José Victor Soares Scursulim

Computação Quântica e Teoria Quântica de Correção de Erros

Vitória - ES, Brasil

2021

José Victor Soares Scursulim

Computação Quântica e Teoria Quântica de Correção de Erros

Dissertação apresentada ao PPGFIS-UFES como requisito essencial para a obtenção do grau de Mestre em Física.

Universidade Federal do Espírito Santo – UFES

Centro de Ciências Exatas - CCE

Departamento de Física - DFIS

Programa de Pós-Graduação em Física - PPGFIS

Orientador: Prof. Dr. Galen Mihaylov Sotkov

Coorientador: Prof. Dr. Ulysses Camara da Silva

Vitória - ES, Brasil

2021

Ficha catalográfica disponibilizada pelo Sistema Integrado de Bibliotecas - SIBI/UFES e elaborada pelo autor

---

Soares Scursulim, José Victor, 1993-
S676c Computação quântica e teoria quântica de correção de erros / José Victor Soares Scursulim. - 2021.
198 f. : il.

Orientador: Galen Mihaylov Sotkov.
Coorientador: Ulysses Camara da Silva.
Dissertação (Mestrado em Física) - Universidade Federal do Espírito Santo, Centro de Ciências Exatas.

1. Computação quântica. 2. Teoria quântica. I. Mihaylov Sotkov, Galen. II. Camara da Silva, Ulysses. III. Universidade Federal do Espírito Santo. Centro de Ciências Exatas. IV. Título.

CDU: 53

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

CENTRO DE CIÊNCIAS EXATAS

PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA

"Computação Quântica e Teoria Quântica de Correção de Erros"

JOSÉ VICTOR SOARES SCURSULIM

Dissertação submetida ao Programa de Pós-Graduação em Física da Universidade Federal do Espírito Santo, por videoconferência, como requisito parcial para a obtenção do título de Mestre em Física.
Aprovada por:

Prof. Dr. Itzhak Roditi
(CBPF)

Prof. Dr. Galen Mihaylov Sotkov
(Orientador- PPGFis/UFES)

Prof. Dr. Jardel da Costa Brozeguini
(IFES- Cariacica-ES)

Prof. Dr. Jorge Luis Gonzalez Alfonso
(PPGFis/UFES)

Prof. Dr. Ulysses Câmara da Silva
(DFIS/UFES)

Vitória-ES, 22 de novembro de 2021

Agradecimentos

Gostaria de começar agradecendo a minha família por tudo que me proporcionaram até aqui, sem o esforço de vocês talvez eu nunca teria tido as oportunidades que tive na vida e sou muito grato por elas, além disso não posso deixar de mencionar todo carinho e apoio emocional que recebi ao longos desses anos, talvez não existam palavras ou ações que possam refletir meu agradecimento por essa contribuição essencial na minha vida, mas saibam que amo vocês e que em cada equação que escrevo carrego um pouquinho da imensa gratidão que sinto por tudo que fizeram por mim até aqui.

Aqui também preciso deixar meus agradecimentos aos meus amigos e minhas amigas que se uniram a mim na caminhada da vida, sem vocês eu não teria memórias incríveis para relembrar várias vezes em nossos encontros. Espero que possamos continuar compartilhando nossas conquistas e experiências por muitos anos, desejo a todos vocês muito sucesso nos caminhos que escolheram trilhar e que sejam felizes e realizados em seus projetos de vida. Agradeço pelo impacto positivo que tiveram em minha vida, pois me ajudaram a chegar até aqui, pois como escreveu Guimarães Rosa: "é junto dos bão que a gente fica mió" e também como disse o filósofo estóico Sêneca: "Não te interesse sobre a quantidade de amigos, mas sim sobre a qualidade de vossos amigos".

Também preciso agradecer ao arquétipo da dupla átomo e arquétipo, por ajudar o átomo a encontrar a estabilidade necessária para construir as mais diversas combinações de elementos que possam contribuir positivamente para este e seus semelhantes ao redor.

Devo registrar aqui meu agradecimento ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo financiamento da minha pesquisa durante esses anos dedicados ao mestrado em Física.

Por fim queria agradecer a *International Business Machines* (IBM) por me conceder uma conta de pesquisador para realizar os testes no computador quântico Lagos com 7 qubits e também a comunidade de Qiskit Adovocates por me receberem tão bem no grupo, espero realizar muitas contribuições para a comunidade e ajudá-la a crescer, seja no Brasil ou ao redor do mundo.

"Fall in love with some activity, and do it! Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough. Work as hard and as much as you want to on the things you like to do the best. Don't think about what you want to be, but what you want to do. Keep up some kind of a minimum with other things so that society doesn't stop you from doing anything at all."

