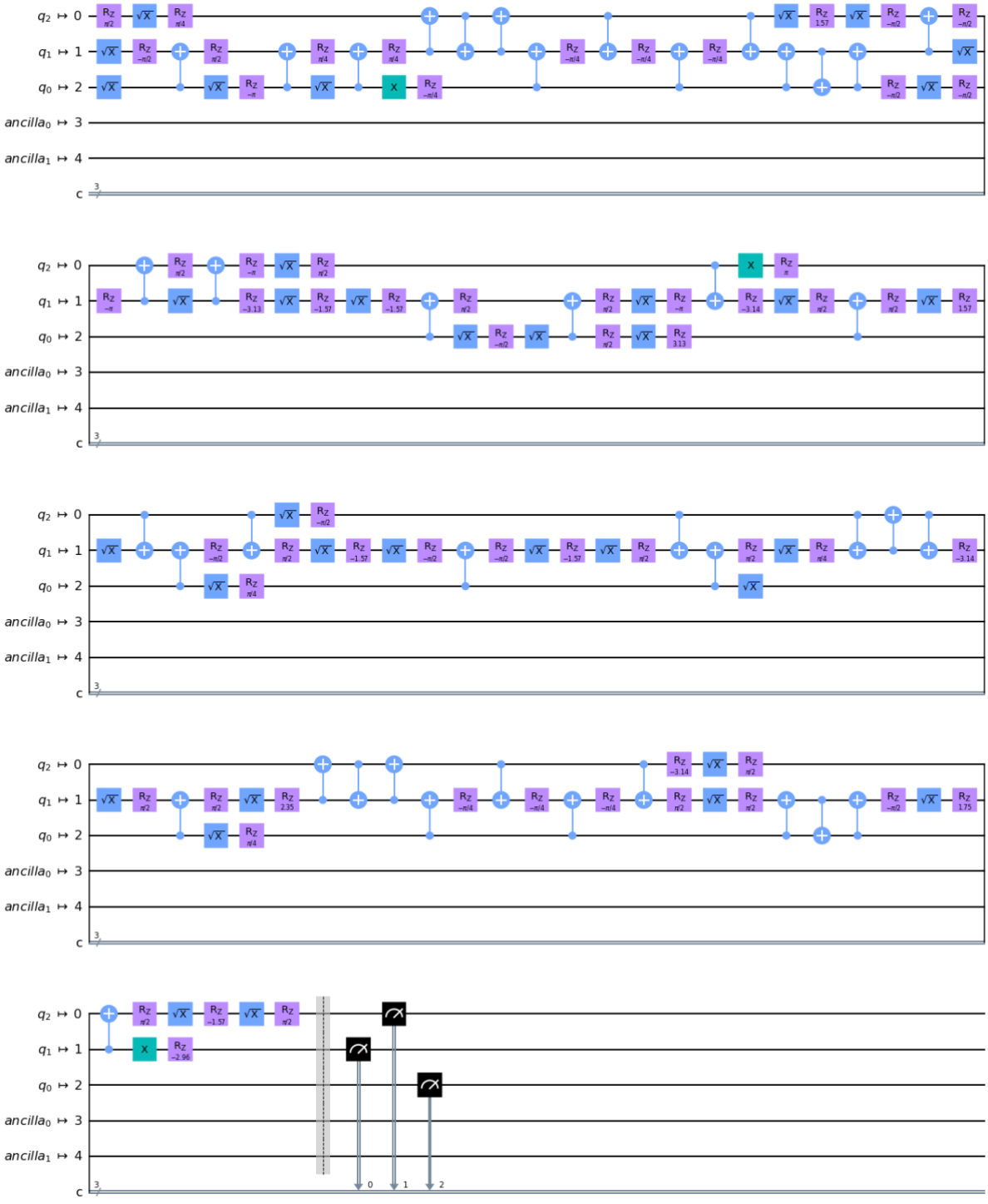

Figura 32. Ilustração do circuito de 3 *qubits* com *t* ideal para execução no *hardware* Belem da IBM.

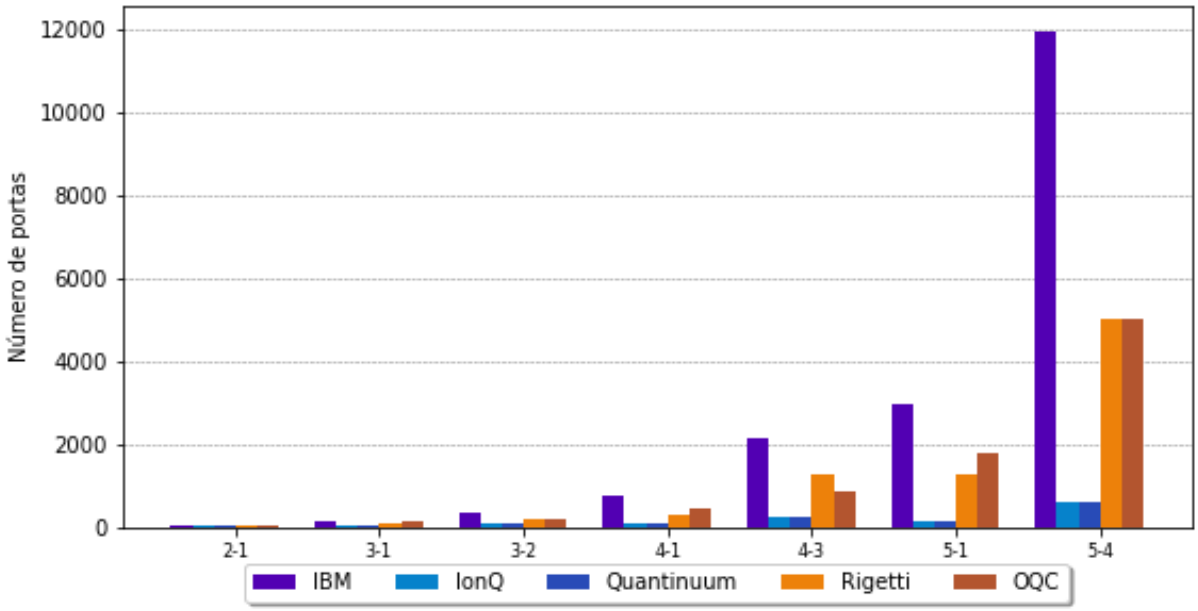

Figura 33. Gráfico comparativo do número de portas necessárias para implementação em cada processador.

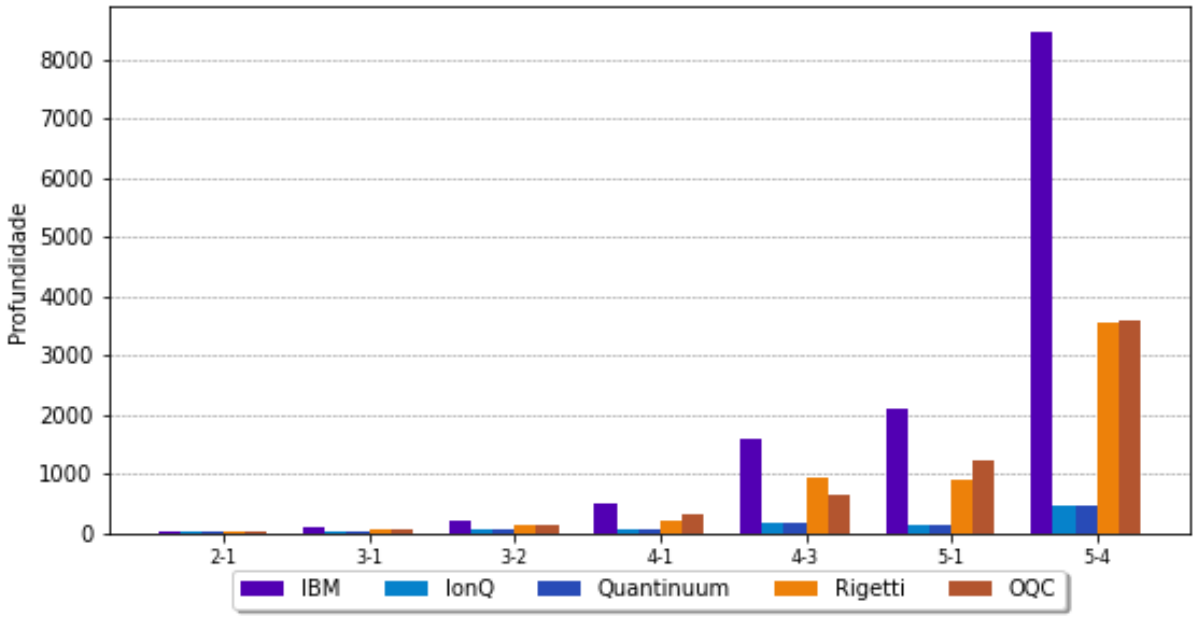

Figura 34. Gráfico comparativo da profundidade do circuito quântico em cada processador.

4 RESULTADOS

Os resultados foram separados conforme a quantidade de *shots* executados.

4.1 RESULTADOS OBTIDOS COM 500 *SHOTS*

Para 500 *shots* houve sucesso em rodar o algoritmo para os *hardwares* da IBM, IonQ e Quantinuum. Devido ao alto custo do processador da Quantinuum, não foi possível executar o circuito com 4 *qubits* – para essa configuração foi utilizado o emulador da empresa que usa um modelo físico realista e um modelo de ruído do H1-2 (AZURE QUANTUM, 2022b).

Na **Tabela 4.1** estão os resultados de desempenho para os processadores testados, onde *ASP Teórico* é o valor calculado analiticamente, conforme **Tabela 3.2**, *ASP Efetivo* é o valor adquirido através da execução do circuito nos *hardwares*, *Tempo de Execução* é o tempo requerido para executar o algoritmo no processador quântico e *Tempo Total* é o tempo decorrido desde o envio da tarefa até o recebimento dos resultados (*Tempo de Execução* + Tempo em Fila).

Tabela 4.1 Informações de desempenho dos processadores para 500 *shots*.

Nº de *qubits*	**Nº de iterações**	**ASP Teórico (%)**	***Hardware***	**ASP Efetivo (%)**	**Tempo de Execução**	**Tempo Total**
2	*1*	100	Simulador Aer	100	-	-
			Belem (IBM)	89.8	00:00:02	03:31:00
			Harmony (IonQ)	96.0	00:00:04	00:15:45
			H1-2 (Quantinuum)	98.2	00:01:27	11:07:00
3	*1*	78.13	Simulador Aer	78.0	-	-
			Belem (IBM)	23.2	00:00:03	03:31:00
			Harmony (IonQ)	53.6	00:00:05	00:17:06
			H1-2 (Quantinuum)	74.2	00:01:48	11:23:16

3	*2*	94.53	Simulador Aer	94.2	-	-
			Belem (IBM)	23.2	00:00:03	03:31:00
			Harmony (IonQ)	44.2	00:00:06	00:18:57
			H1-2 (Quantinuum)	73.8	00:02:27	11:07:42
4	*1*	47,27	Simulador Aer	48.6	-	-
			Belem (IBM)	4.8	00:00:03	03:31:00
			Harmony (IonQ)	14.4	00:00:07	00:19:20
			H1-2 sim (Quantinuum)	41.2	-	-
4	*3*	96,13	Simulador Aer	97.4	-	-
			Belem (IBM)	5.2	00:00:05	23:19:56
			Harmony (IonQ)	7.0	00:00:14	00:23:45
			H1-2 sim (Quantinuum)	69.4	-	-
5	*1*	25,83	Simulador Aer	23.8	-	-
			Belem (IBM)	1.6	00:00:04	23:19:56
			Harmony (IonQ)	3.2	00:00:11	00:24:49
			H1-2 (Quantinuum)	20	00:33:53	20:09:25
5	*4*	99,92	Simulador Aer	100	-	-
			Belem (IBM)	1.2	00:00:08	02:11:56
			Harmony (IonQ)	2.6	00:00:32	00:26:43
			H1-2 (Quantinuum)	17	00:37:01	19:29:36

Na **Figura 35** está o gráfico comparativo do valor de ASP para cada processador em cada um dos circuitos implementados. Já na **Figura 36** estão os tempos de execução que cada processador levou para executar cada um dos circuitos. O custo total para implementação em cada *hardware* está descrito na **Tabela 4.2**.