- Richard Phillips Feynman

Resumo

A computação quântica deve ser o marco inicial da segunda revolução tecnológica causada pelo aprofundamento do conhecimento da humanidade acerca das propriedades peculiares da mecânica quântica. Para que possamos desfrutar de todo o potencial que esta nova tecnologia promete nos proporcionar, é preciso desenvolver um modelo de computação quântica tolerante a erros. Para isso, devemos ter códigos quânticos de correção de erros implementados que garantam uma supressão exponencial dos erros em cada etapa do processo de execução de um algoritmo. Além disso, precisamos aprender como isolar melhor os qubits para que estes não sofram com os efeitos da descoerência. A mecânica quântica nos impõe sérias restrições para a construção de códigos de correção de erros, uma delas é a impossibilidade de usar cópias como redundância, uma vez que o teorema da não clonagem nos impede de criar uma copiadora universal de estados quânticos. Outro problema é que o processo de medida causa um colapso no estado quântico, o que leva a uma corrupção da informação codificada no estado. Felizmente, encontramos mecanismos para contornar essas questões através do emaranhamento quântico e o formalismo de estabilizadores. O primeiro nos permite recuperar a ideia de redundância através da deslocalização da informação entre os espaços de Hilbert das partes que compartilham as correlações quânticas, já o segundo permite obter informação sobre erros que podem ter corrompido a informação quântica presente no estado sem a necessidade de realizar uma medida direta neste. Vamos considerar exemplos de *surface codes* que representam um dos códigos quânticos que utilizam emaranhamento e estabilizadores para corrigir os erros. Este tipo de código é um dos principais candidatos a serem implementados experimentalmente em larga escala e já apresentam resultados promissores em estudos preliminares.

Palavras-chave: Informação; Mecânica Quântica; Computação Quântica; Estados Emaranhados; Correção Quântica de Erros.

Abstract

The quantum computing must be the initial mark of the second technological revolution caused by the deepening knowledge of the humankind about the peculiar features of the quantum mechanics. If we want to unlock all the potential of this new technology, a development of a fault-tolerant quantum computing will be needed. In order to reach this stage, it is necessary an implementation of quantum error correcting codes that guarantee an exponential suppression of the errors at each step of the execution of an algorithm. Besides that, we need to learn how to isolate our qubits aiming a suppression of the decoherence effects. Quantum mechanics imposes various constraints to the process of construction of the error correction codes, one of them is the impossibility of using copies as redundancy mechanism, once no-cloning theorem does not allow the creation of an universal copy machine of quantum states. Another issue to be noted within this context, is that the measurement process causes a collapse in the quantum state, thus leading to a corruption of the information encoded in the state. Fortunately, there are some ways to bypass the problems pointed out through the entanglement and the stabilizer formalism, the first one allows us to recover the redundancy through the delocalization of the information in the Hilbert spaces of the parts that share quantum correlations, while the second one provides a process of gathering information about the errors that can occur in the quantum state without measuring it. The surface codes are one of the quantum codes that use the features highlighted to correct the errors, this type of code is one of the main candidates to be experimentally implemented on a large scale and it presents promising results in the preliminary studies.

Keywords: Information; Quantum Mechanics; Quantum Computing; Entangled States; Quantum Error Correction.

Lista de ilustrações

Sumário

Introdução

Na história da humanidade, podemos ver que diversas ferramentas foram criadas com o intuito de melhorar nosso entendimento da natureza. As grandes revoluções tecnológicas sempre estiveram acompanhadas de descobertas sobre novas formas de geração de energia e também do desenvolvimento de novas formas de computar, armazenar e transmitir informação. Existem vestígios que comprovam que ferramentas voltadas a computação e desenvolvimento de ideias algorítmicas rudimentares surgiram em várias sociedades ao longo dos séculos, inclusive na antiguidade, há indícios que apontam que no tempo de Hammurabi, os Babilônios desenvolveram ideias nesse sentido [1]. Contudo, ideias mais elaboradas surgiram no século XIX, um exemplo é o de Charles Babbage, que propôs uma máquina analítica, um tipo de computador mecânico que teve seu primeiro programa criado por Ada Lovelace, a primeira programadora da história. Este algoritmo tinha como objetivo calcular valores de funções matemáticas.

A ciência da computação moderna vem sendo desenvolvida desde 1936, neste ano o matemático britânico Alan Turing publicou trabalhos que moldaram conceitos importantes que são usados amplamente nas teorias atuais, por exemplo os conceitos de computador programável e computador universal. Um grande passo no campo de hardware foi dado no ano de 1947, quando John Bardeen, Walter Brattain e Will Shockley desenvolveram o transístor [1]. Em 1948, Claude Shannon publicou seu famoso artigo intitulado *A mathematical Theory of Communication* [2], onde forneceu uma definição precisa e matemática da informação, permitindo mensurar a quantidade de informação que poderia ser comunicada entre dois sistemas diferentes e definir os limites da taxa de transmissão do canal de comunicação. Além disso, ele conseguiu obter uma relação entre ruído e sinal, criou e provou um teorema sobre a existência de códigos de correção de erros, o que acabou possibilitando o desenvolvimento desses códigos, nos permitindo alcançar o estágio atual de comunicação. Outro personagem importante para os campos da física e da computação, foi o matemático hungáro John von Neumann, tendo trabalhado na axiomatização da mecânica quântica, cujos frutos deste trabalho culminaram na obra *Mathematical foundations of quantum mechanics* [3]. Além disso desenvolveu o conceito de entropia de sistemas quânticos. No campo da computação, von Neumann sugeriu que se abandonasse as instruções por cartões furados por programas gravados em memória, pois isso levaria a um ganho de velocidade na execução dos algoritmos,o que levou a uma nova forma de projetar os computadores que ficou conhecida como arquitetura de von Neumann. Outra grande contribuição foi a introdução do conceito de construtor universal, que engloba a ideia de computação universal introduzida por Alan Turing e adiciona a habilidade de poder criar outras máquinas iguais a ele. Em 1965, Gordon Moore, um dos

fundadores da empresa de processadores Intel, constatou empiricamente que o número de transistores em uma unidade central de processamento (CPU) deveria dobrar a cada 24 meses, posteriormente esta constatação ficou conhecida como Lei de Moore.

No início dos anos 80, alguns pesquisadores começaram a vislumbrar novas formas de se fazer computação, uma vez que passamos a conhecer os limites físicos dos computadores clássicos, dado que a Lei de Moore encontrava-se ameaçada devido a miniaturização do transístor, pois esta nos levava a desafios técnicos que dificultavam a manutenção do ritmo de crescimento esperado. Existiam projeções mostrando que em algumas décadas chegaríamos em uma escala de tamanho, onde a manifestação de fenômenos ligados a física quântica, por exemplo o tunelamento, nos forçariam a levá-los em consideração para ter pleno funcionamento das máquinas. Hoje em dia, nos encontramos em um cenário onde a litografia dos chips está na casa dos 7 nm, com projeções de termos chips em 2 nm dentro de alguns anos. Ou seja, o avanço tecnológico está nos levando a termos transistores feitos de poucos átomos e efeitos quânticos cada vez mais relevantes, fazendo com que a computação quântica pareça um caminho natural na evolução da computação.

Aliadas ao problema exposto acima, outras "forças" nos levavam a buscar novos paradigmas de computação, uma delas eram os limites impostos pelas classes de complexidade, as quais um computador clássico, por mais potente que fosse, conseguisse resolver um problema pertencente a essas classes em tempo hábil. Foi nesse contexto, que Paul Benioff [4] e Richard Feynman [5] propuseram os primeiros modelos de computação quântica, ambos argumentando que a natureza é baseada nas leis da mecânica quântica, evidenciando que simulações de sistemas quânticos deveriam ser realizadas em máquinas que obedeçam as regras do mundo quântico, para que não tenhamos problemas em realizar a computação em tempo hábil ou para que não fosse necessário recorrer a aproximações.

Na década de 90, começaram a surgir os primeiros algoritmos quânticos que demonstravam formas mais eficientes, do ponto de vista computacional, de se realizar algumas tarefas. Alguns exemplos são: os algoritmos de Deutsch-Josza [6], Bernstein-Vazirani [7], Grover [8] e Shor [9], sendo o último o de maior impacto, pois resolve o problema da fatoração de números em fatores primos de maneira mais eficiente, o que gera uma ameaça a alguns sistemas de criptografia utilizados hoje em dia, RSA e ECC, uma vez que se baseiam nesse tipo de problema. Este feito, despertou um maior interesse da academia e da indústria na computação quântica, contudo havia um grande obstáculo no caminho, que era a questão de como lidar com erros durante o processo de computação e transmissão de informação quântica, já que não adiantaria ter uma forma mais eficiente de se fazer as coisas, se não somos capazes de lidar com os erros e garantir a obtenção de resultados corretos. Infelizmente, grande parte da teoria clássica usada na correção de erros não podia ser automaticamente reaproveitada no contexto quântico. Primeiro porque se baseava em medição direta dos bits, buscando obter informações sobre os erros

contudo no contexto quântico isso não é possível, pois o ato de medir causa um colapso da função de onda do sistema, levando a perda de parte da informação codificada na função de onda. Outra complicação, surge da impossibilidade de usar redundância de informação através de clonagens da mesma devido a uma restrição imposta pela mecânica quântica, que é conhecida pelo nome de teorema da não clonagem. Por fim, temos o fato de que os sistemas quânticos estão sujeitos a decoerência, ou seja, perda de seu comportamento quântico, causada por interações com o ambiente, e são mais suscetíveis a novas classes de erros, nos levando a ter um cuidado maior com interações externas e no manuseio dos qubits.