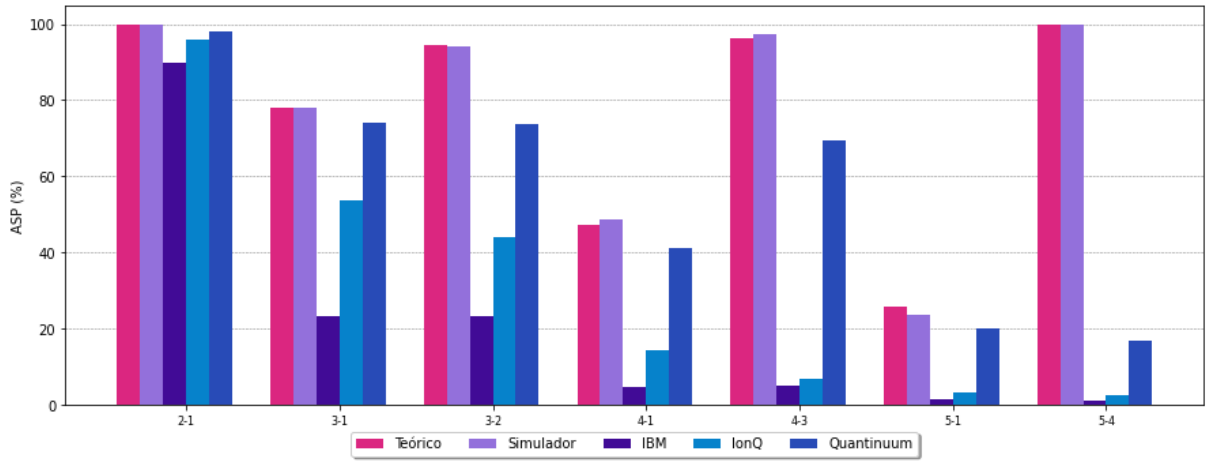

Figura 35. Gráfico do *ASP* obtido em cada processador para cada um dos circuitos.

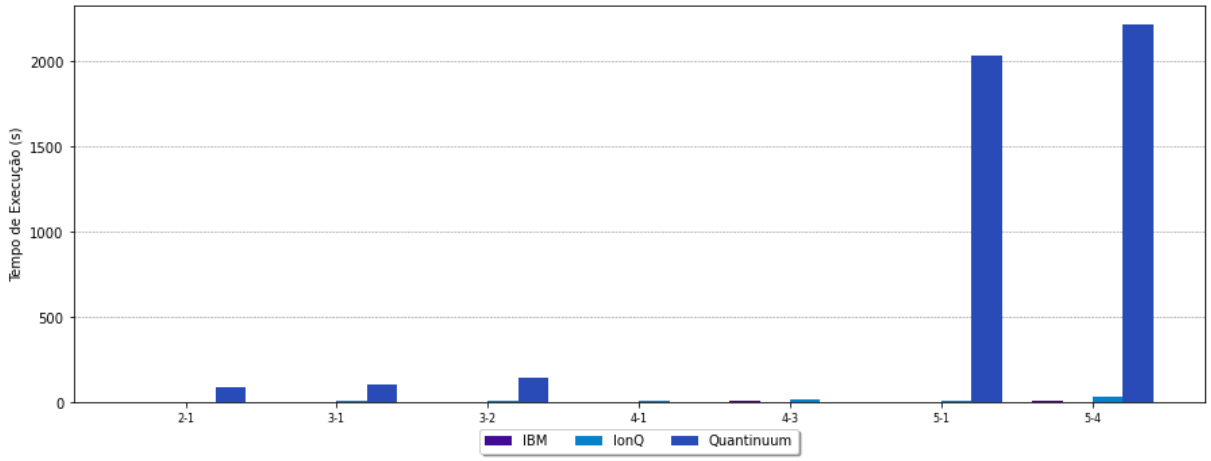

Figura 36. Tempo de execução requerido por cada processador para cada um dos circuitos.

Tabela 4.2 Custo total para execução dos circuitos.

Hardware	**Custo**
Belem (IBM)	U$ 0.00
Harmony (IonQ)	U$ 83.92
H1-2 (Quantinuum)	U$ 600.20

4.1.1 Histogramas

A seguir estão os histogramas gerados para cada um dos circuitos implementados com 500 *shots*.

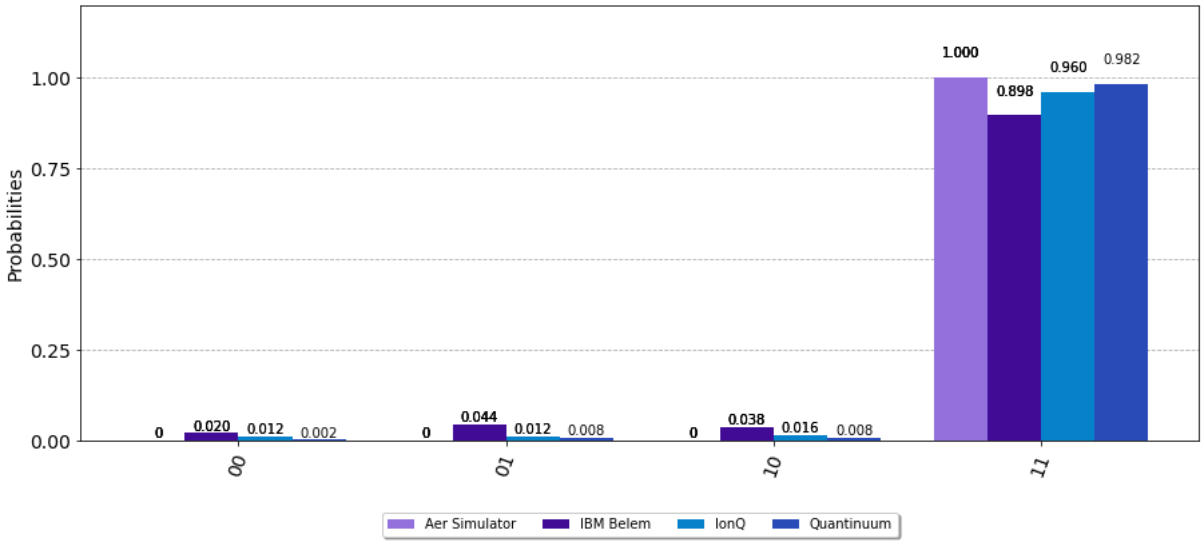

Figura 37. Histograma com resultados para 2 *qubits*.

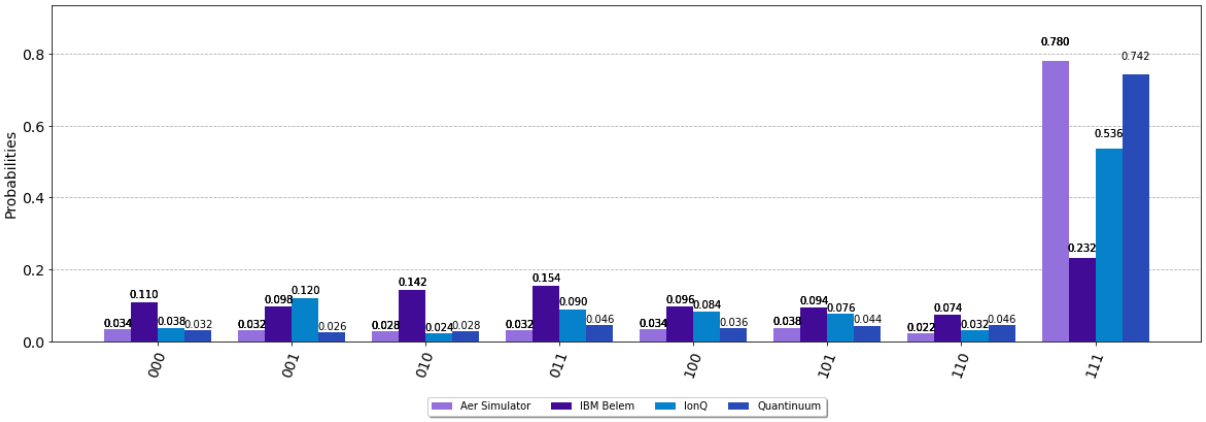

Figura 38. Histograma com resultados para 3 qubits e 1 iteração.

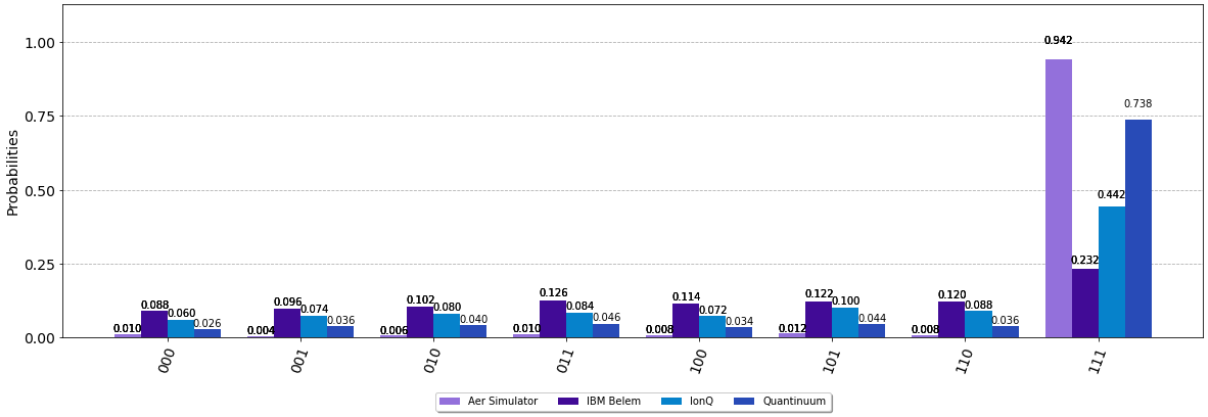

Figura 39. Histograma com resultados para 3 *qubits* e número de iterações ideal.

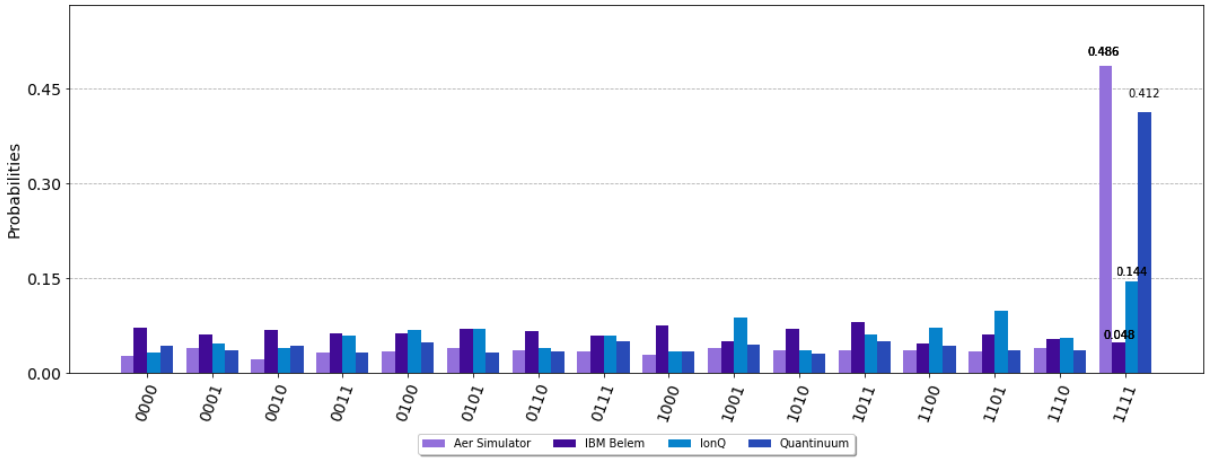

Figura 40. Histograma com resultados para 4 *qubits* e 1 iteração.

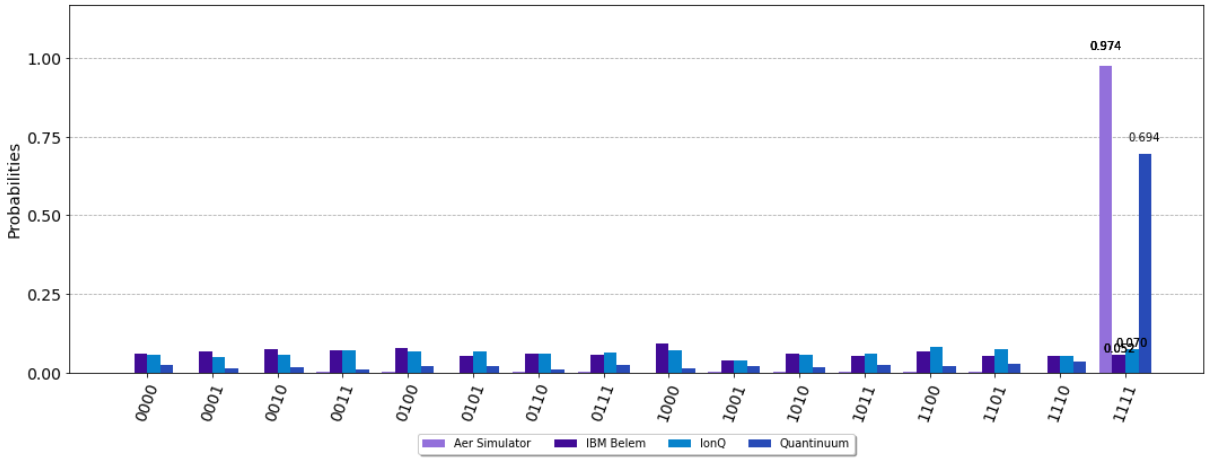

Figura 41. Histograma com resultados para 4 *qubits* e número de iterações ideal.

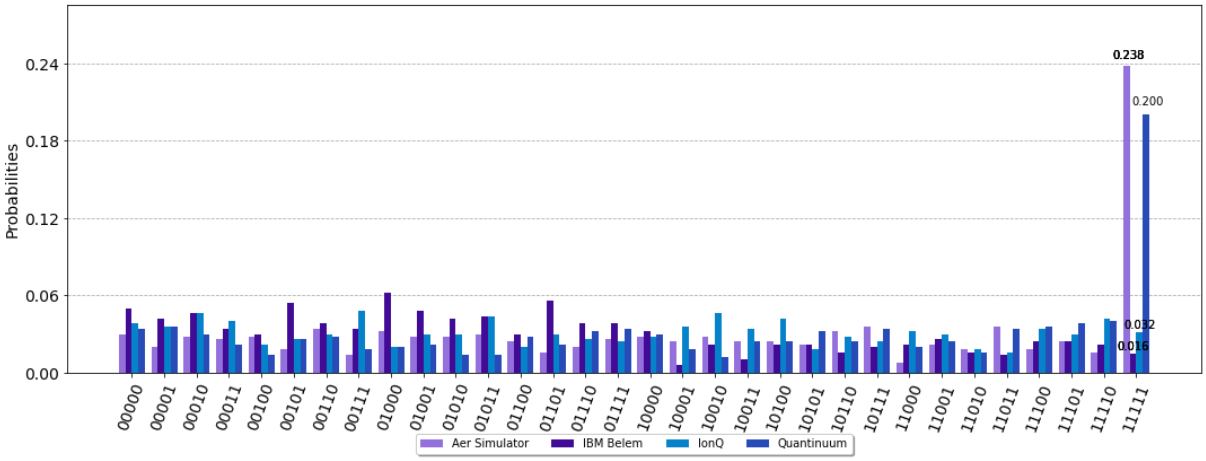

Figura 42. Histograma com resultados para 5 *qubits* e 1 iteração.

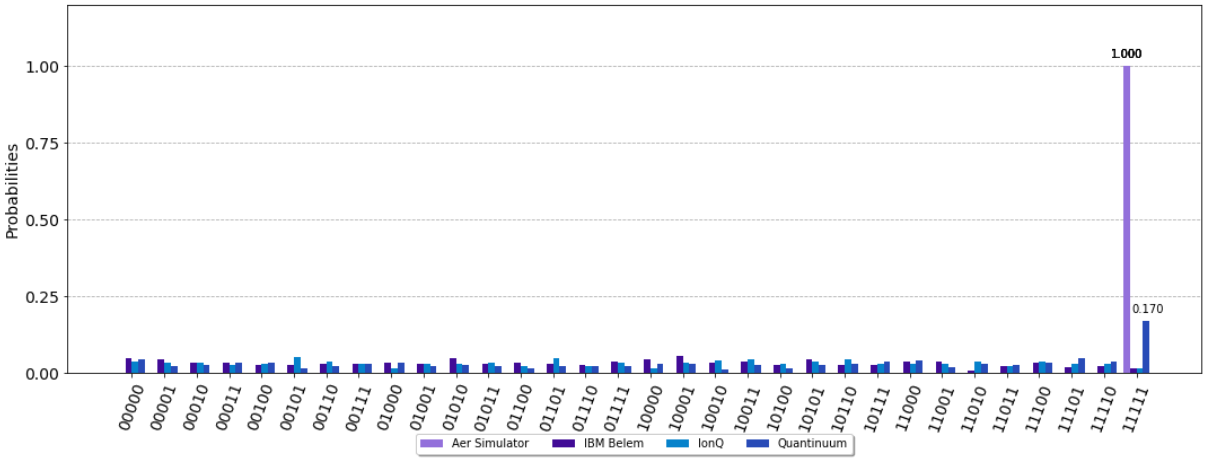

Figura 43. Histograma com resultados para 5 *qubits* e número de iterações ideal.

4.2 RESULTADOS OBTIDOS COM 1000 *SHOTS*

Para a implementação com 1000 repetições do circuito foram utilizados os processadores da IBM, IonQ, Rigetti e OQC.

A Amazon não oferece acesso ao tempo de execução das tarefas enviadas aos *hardwares* quânticos disponibilizados. Por esse motivo essa métrica não pode ser analisada nesse cenário.

Além disso, o processador da Rigetti não obteve sucesso ao rodar o circuito para 5 *qubits* com o número de iterações ideal – uma mensagem de "circuito muito extenso" retornava nesse cenário. O mesmo ocorreu para o processador Lucy, da OQC, para 4 e 5 *qubits*, em ambas as configurações (uma iteração ou número de iterações ideal). Por esse motivo, na **Tabela 4.3** o *ASP Efetivo* correspondente a esses processadores está zerado nesses casos.

Tabela 4.3 Informações de desempenho dos processadores para 1000 *shots*.

Nº de *qubits*	**Nº de iterações**	***ASP* Teórico (%)**	***Hardware***	**ASP Efetivo (%)**
2	*1*	100	Simulador Aer	100
			Belem (IBM)	86.6
			Harmony (IonQ)	94.2
			Aspen M-2 (Rigetti)	84.6
			Lucy (OQC)	36.6
3	*1*	78.13	Simulador Aer	78.5
			Belem (IBM)	18.4
			Harmony (IonQ)	10.4
			Aspen M-2 (Rigetti)	11.9
			Lucy (OQC)	15.6
3	*2*	94.53	Simulador Aer	94.7
			Belem (IBM)	13.5
			Harmony (IonQ)	11.9
			Aspen M-2 (Rigetti)	23.1
			Lucy (OQC)	10.6
4	*1*	47,27	Simulador Aer	46.1
			Belem (IBM)	3.6
			Harmony (IonQ)	7.3
			Aspen M-2 (Rigetti)	12.1
			Lucy (OQC)	0.0

4	*3*	96,13	Simulador Aer	96.0
			Belem (IBM)	4.1
			Harmony (IonQ)	7.3
			Aspen M-2 (Rigetti)	16.7
			Lucy (OQC)	0.0
5	*1*	25,83	Simulador Aer	26.4
			Belem (IBM)	1.0
			Harmony (IonQ)	3.7
			Aspen M-2 (Rigetti)	12.7
			Lucy (OQC)	0.0
5	*4*	99,92	Simulador Aer	100
			Belem (IBM)	0.7
			Harmony (IonQ)	3.8
			Aspen M-2 (Rigetti)	0.0
			Lucy (OQC)	0.0

A **Figura 44** mostra o gráfico comparativo dos valores de *ASP* obtidos em cada processador e, na **Tabela 4.4**, está o custo total de cada *hardware* para execução dos circuitos.

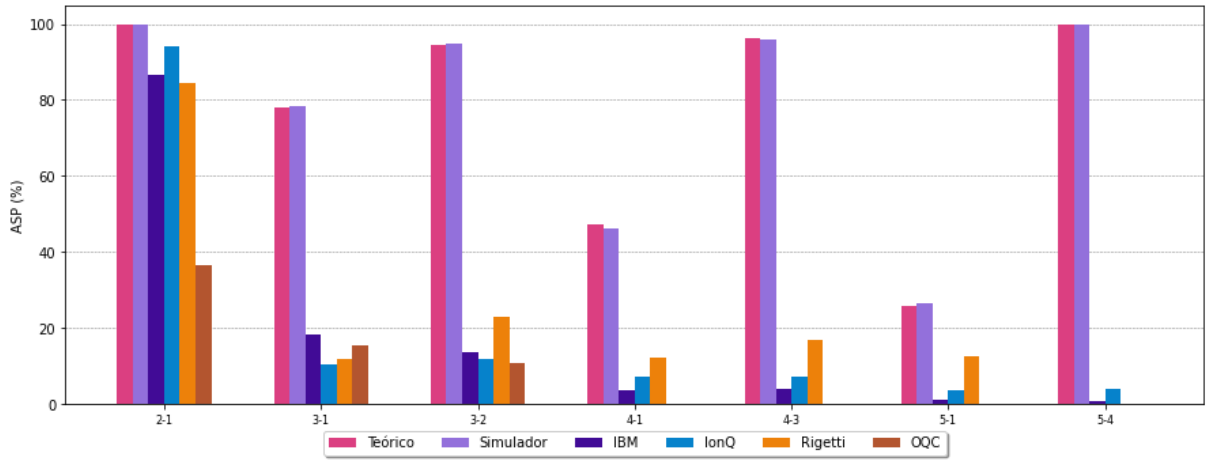

Figura 44. Gráfico do *ASP* obtido em cada processador para cada um dos circuitos.