Os desafios mencionados acima, impediam a decolagem da computação quântica até 1995, quando Peter Shor demonstrou que usando emaranhamento entre os qubits, seríamos capazes de realizar redundância da informação codificada, nos permitindo protegê-la de erros coerentes [10]. Isto foi suficiente para retirar o receio de que não teríamos mecanismos para lidar com os erros, permitindo a viabilização de projetos que buscavam o desenvolvimento destas máquinas quânticas. Atualmente, temos diversas iniciativas na academia e no setor privado (IBM, Google, Amazon, Microsoft, Xanadu, Righetti, Honeywell e outras), buscando encontrar o melhor candidato a hardware quântico escalável (do ponto de vista tecnológico e econômico), nessa disputa temos soluções usando qubits supercondutores [11], fotônica [12], armadilha de íons, semicondutores e outras, contudo ainda não há um consenso sobre qual seria a melhor abordagem. Independente de qual destas opções venha a assumir o posto de hardware quântico dominante, devemos ressaltar os diversos avanços obtidos no campo, demonstrações de supremacia quântica [13, 14, 15], solução de problemas de química quântica através do uso de algoritmos quânticos variacionais [16], demonstração de viabilidade de códigos quânticos de correção de erros [17, 18, 19, 20, 21, 22] e outros. Além destes feitos, temos também *roadmaps* de grandes empresas, como IBM e Google, planejando apresentar dispositivos com mais de 1000 qubits nos próximos anos, aliado a desenvolvimento de linguagens de programação (Qiskit, Cirq, Q# e outras), disponibilização de acesso aos dispositivos via nuvem, para que pesquisadores e interessados em soluções quânticas, possam desenvolver seus trabalhos e testá-los em computadores quânticos reais, por fim, projetos visando a disseminação desse conhecimento e a capacitação das futuras gerações.

Apesar dos avanços, ainda existem desafios a serem superados e temas a serem debatidos, pois como é válido ressaltar, o mais provável é que computadores clássicos e quânticos devam coexistir, trabalhando em cooperação, até que o atual estágio de dispositivos NISQ (*Noise Intermediate Scale Quantum*) e sem correção quântica de erros seja superado. Uma vez que ainda não se sabe por completo o quanto a computação quântica expande as classes de problemas que podem ser resolvidos em tempo hábil, lembrando que em alguns casos há desempenho equiparado com uma máquina clássica, espera-se que no futuro próximo, seja comum ver soluções híbridas. Ou seja, máquinas

que utilizam a computação quântica até um certo ponto e transmitindo o resultado para um computador clássico. Por exemplo, em algoritmos variacionais usados no presente momento, onde o *ansatz* é preparado em um computador quântico, mas com a otimização de parâmetros sendo feita por uma solução clássica.

A estrutura do texto é particionada em quatro capítulos onde são discutidos os temas relacionados ao trabalho. No capítulo 1, apresentamos os principais conceitos da teoria clássica de informação e de computação clássica. Nele desenvolvemos os conceitos de entropia de Shannon, as ideias de códigos de correção de erros e como estabelecer um modelo de computação baseada em circuitos lógicos. No capítulo 2, é feita uma revisão da mecânica quântica, onde definimos a base que será usada para desenvolver as ideias de computação quântica, ou seja, sistemas de spin-1/2, matrizes de densidade e estados emaranhados. No capítulo 3, é introduzido modelo de circuitos da computação quântica com a apresentação das principais portas, além disso também são mostradas algumas possíveis implementações dessas portas através de sistemas quânticos e por fim discutimos como o emaranhado é um recurso importante para os campos de informação e computação quântica. No capítulo 4, estudamos alguns tipos de ruídos que podem comprometer a execução de algoritmos quânticos e apresentamos o formalismo de estabilizadores. Este nos permite criar códigos quânticos de correção de erros que são capazes de corrigir alguns dos erros apresentados. No final é realizada uma breve discussão sobre um teste de performance do *surface code* de detecção [[4,1,2]].

1 Teoria de Informação e Computação Clássica

"Information is the resolution of uncertainty."
- Claude Shannon.

"All things physical are information-theoretic in origin."
- John Archibald Wheeler.

1.1 Conceitos básicos da Teoria da Informação

Para descrever o universo, usamos quantidades físicas como massa, velocidade, carga e outras quantidades, têm em comum é que todas elas são mensuráveis. Atualmente, a informação é dada como um conceito e uma quantidade física fundamental no entendimento da natureza, contudo antes de Claude Shannon nos dar uma definição mais precisa e matemática dela, éramos incapazes de quantificá-la e isso fazia com que a víssemos como algo puramente abstrato. De fato a informação é uma quantidade física [23, 24] e não apenas isso, temos evidências de uma forte ligação entre a teoria de informação com a termodinâmica, isso devido às discussões geradas pelo problema do demônio de Maxwell e por estudos sobre calor gerado em processos de computação. Podemos mencionar como exemplo, o artigo de Rolf Landauer de título *"Irreversibility and Heat Generation in the Computing Process"*, onde é demonstrado que processos lógicos irreversíveis possuem relação com processos dissipativos [25]. Alguns anos após a publicação de Landauer, Charles Bennett conseguiu mostrar que havia uma maneira de converter as portas lógicas irreversíveis em portas lógicas reversíveis, fazendo com que os processos físicos por trás dessas operações fossem não dissipativos [26], contudo alguns processos ainda permaneciam dissipativos, como o caso de apagar uma informação. Atualmente a solução mais aceita para o paradoxo do demônio de Maxwell é baseada nessa proposta de Charles Bennett. A discussão sobre física e informação é longa, a ideia aqui foi apenas dar uma breve motivação para o estudo da teoria de informação, que providencia os conceitos que vamos usar como ferramentas ao longo do texto.