Tabela 4.4 Custo total para execução dos circuitos.

Hardware	**Custo**
Belem (IBM)	U$ 0.00
Harmony (IonQ)	U$ 197.80
Aspen M-2 (Rigetti)	U$ 3.25
Lucy (OQC)	U$ 4.55

4.2.1 Histogramas

A seguir estão os histogramas gerados para cada um dos circuitos implementados com 1000 *shots*.

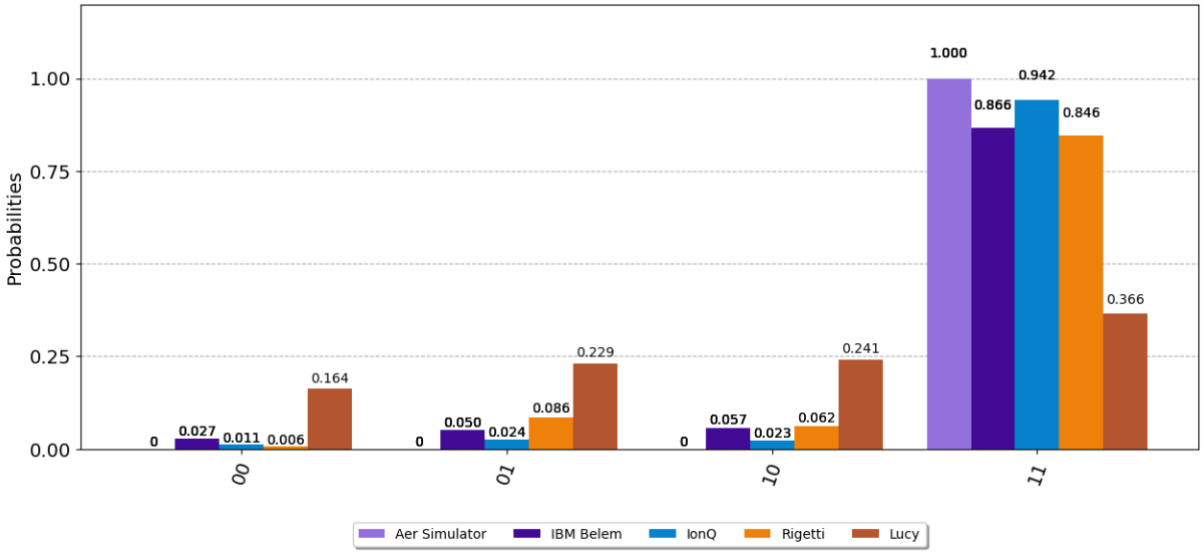

Figura 45. Histograma com resultados para 2 *qubits*.

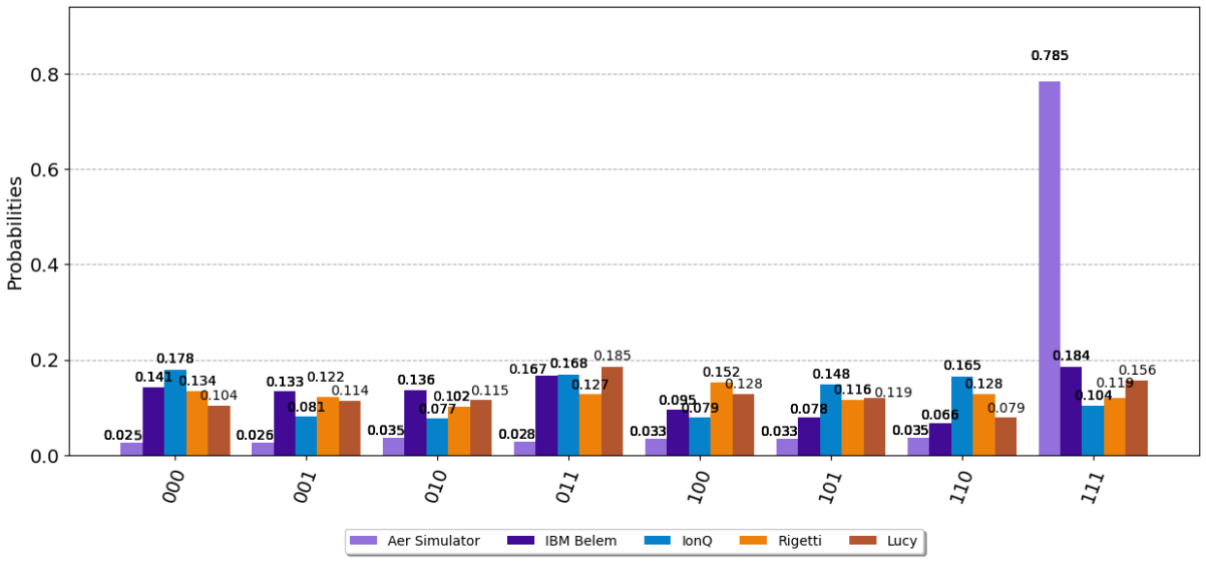

Figura 46. Histograma com resultados para 3 qubits e 1 iteração.

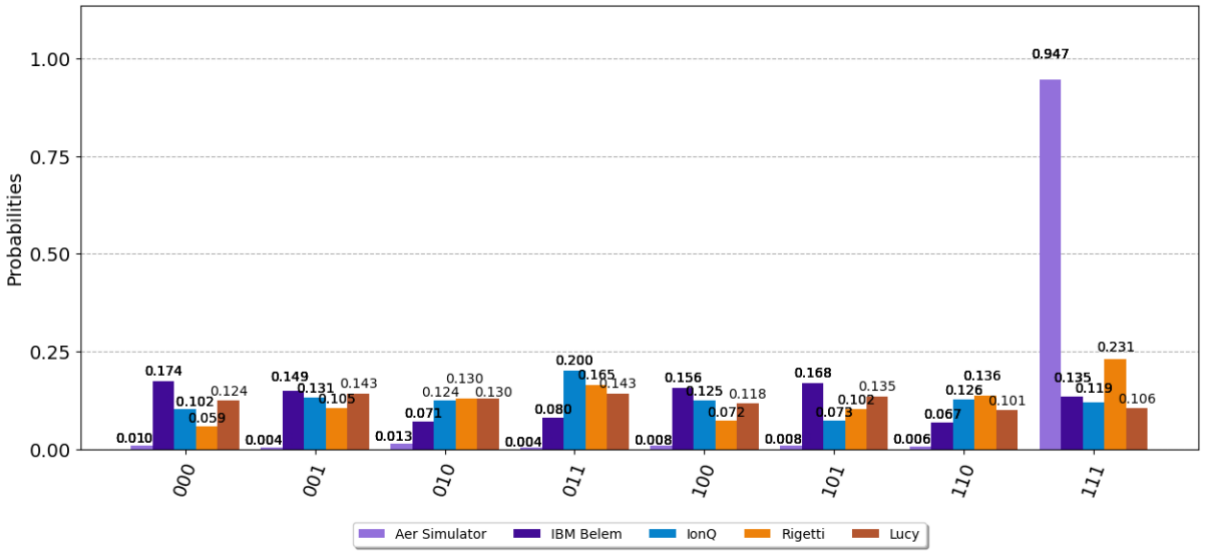

Figura 47. Histograma com resultados para 3 *qubits* e número de iterações ideal.

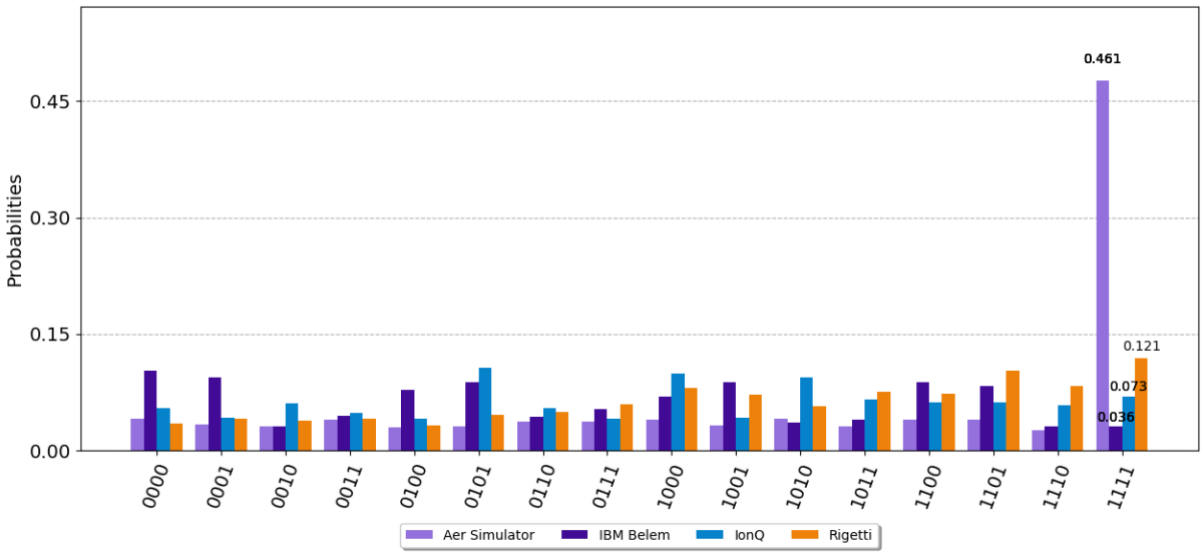

Figura 48. Histograma com resultados para 4 *qubits* e 1 iteração.

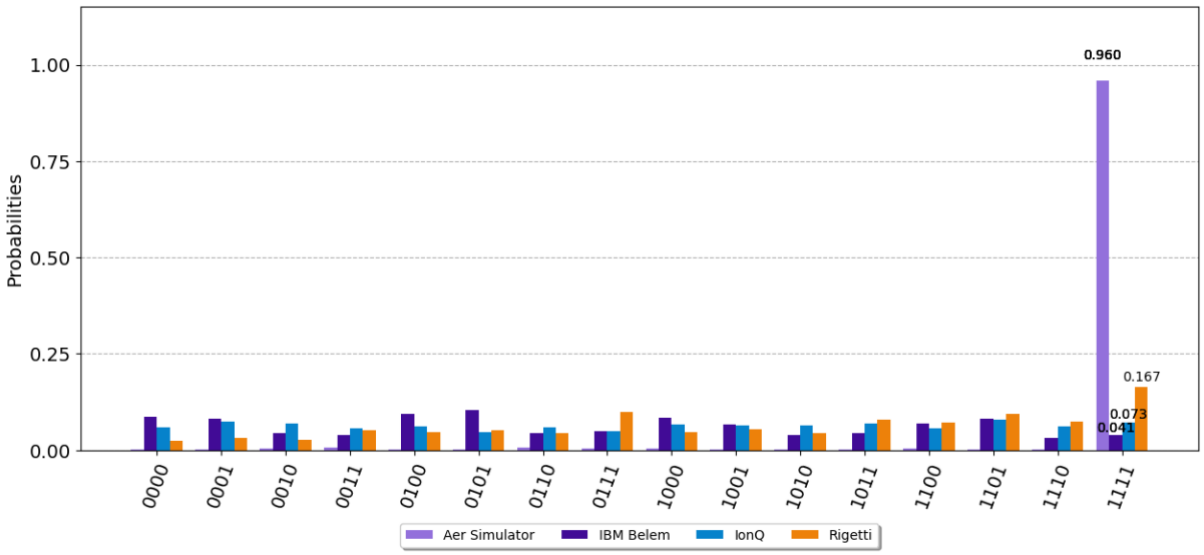

Figura 49. Histograma com resultados para 4 *qubits* e número de iterações ideal.

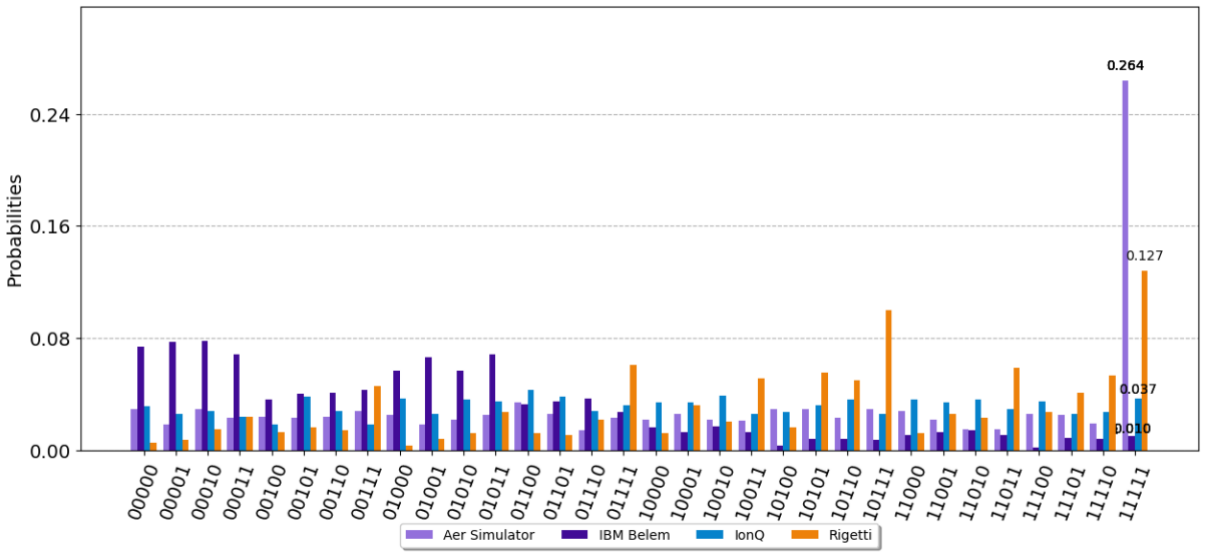

Figura 50. Histograma com resultados para 5 *qubits* e 1 iteração.

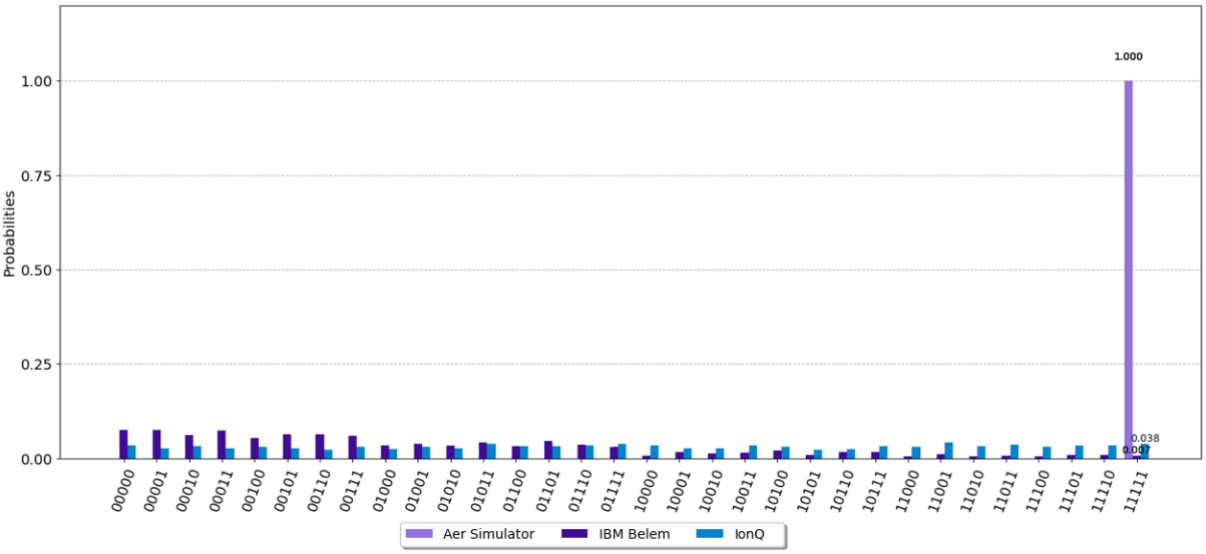

Figura 51. Histograma com resultados para 5 *qubits* e número de iterações ideal.

4.3 RESULTADOS OBTIDOS COM 2000 *SHOTS*

Para a implementação com 2000 repetições do circuito foram utilizados com sucesso apenas os processadores da IBM e da IonQ, cujos resultados estão descritos na **Tabela 4.5**.

Tabela 4.5 Informações de desempenho dos processadores para 2000 *shots*.

Nº de *qubits*	**Nº de iterações**	***ASP* Teórico (%)**	***Hardware***	**ASP Efetivo (%)**	**Tempo de Execução**	**Tempo Total**
2	*1*	100	Simulador Aer	100	-	-
			Belem (IBM)	89.50	00:00:02	01:53:58
			Harmony (IonQ)	96.90	00:00:13	00:19:26
3	*1*	78.13	Simulador Aer	78.00	-	-
			Belem (IBM)	20.80	00:00:03	01:46:47
			Harmony (IonQ)	49.55	00:00:17	00:03:08
3	*2*	94.53	Simulador Aer	95.10	-	-
			Belem (IBM)	17.90	00:00:03	00:47:14
			Harmony (IonQ)	26.10	00:00:23	00:02:59
4	*1*	47,27	Simulador Aer	47.9	-	-
			Belem (IBM)	5.70	00:00:04	00:47:31
			Harmony (IonQ)	22.60	00:00:28	00:05:28
4	*3*	96,13	Simulador Aer	96.50	-	-
			Belem (IBM)	4.70	00:00:05	00:47:47
			Harmony (IonQ)	6.45	00:00:54	00:05:31
5	*1*	25,83	Simulador Aer	24.90	-	-
			Belem (IBM)	1.10	00:00:05	00:47:24
			Harmony (IonQ)	2.95	00:00:41	00:03:36
5	*4*	99,92	Simulador Aer	100	-	-
			Belem (IBM)	0.80	00:00:12	00:48:12
			Harmony (IonQ)	4.00	00:02:05	00:39:09

A **Figura 52** mostra o gráfico comparativo dos valores de *ASP* obtidos em cada processador e a **Figura 53** mostra o tempo utilizado por cada processador para executar cada circuito. Na **Tabela 4.4** está o custo total de cada *hardware* para execução dos circuitos.

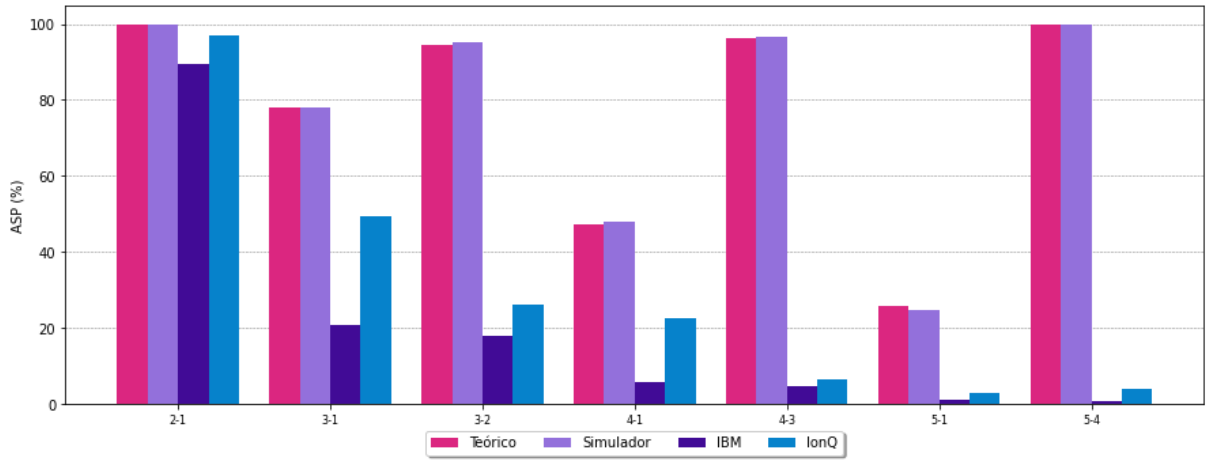

Figura 52. Gráfico do *ASP* obtido em cada processador para cada um dos circuitos.

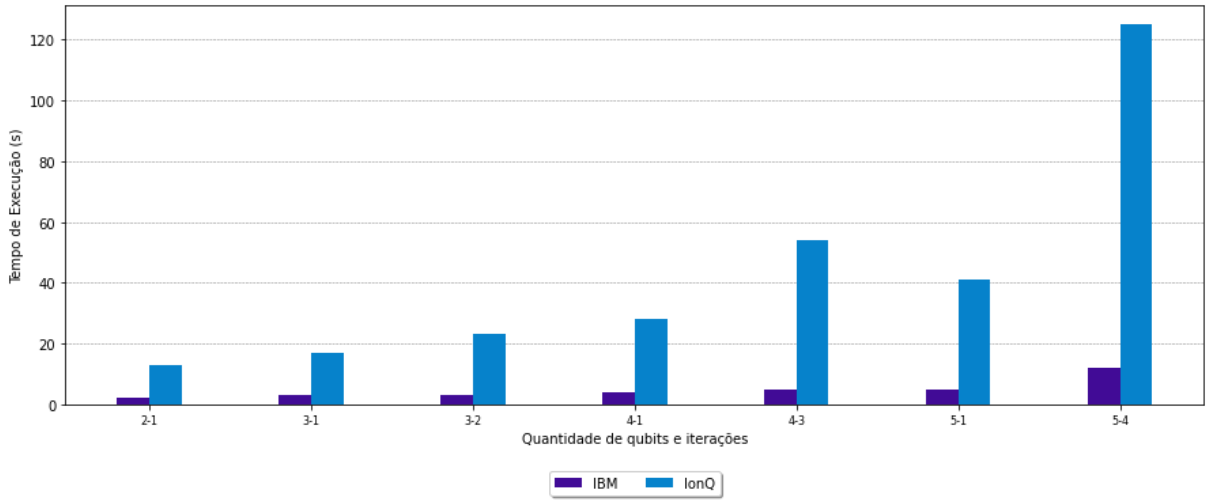

Figura 53. Tempo de execução requerido por cada processador para cada um dos circuitos.