De início, devemos apresentar a unidade de medida da informação, ou seja, o bit. A definição de bit é dada como a quantidade de informação necessária para escolher entre duas alternativas equiprováveis, já a origem do termo bit deriva da contração das palavras *binary digit* (dígito binário), que associamos aos caracteres do alfabeto binário $\mathcal{B} = \{0, 1\}$ usados na computação clássica. Da definição de bit, podemos concluir que há uma relação

de dois para um entre alternativas equiprováveis e bits, portanto se considerarmos um cenário em que precisamos escolher entre m alternativas equiprováveis, então o número n de bits necessários será igual a

$$n = \log_2 m. \tag{1.1}$$

No caso em que conhecemos a quantidade de bits e buscamos saber entre quantas alternativas equiprováveis podemos escolher, basta tomar o inverso da equação acima e teremos a resposta.

Um dos problemas em que Shannon estava interessado quando escreveu o artigo de 1948, era da comunicação entre duas partes através de um canal de transmissão ruidoso [2, 27], neste cenário temos um emissor e um destinatário, o primeiro deve colher dados de uma fonte de informação e codificá-los, visando protegê-los de erros, para então realizar o envio. Durante o processo de transmissão, há uma certa probabilidade de que os dados sejam corrompidos devido a existência de ruído no canal de comunicação. Uma vez que a informação foi codificada seguindo um código para correção de erros, existe a possibilidade de recuperar o conteúdo original da mensagem a partir da mensagem corrompida, esta tarefa fica a cargo do destinatário e é executada durante a etapa de decodificação, sendo assim, para que a comunicação seja considerada um sucesso, o destinatário deve conseguir obter a mensagem original após o processo de decodificação. Agora que temos uma descrição do problema a ser explorado, vamos definir a terminologia e os conceitos com maior nível de detalhes.

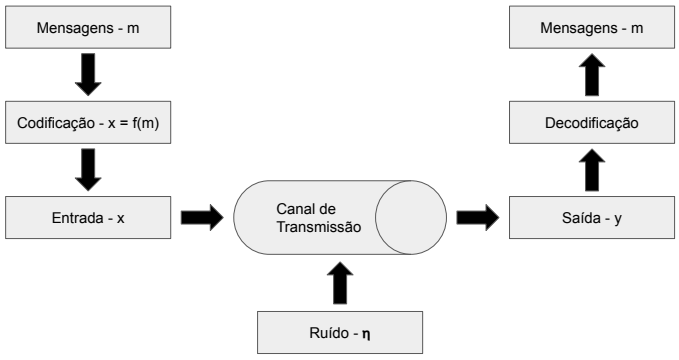


Figura 1 – Representação gráfica de um canal de comunicação.

Nosso ponto de partida é a fonte das mensagens, aqui vamos considerar que uma mensagem m é uma sequência ordenada de k símbolos

$$m = (m_1, \ldots, m_k), \tag{1.2}$$

onde cada símbolo pode representar um número ou uma letra. Os símbolos presentes na mensagem, estão associados a resultados de eventos que ocorrem na fonte de mensagens,

que têm ligação com a variável aleatória M, esta pode assumir qualquer valor de um alfabeto com a símbolos

$$\mathcal{A}_M = \{m_1, \ldots, m_a\}. \tag{1.3}$$

A probabilidade da fonte gerar qualquer um dos símbolos presentes no alfabeto (1.3) é definida pela seguinte distribuição de probabilidade

$$p(M) = \{p(m_1), \ldots, p(m_a)\}, \tag{1.4}$$

que deve satisfazer a condição

$$\sum_{i=1}^{a} p(m_i) = 1. \tag{1.5}$$

A segunda etapa no processo de comunicação descrito anteriormente, consiste na codificação da mensagem original, onde buscamos adicionar redundância para protegê-la contra erros e eventualmente um incremento na eficiência do canal de transmissão. Em termos matemáticos, o que buscamos é uma função f que atuando sobre a mensagem m, nos retorne uma sequência de palavras-código $x = (x_1, \ldots, x_l)$, ou seja, buscamos uma expressão do tipo $x = f(m)$. É válido ressaltar que a palavra-código e a mensagem não precisam necessariamente ter o mesmo tamanho, pois podemos aplicar um processo de compressão na mensagem, a fim de remover redundâncias que estejam presentes, contudo devemos avaliar o tipo de compressão aplicada, pois eventualmente esse processo pode comprometer a recuperação exata da mensagem, uma vez que alguma informação tenha sido descartada durante a compressão. Cada palavra-código tem relação com um valor da variável aleatória X, esta pode adotar qualquer um dos b valores presentes no livro de código