Tabela 4.6 Custo total para execução dos circuitos.

Hardware	**Custo**
Belem (IBM)	U$ 0.00
Harmony (IonQ)	U$ 332.58

4.3.1 Histogramas

A seguir estão os histogramas gerados para cada um dos circuitos implementados com 2000 *shots*.

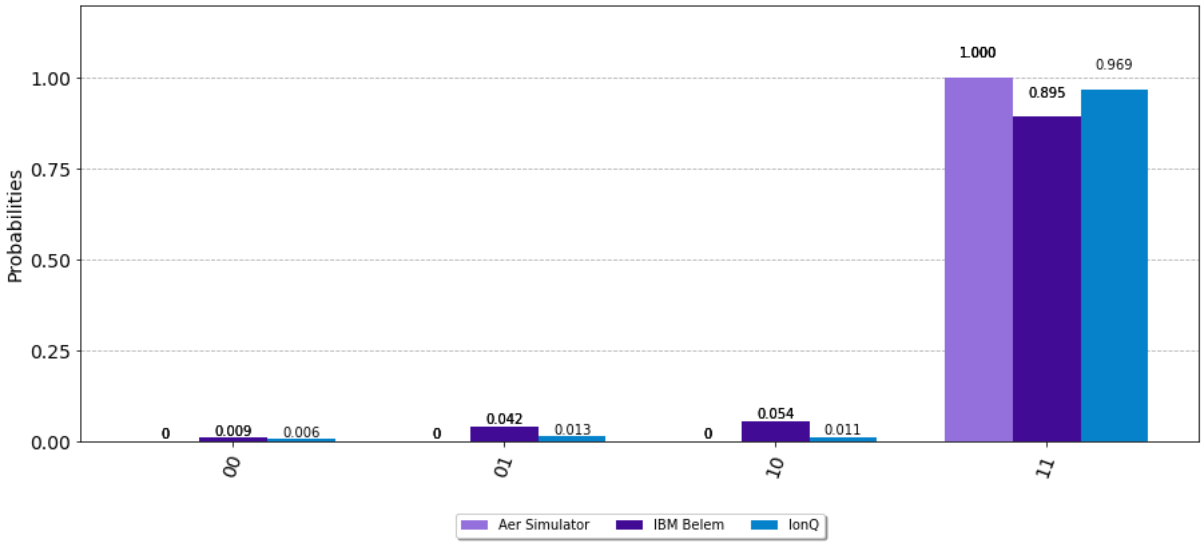

Figura 54. Histograma com resultados para 2 *qubits*.

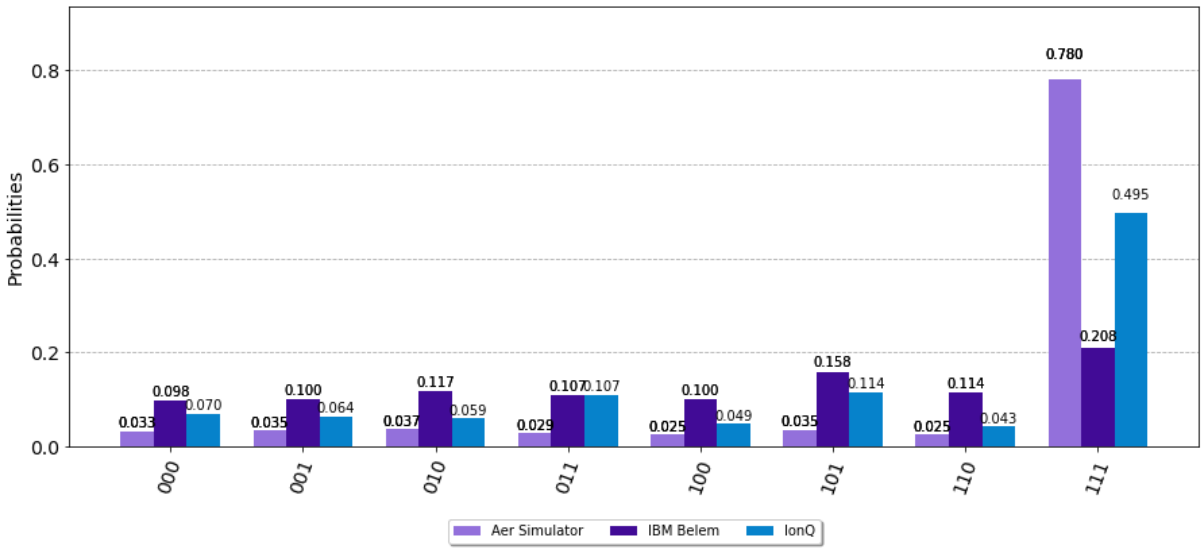

Figura 55. Histograma com resultados para 3 qubits e 1 iteração.

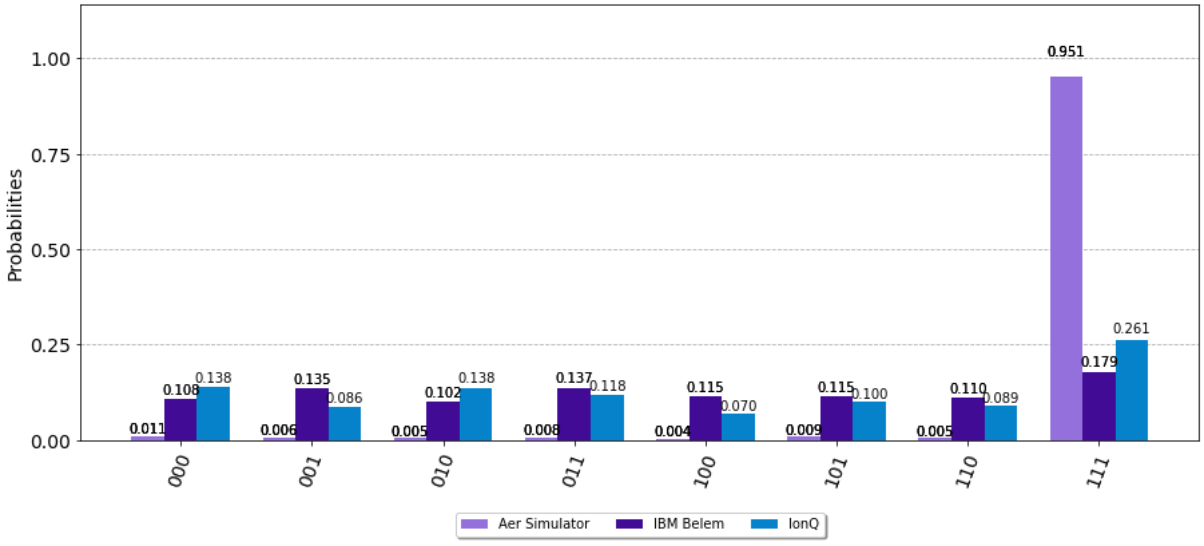

Figura 56. Histograma com resultados para 3 *qubits* e número de iterações ideal.

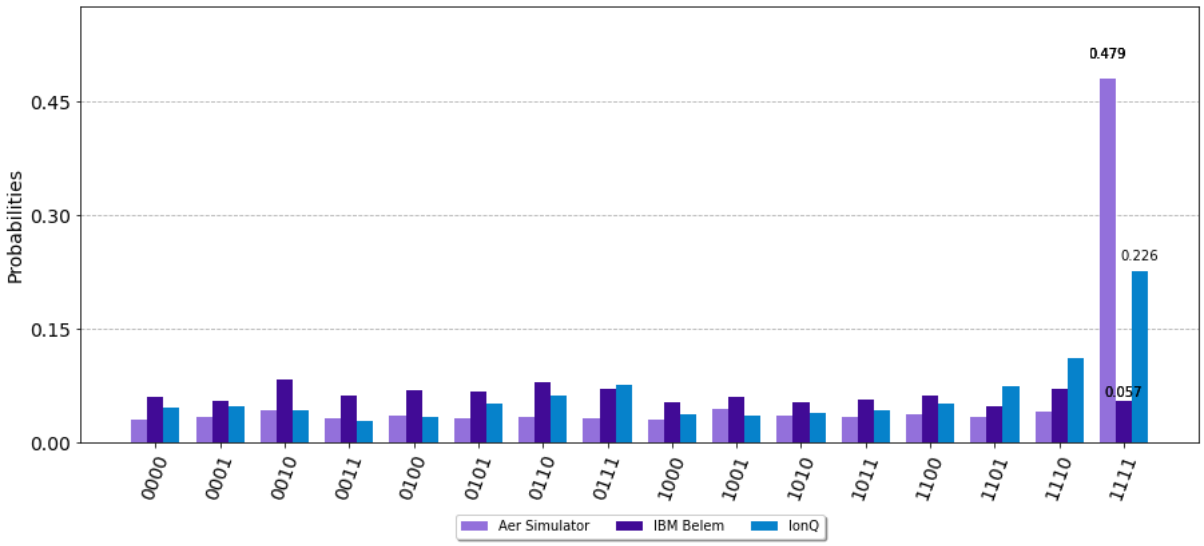

Figura 57. Histograma com resultados para 4 *qubits* e 1 iteração.

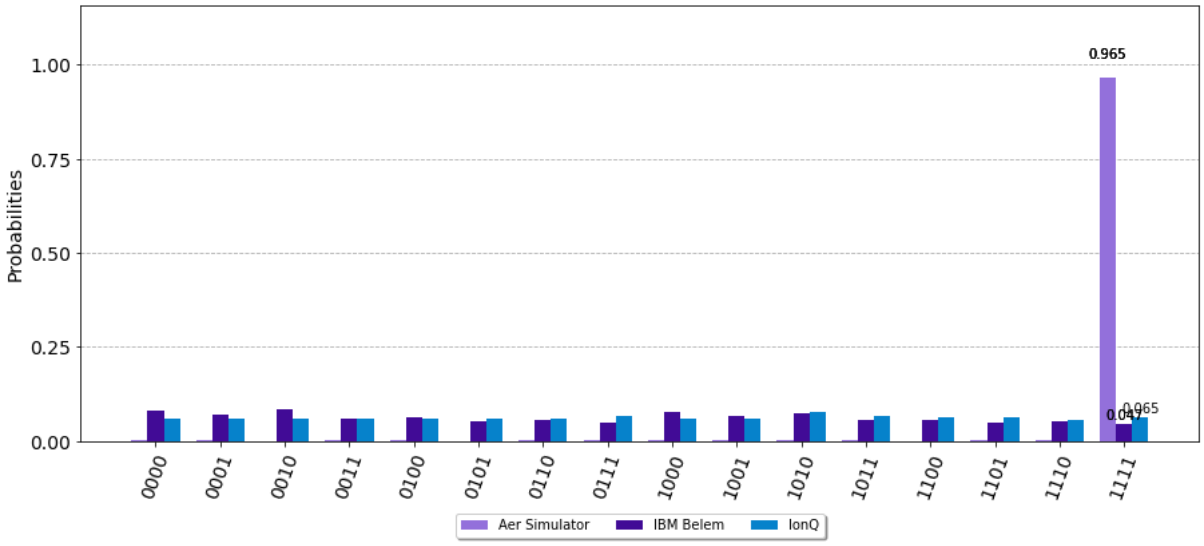

Figura 58. Histograma com resultados para 4 *qubits* e número de iterações ideal.

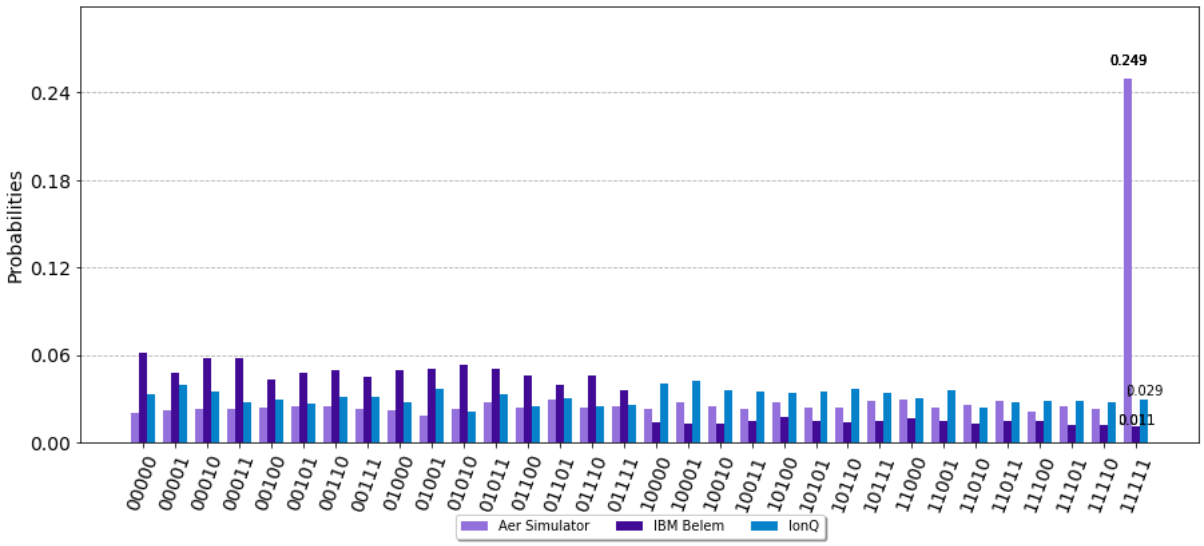

Figura 59. Histograma com resultados para 5 *qubits* e 1 iteração.

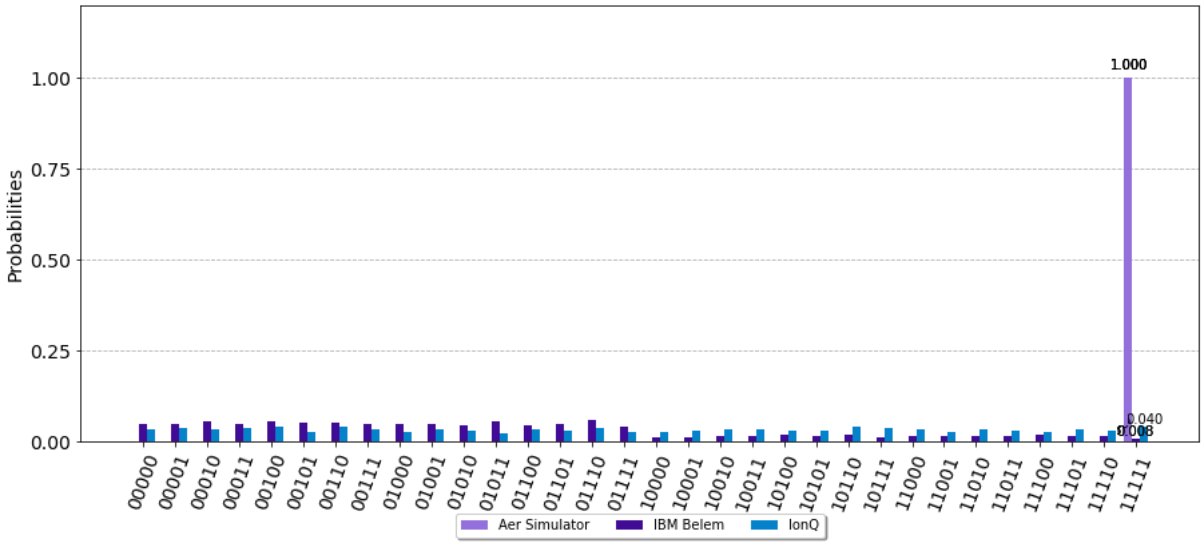

Figura 60. Histograma com resultados para 5 *qubits* e número de iterações ideal.

5 CONCLUSÃO

Os resultados são similares aos encontrados em (ADEDOYIN et al., 2022; BLOMKVIST KARLSSON; STRÖMBERG, 2018; FIGGATT et al., 2017; HU, 2018; JOSHI; GUPTA, 2021; MANDVIWALLA; OHSHIRO; JI, 2018). Contudo, nesse projeto mais *hardwares* foram testados.

A plataforma da Amazon está com um erro de integração ao Qiskit, que faz o circuito ser executado sempre 1000 vezes, não importando o valor de *shots* estipulado pelo algoritmo. Assim, os *hardwares* Lucy, da OQC, e Aspen M-2, da Rigetti, só puderam ser testados nessa condição.

A tecnologia *Coaxmon* da OQC, a *priori*, não trouxe grandes benefícios de performance ao *hardware*, pelo contrário: mesmo tendo uma quantidade de portas e profundidade similares ao processador da Rigetti – o que indica uma topologia de construção equivalente – seus valores de *ASP* foram significativamente inferiores e os piores obtidos dentre todos os *hardwares* testados nesse estudo.

O Aspen M-2, da Rigetti, se destaca por ser um processador dos mais acessíveis, como mostra a **Tabela 4.4**, e com uma boa performance obtida para os valores de *ASP*. Além disso, esse *hardware*, juntamente com o da IonQ, são os únicos disponíveis em mais de uma plataforma. Infelizmente, devido a limitação de integração da plataforma da Amazon ao Qiskit, não foi possível obter seus tempos de processamento. Na plataforma Azure, da Microsoft, ele estava inativo na época de realização desse estudo.

O processador Harmony, da IonQ, possui o melhor custo-benefício de acordo com os resultados obtidos nesse estudo. Além disso, a empresa se destaca pela abrangência de todos os provedores, bibliotecas e ferramentas de nuvem mais populares (IONQ, 2022).

Como fica evidente no gráfico da **Figura 35** o *hardware* com melhor desempenho de *ASP* foi o da Quantinuum. Entretanto, este também é o processador com o custo mais elevado, como mostra a **Tabela 4.2**, motivo pelo qual foi possível realizar apenas execuções de 500 *shots* nesse processador. Além disso, é o mais demorado para executar os circuitos, chegando a mais de 30 minutos para executar o circuito de 5 *qubits*, sem contar o tempo em fila.

Embora o *hardware* da IBM não tenha tido um desempenho dos melhores, a empresa se destaca pela acessibilidade de seus processadores. É a única onde é possível executar 20.000 *shots* gratuitamente. Para se ter uma ideia, o preço da IonQ para executar um circuito de 5 *qubits* com número de iterações ideal (4 iterações) seria de U$1.733,40. No da Quantinuum esse valor seria U$11.613,00, por exemplo. Enquanto que, no processador da IBM, essa tarefa seria gratuita. Outro ponto forte é a rapidez do processador, que levou apenas alguns segundos para executar cada tarefa, sendo o processador mais rápido entre os testados, mesmo tendo que aplicar muito mais operações do que os outros devido ao elevado número de portas que necessitou para a implementação do algoritmo.

Além disso, a plataforma da IBM é a que conta com uma experiência de usuário mais rica. O sistema é todo integrado – com histogramas desenhados diretamente na plataforma, por exemplo – e sem a necessidade de ter que lidar com diversos *tokens*. Basta escolher o provedor e executar o algoritmo. As demais têm um uso um pouco mais dificultado e diversas vezes apresentam falhas na execução da tarefa – o que é bem frustrante pois o tempo da tarefa em fila costuma ser elevado nessas plataformas (com média em torno de 20h). No da IBM isso não costuma ocorrer e o tempo médio em fila é significativamente menor (em torno de 3h).

Embora tenha se tentado incluir os *hardwares* da Google nesse estudo, atualmente a empresa não os disponibiliza para acesso ao público, liberando acesso apenas a seus simuladores. É possível utilizar o SDK da empresa, Cirq, para executar circuitos em *hardwares* de terceiros. Contudo, nesse estudo se limitou à utilização apenas do Qiskit a fim de evitar problemas de performance de algoritmo.

Por fim, vale ressaltar que, pelo menos para o público em geral, ainda é mais vantajoso realizar buscas em bancos de dados utilizando algoritmos clássicos. Também fica evidente pelos gráficos apresentados que, devido ao baixo número de *qubits* disponibilizados atualmente e ao nível de ruido presente nos *hardwares*, é mais rápido e preciso utilizar simuladores quânticos do que os processadores efetivamente quânticos.

Ainda assim, é preciso lembrar que em menos de 50 anos a computação clássica deu um salto de enorme, partindo de processadores com alguns milhares de transistores[16] em 1970 a dezenas de bilhões deles atualmente (OWD, 2017). Os computadores quânticos disponíveis atualmente são apenas os primeiros a serem construídos, e, se a tendência se mantiver, logo estarão disponíveis processadores com um número bem maior de *qubits* e com menos ruído, tornando possível utilizar algoritmos como os de Grover para buscas muito mais rápidas em bancos de dados do que os algoritmos clássicos atuais.

[16] Os transistores são os dispositivos físicos que implementam os *bits* clássicos, 0 e 1.

5.1 DIREÇÕES FUTURAS

Um processador que tem chamado a atenção por sua tecnologia percursora é o Borealis, da empresa Xanadu, que possui *qubits* baseados em fotônica quântica e usa fontes de luz quânticas que emitem pulsos de luz comprimidos para compatibilidade com a computação quântica variável contínua, um paradigma que usa estados quânticos contínuos, conhecidos como *qumodes*. O dispositivo implementa um protocolo específico, conhecido como *Gaussian Boson Sampling* (GBS) (XANADU, 2022). Ele está disponível através da plataforma da Amazon e necessita do *Amazon Braket Python* SDK para ser implementado, por essa razão não foi explorado nesse estudo. Contudo, é uma boa opção para estudos futuros.

Como o *hardware* da Rigetti está disponível também na Azure, é interessante utilizá-lo também nessa plataforma a fim de saber seu tempo de processamento – informação não disponível pela Amazon – e comparar valores.

Durante a execução desse projeto, um patrocínio de U$1.000,00 foi aprovado pela Microsoft para ser utilizado em tarefas do processador da Quantinuum, pela plataforma Azure, para desenvolvimento de pesquisa. Com isso, a ideia é expandir alguma métrica aferida nesse projeto e realizar um novo estudo, a ser publicado no meio científico em breve.

6 CRONOGRAMA

A elaboração e execução do projeto seguiu o cronograma apresentado na **Tabela 6.1**.