$$\mathcal{C}_X = \{x_1, \ldots, x_b\}. \tag{1.6}$$

A distribuição de probabilidade que nos dá a probabilidade de cada palavra-código é definida como

$$p(X) = \{p(x_1), \ldots, p(x_b)\}. \tag{1.7}$$

Resumindo as definições apresentadas acima, podemos dizer que um código é um mapeamento entre uma lista de símbolos e suas palavras-códigos correspondentes.

Uma vez que o remetente finaliza o processo de codificação, ele precisa dar entrada no canal de transmissão com a mensagem codificada x. Esta será transmitida para o destinatário e será captada na saída do canal de transmissão, o resultado esperado no final desse processo é uma mensagem codificada $y = (y_1, \ldots, y_l)$, de mesmo tamanho da mensagem de entrada x, porém podendo apresentar conteúdo distinto, já que pode ter

sido corrompida por ruídos durante a comunicação. Cada símbolo presente em y é um valor associado a uma variável aleatória Y, que pode adotar qualquer um dos a valores presentes no seguinte alfabeto

$$\mathcal{A}_y = \{y_1, \ldots, y_a\}. \tag{1.8}$$

A probabilidade de ocorrência de cada um dos símbolos presentes no alfabeto acima é dada pela seguinte distribuição de probabilidade

$$p(Y) = \{p(y_1), \ldots, p(y_a)\}. \tag{1.9}$$

As mensagens recebidas y, devem ter alguma relação com as mensagens x que foram transmitidas pelo remetente, para que o processo de decodificação possa ser bem sucedido, caso contrário o resultado final será uma mensagem incorreta. O mapeamento entre as mensagens enviadas e as mensagens obtidas na saída, e vice-versa, é dado por um código. O que chamamos de código é o conjunto de operações formado pela elaboração da mensagem a partir dos dados, codificação e decodificação. É sabido que o ruído presente no canal de transmissão pode nos induzir a erros, então devemos buscar mecanismos para avaliar o desempenho dos códigos em meio a ruídos, nos permitindo melhorar a eficiência de transmissão deles. Uma métrica que pode ser utilizada é a taxa de erros em um código, esta é definida como o número de entradas incorretas associadas com o livro-código divido pelo número de possíveis entradas.

Outro ponto importante a ser avaliado é a capacidade do canal, esta é definida como a quantidade máxima de informação que pode ser transmitida da entrada do canal para a sua saída, sendo medida em termos da quantidade de informação por símbolo. Em um cenário de canal perfeito, ou seja, sem a presença de ruído, podemos dizer que este trabalha em 100% de sua capacidade, mas nos casos em há presença de ruído, parte da capacidade original do canal é tomada pelo ruído fazendo com que sua capacidade de transmissão diminua, o que nos faz buscar por códigos mais eficientes, a fim de não perder eficiência. É válido ressaltar que a capacidade de um canal se difere da taxa com a qual a informação é realmente transmitida pelo canal, pois essa taxa é medida pelo número de bits transmitidos por segundo e depende apenas do código usado para transmitir a mensagem. Outra questão a ser destacada é que a capacidade do canal fornece um limite superior para a taxa de transmissão.

Para formular a definição matemática da informação, Shannon levou em consideração as seguintes propriedades [27]:

- Continuidade: a quantidade de informação associada com um resultado de um evento aumenta ou diminui continuamente de acordo com a mudança na probabilidade de obtenção do resultado.

- Simétrica: a quantidade de informação associada com uma sequência de resultados, não deve depender da ordem de ocorrência deles.
- Valor máximo: a quantidade de informação associada com o conjunto de resultados não pode aumentar se esses resultados já são equiprováveis.
- Aditividade: a informação associada com um conjunto de resultados é obtida pela adição da informação de cada um dos resultados individuais.

Após uma análise das propriedades, ele chegou a conclusão de que a função matemática capaz de satisfazer esses pré-requisitos era a função logaritmo e a partir disso definiu a quantidade que ficou conhecida como informação de Shannon ou também como surpresa