Tabela 6.1 Cronograma do Projeto

Período	Etapa	Situação
25/01/2022 –15/02/2022	Estudo sobre desenvolvimento de trabalhos de conclusão de Engenharia	Concluído
16/02/2022 – 25/04/2022	Revisão bibliográfica e desenvolvimento da fundamentação teórica	Concluído
26/04/2022 – 17/05/2022	Estudo sobre o panorama global e nacional da computação quântica	Concluído
18/05/2022 – 12/06/2022	Recesso Escolar	-
13/06/2022 – 03/07/2022	Desenvolvimento dos tópicos restantes do projeto e estudo das plataformas de implementação	Concluído
04/07/2022 – 15/07/2022	Ajustes finais e apresentação do TDEF I para a banca de avaliação	Concluído
16/07/2022 – 10/08/2022	Revisão bibliográfica sobre técnicas para aferir desempenho de *hardwares* quânticos	Concluído
16/08/2022 – 25/08/2022	Revisão bibliográfica sobre técnicas de codificação de bases quânticas	Concluído
26/08/2022 – 20/09/2022	Implementação do algoritmo e análise dos resultados	Concluído
21/09/2022 – 01/10/2022	Ajustes finais e apresentação do TDEF II para a banca de avaliação	Concluído

REFERÊNCIAS

ADEDOYIN, A. et al. Quantum Algorithm Implementations for Beginners. 2022.

AMAZON. **Amazon Braket Python SDK**. Disponível em: <https://amazon-braket-sdk-python.readthedocs.io/en/latest/>. Acesso em: 15 set. 2022a.

AMAZON. **Introducing the Qiskit provider for Amazon Braket**. Disponível em: <https://aws.amazon.com/pt/blogs/quantum-computing/introducing-the-qiskit-provider-for-amazon-braket/>. Acesso em: 15 set. 2022b.

AMAZON BRAKET PRICING. **Quantum Computer and Simulator**. Disponível em: <https://aws.amazon.com/pt/braket/pricing/>. Acesso em: 15 set. 2022.

AMAZON WEB SERVICES. **OQC**. Disponível em: <https://aws.amazon.com/pt/braket/quantum-computers/oqc/?nc1=h_ls>. Acesso em: 16 set. 2022.

AZURE QUANTUM. **Provedor de computação quântica do IonQ para o Azure Quantum**. Disponível em: <https://docs.microsoft.com/pt-br/azure/quantum/provider-ionq#ionq-harmony-quantum-computer>. Acesso em: 13 set. 2022a.

AZURE QUANTUM. **Provedor de Quantinuum**. Disponível em: <https://docs.microsoft.com/pt-br/azure/quantum/provider-quantinuum>. Acesso em: 14 set. 2022b.

BLOMKVIST KARLSSON, V.; STRÖMBERG, P. **4-qubit Grover's algorithm implemented for the ibmqx5 architectureDEGREE PROJECT COMPUTER SCIENCE**. [s.l: s.n.].

BOYER, M. et al. Tight bounds on quantum searching. 1996.

CDOTRENDS. **Google Partners With Education Institutions to Upskill Australians**. Disponível em: <https://www.cdotrends.com/story/16410/google-partners-education-institutions-upskill-australians>. Acesso em: 4 maio. 2022.

DA CUNHA, C. R. **History & Binary Representation**. [s.l: s.n.].

DEUTSCH, D.; JOZSA, R. Rapid solution of problems by quantum computation. **Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences**, v. 439, n. 1907, p. 553–558, 8 dez. 1992.

DIRAC, P. A. M. A new notation for quantum mechanics. **Mathematical Proceedings of the Cambridge Philosophical Society**, v. 35, n. 3, p. 416–418, 24 jul. 1939.

DIVINCENZO, D. P. The Physical Implementation of Quantum Computation. 2000.

FIGGATT, C. et al. **Complete 3-Qubit Grover Search on a Programmable Quantum Computer**. [s.l: s.n.].

GEORGOPOULOS, K.; EMARY, C.; ZULIANI, P. Quantum Computer Benchmarking via Quantum Algorithms. 17 dez. 2021.

GRIFFITHS, D. J.; SCHROETER, D. F. **Introduction to Quantum Mechanics**. [s.l: s.n.].

GROVER, L. K. A fast quantum mechanical algorithm for database search. 1996.

GUNZI, A. **How to calculate the depth of a quantum circuit in Qiskit?** Disponível em: <https://medium.com/arnaldo-gunzi-quantum/how-to-calculate-the-depth-of-a-quantum-circuit-in-qiskit-868505abc104>. Acesso em: 9 set. 2022.

HU, W. Empirical Analysis of Decision Making of an AI Agent on IBM's 5Q Quantum Computer. **Natural Science**, v. 10, n. 01, p. 45–58, 2018.

IBM RESEARCH. **Quantum computation center opens**. Disponível em: <https://www.ibm.com/blogs/research/2019/09/quantum-computation-center/>. Acesso em: 29 ago. 2022.

IBM RESEARCH BLOG. **IBM Quantum breaks the 100-qubit processor barrier**. Disponível em: <https://research.ibm.com/blog/127-qubit-quantum-processor-eagle#pageStart>. Acesso em: 23 abr. 2022.

IBM RESEARCH BLOG. **IBM Quantum Spring Challenge**. Disponível em: <https://research.ibm.com/blog/quantum-spring-challenge-2022>. Acesso em: 4 maio. 2022.

IEEE QUANTUM. **Summary of the IEEE Workshop on Benchmarking Quantum Computational Devices and Systems**. Disponível em: <https://quantum.ieee.org/education/quantum-supremacy-and-quantum-computer-performance>. Acesso em: 1 jul. 2022.

IONQ. **Trapped Ion Quantum Computing**. Disponível em: <https://ionq.com/>. Acesso em: 29 set. 2022.

JAEGER, L. **The Second Quantum Revolution**. [s.l.] Springer, 2018. v. 1

JOSHI, S.; GUPTA, D. Grover's Algorithm in a 4-Qubit Search Space. **Journal of Quantum Computing**, v. 3, n. 4, p. 137–150, 2021.

LANGIONE, M. et al. **The Race to Quantum Advantage Depends on Benchmarking**. [s.l: s.n.]. Disponível em: <https://www.bcg.com/publications/2022/value-of-quantum-computing-benchmarks>. Acesso em: 1 jul. 2022.

LINKE, N. M. et al. Experimental comparison of two quantum computing architectures. **Proceedings of the National Academy of Sciences of the United States of America**, v. 114, n. 13, p. 3305–3310, 28 mar. 2017.

MANDVIWALLA, A.; OHSHIRO, K.; JI, B. **Implementing Grover's Algorithm on the IBM Quantum Computers**. [s.l: s.n.].

MICROSOFT. **Implementar o algoritmo de pesquisa do Grover no Q#**. Disponível em: <https://docs.microsoft.com/pt-br/azure/quantum/tutorial-qdk-grovers-search?tabs=tabid-visualstudio>. Acesso em: 22 jun. 2022.

MILLER, D. A. B. **Quantum Mechanics for Scientists and Engineers**. [s.l.] Cambridge, 2008.

MILLS, A. R. et al. Two-qubit silicon quantum processor with operation fidelity exceeding 99%. 23 nov. 2021.

OWD. **Moore's Law: The number of transistors per microprocessor**. Disponível em: <https://ourworldindata.org/grapher/transistors-per-microprocessor>. Acesso em: 16 out. 2022.

NIELSEN, M. A.; CHUANG, I. L. **Quantum computation and quantum information**. [s.l.] Cambridge University Press, 2010.

OXFORD QUANTUM CIRCUITS. **Technology**. Disponível em: <https://oxfordquantumcircuits.com/technology>. Acesso em: 16 set. 2022.

PATINFORMATICS LLC. Practical Quantum Computing: A Patent Landscape Report. 2017.

PIRES, F. **Quantum Computing Benchmarks Begin to Take Shape**. Disponível em: <https://www.tomshardware.com/news/quantum-computing-benchmarks-begin-to-take-shape>. Acesso em: 1 jul. 2022.

PRADO, S.; DILLENBURG, R. O Algoritmo de Grover. 2014.

PRESKILL, J. Quantum computing in the NISQ era and beyond. **Quantum**, v. 2, 6 ago. 2018.

QISKIT. **Introducing Qiskit Aer: A high performance simulator framework for quantum circuits**. Disponível em: <https://medium.com/qiskit/qiskit-aer-d09d0fac7759>. Acesso em: 24 set. 2022.

QISKIT. **Algoritmo de Grover e Amplificação De Amplitude**. Disponível em: <https://qiskit.org/documentation/locale/pt_BR/tutorials/algorithms/06_grover.html>. Acesso em: 22 jun. 2022a.

QISKIT. **Grover's Algorithm**. Disponível em: <https://qiskit.org/textbook/ch-algorithms/grover.html>. Acesso em: 22 jun. 2022b.

QISKIT 0.38.0. **Transpiler (qiskit.transpiler)**. Disponível em: <https://qiskit.org/documentation/apidoc/transpiler.html>. Acesso em: 16 set. 2022.

QUANTINUUM. **Products | H1**. Disponível em: <https://www.quantinuum.com/products/h1>. Acesso em: 15 set. 2022.

Quantum Manifesto A New Era of Technology. 2016.

QURECA. **Overview on quantum initiatives worldwide**. Disponível em: <https://qureca.com/overview-on-quantum-initiatives-worldwide-update-2022/>. Acesso em: 30 jun. 2022.

RIGETTI. **Rigetti QCS**. Disponível em: <https://qcs.rigetti.com/qpus>. Acesso em: 15 set. 2022.

R.P. FEYNMAN. **Simulating physics with computers**. [s.l.] Int. J. Theor. Phys., 1982.

SCHULD, M.; PETRUCCIONE, F. Supervised Learning with Quantum Computers. Em: [s.l: s.n.]. p. 108–114.

SCHULD, M.; PETRUCCIONE, F. **Supervised Learning with Quantum Computers**. [s.l: s.n.].

SCHULD, M.; PETRUCCIONE, F. Search and Amplitude Amplification. Em: **Machine Learning with Quantum Computers**. Second Edition ed. [s.l: s.n.]. p. 256–269.

SHOR, P. W. **Algorithms for quantum computation: discrete logarithms and factoring**. Proceedings 35th Annual Symposium on Foundations of Computer Science. **Anais**...1994.

WHALEN, J. **Chinese scientists are at the forefront of the quantum revolution**. Disponível em: <https://www.washingtonpost.com/business/2019/08/18/quantum-revolution-is-coming-chinese-scientists-are-forefront/>. Acesso em: 4 maio. 2022.

WRIGHT, K. et al. Benchmarking an 11-qubit quantum computer. **Nature Communications**, v. 10, n. 1, 1 dez. 2019.

XANADU. **Borealis**. Disponível em: <https://www.xanadu.ai/products/borealis/>. Acesso em: 29 set. 2022.

YAN, S. **China launches satellite aimed at "hack-proof" communications**. Disponível em: <https://money.cnn.com/2016/08/16/technology/china-quantum-satellite/index.html>. Acesso em: 30 abr. 2022.

YE, A. **Grover's Algorithm — Quantum Computing**. Disponível em: <https://medium.com/swlh/grovers-algorithm-quantum-computing-1171e826bcfb>. Acesso em: 3 ago. 2022.

ZHANG, H. et al. Realization of quantum secure direct communication over 100 km fiber with time-bin and phase quantum states. **Light: Science & Applications**, v. 11, n. 1, p. 83, 6 dez. 2022.

ZWERVER, A. M. J. et al. Qubits made by advanced semiconductor manufacturing. **Nature Electronics**, v. 5, n. 3, p. 184–190, mar. 2022.