$$h(x) = -\log_2 p(x), \tag{1.10}$$

onde $p(x)$ representa a probabilidade de ocorrência de um certo valor da variável aleatória em observação. A equação (1.10) traça uma relação entre o nível de previsibilidade de um evento e a quantidade de informação que se obtém da observação do evento. Por exemplo, considere que temos duas moedas, a primeira possui faces iguais e a segunda faces distintas. O evento a ser observado é a face obtida após um lançamento. Ao lançarmos a primeira moeda sabendo que ela possui faces iguais o resultado é completamente previsível, e portanto não nos dá nenhuma informação nova sobre o sistema. No caso da segunda moeda, como ela possui faces distintas há um certo grau de incerteza sobre qual será o resultado do lançamento, então se considerarmos que ambas as faces possuem a mesma probabilidade de ocorrência $p = 1/2$, então o que temos é ganhamos um bit de informação ao observarmos o resultado do lançamento da segunda moeda. Na Figura 2 (a), podemos observar como (1.10) se comporta de acordo com a probabilidade de ocorrência de um resultado associado a variável aleatória. Note que quando um resultado de um evento possui probabilidade ocorrência $p(x) = 1$, ele se torna completamente completamente previsível e não nos dá nenhuma informação, já no extremo oposto, ou seja, quando tende a ser improvável, ou seja, $p(x) \to 0$, a quantidade de informação fornecida tende a infinito.

Em geral, estamos mais interessados em conhecer quanta surpresa, em média, está associada com o conjunto dos resultados possíveis, ou seja, desejamos encontrar a surpresa média de uma distribuição de probabilidade associada a uma variável aleatória de interesse. Tendo essa informação em mente, vamos considerar uma variável aleatória X que está associada a uma distribuição de probabilidade $p(X)$, com esses dados em mãos podemos calcular a surpresa média da variável X da seguinte forma

$$H(X) = E[-\log_2(X)], \tag{1.11}$$

onde $E[\dots]$ simboliza o valor esperado e $H(X)$ representa a entropia da distribuição $p(X)$. Caso a variável aleatória em questão seja discreta, a entropia $H(X)$ pode ser escrita como

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x). \tag{1.12}$$

A equação[1] (1.12) é um dos principais resultados da teoria da informação e ficou conhecida como entropia de Shannon[2]. A entropia de Shannon também pode ser aplicada a variáveis aleatórias contínuas, contudo devemos tomar certos cuidados, pois na transição do discreto para o contínuo aparece um termo cujo resultado da integral é infinito, isso é um problema já que não nos fornece nenhuma informação útil, para contornar essa questão utiliza-se o conceito de entropia diferencial [27], onde o termo infinito é ignorado e considera-se apenas o resultado da integral abaixo

$$H(X) = -\int_{-\infty}^{\infty} p(x)\log_2 p(x)dx. \tag{1.13}$$

Em essência, entropia é uma medida de incerteza e nesse contexto possui uma relação interessante com o conceito de informação. Considere que estamos investigando um fenômeno, inicialmente temos uma grande incerteza sobre este, porém eventualmente diminuímos nosso grau de ignorância através de experimentos, então de certo modo essa diminuição na incerteza (entropia) está ligada a um ganho de informação, então dessa forma podemos encarar entropia e informação como dois lados de uma mesma moeda, ou seja, receber uma certa quantidade informação é equivalente a diminuição da mesma quantidade de entropia. No contexto de teoria de informação, podemos interpretar (1.12) como a quantidade de bits necessária para codificar uma determinada informação ou até mesmo como a quantidade de informação necessária para escolher entre $m = 2^{H(X)}$ alternativas equiprováveis.

Antes de finalizarmos essa seção, vamos retornar ao exemplo das moedas para fazer uma discussão sobre a equação (1.12). Considere que queremos observar o grau de incerteza sobre o resultado de um lançamento de uma moeda, para isso vamos precisar de uma variável aleatória discreta X que pode assumir os valores cara (c) ou coroa (k), de acordo com a distribuição de probabilidade $p(X) = \{p(c) = 1 - p, p(k) = p\}$, onde p representa a probabilidade de obter coroa após o lançamento e que $0 \leq p \leq 1$. Uma vez que dispomos da distribuição, estamos aptos a calcular a entropia de Shannon do problema em questão, então se substituirmos os dados em (1.12) chegaremos ao seguinte resultado

$$H(p) = -(1-p)\log_2(1-p) - p\log_2 p, \tag{1.14}$$

que é conhecido como entropia binária de Shannon. Vamos usar o resultado acima para realizar algumas análises, primeiro vamos começar com o caso em que $p = 0$, note que nessa situação temos uma moeda com as duas faces iguais a cara e que por isso o resultado do lançamento é completamente previsível, ou seja, antes mesmo da moeda cair já sabemos

[1] A derivação deste resultado pode ser encontrada nas seguintes referências: apêndice 2 da referência [2] ou no capítulo 10 da referência [28].

[2] O resultado de Shannon é posterior a entropia de von Neumann, este só passou a levar o nome de entropia após conversas entre o próprio von Neumann e Shannon sobre os estudos feitos por Shannon, nessas conversas Shannon teoria consultado von Neumann sobre a possibilidade de usar o termo entropia e ele deu argumentos favoráveis a utilização do termo.

qual face vamos obter, dessa forma não há nenhum grau de incerteza e portanto $H(p) = 0$. O segundo caso a ser analisado é o de $p = 1$, este é similar ao anterior, a diferença é que a moeda possui as duas faces iguais a coroa, mas isso não impede que o resultado seja o mesmo, ou seja, $H(p) = 0$. O terceiro caso de interesse é quando $0 \leq p \leq 1$ e $p \neq 1/2$, aqui temos moedas com cara e coroa, mas existe algum grau de viés, fazendo com que o resultado possua um grau de imprevisibilidade, o que o faz esse cenário ter $0 < H(p) < 1$. Por fim, temos o cenário em que $p = 1/2$, o que torna a distribuição $p(X)$ uniforme, fazendo com que não sejamos capazes de prever o resultado a ser obtido, ou seja, total ignorância. Isso reflete na entropia assumindo o seu valor máximo $H(p) = 1$. Esta análise pode ser resumida ao gráfico presente na Figura 2 (b), nela podemos constatar todos os comportamentos observados caso a caso.

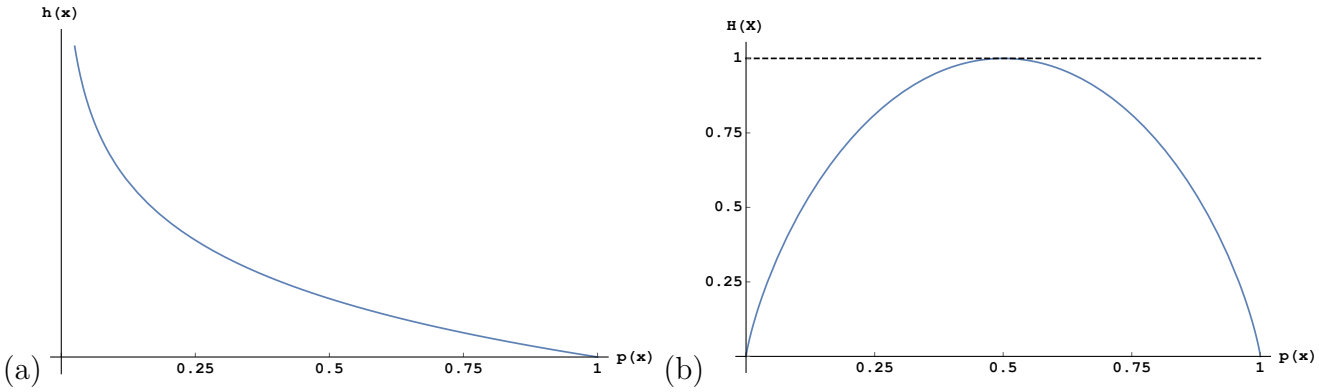


Figura 2 – (a) Comportamento da surpresa de acordo com a previsibilidade do resultado do evento. (b) Entropia Binária de Shannon.

1.2 Codificação e Canais de Comunicação

Na seção anterior, discutimos em detalhes todos os aspectos relacionados ao problema de uma comunicação através de um canal, agora com esses recursos em mãos, vamos dar a definição matemática de características do canal, para que possamos avaliar sua eficiência e outros pontos de interesse. De início devemos nos fazer a seguinte pergunta: um sinal pode ser transmitido de forma eficiente através de um canal de comunicação? A resposta para esta questão é sim, porém o sinal deve possuir as seguintes características: ser transformado em um sinal com valores independentes e se os valores do sinal transformado tiverem uma distribuição que seja otimizada para o canal em questão [27]. Sendo assim, precisamos de definições matemáticas para as características do canal, de forma a garantir que tenhamos recursos para realizar a codificação adequada do sinal.

A característica principal de um canal de comunicação é a sua capacidade C, que nos indica o número máximo de bits que podemos transmitir. Em geral, mede-se essa quantidade em bits por segundo ou bits por símbolo, sua definição matemática se dá

em termos da distribuição de probabilidade $p(X)$ produzida no processo de codificação e da entropia de Shannon $H(X)$, que nos informa a quantidade de informação que o sinal carrega. Diante disso, temos que a capacidade C de um canal sem ruído é definida como a entropia máxima produzida por uma distribuição de probabilidade particular e pode ser escrita como

$$C = \max_{p(X)} H(X). \tag{1.15}$$

A quantidade máxima de símbolos por segundo que pode ser comunicada é definida como

$$R_{max} = \frac{C}{H(X)}. \tag{1.16}$$

Em um cenário ideal a capacidade do canal C é igual a taxa de transmissão R, mas em geral isso nem sempre é verdade, pois o processo de codificação da mensagem pode produzir uma distribuição de probabilidade que não maximize a entropia, levando a um cenário em que $R < C$. Contudo o teorema da codificação de Shannon [2, 27] garante que para qualquer mensagem existe uma codificação tal que o sinal de entrada do canal com C dígitos binários possa ser transmitido, em média, a uma taxa próxima de C bits por unidade de tempo. Note que o teorema de Shannon não nos dá uma fórmula para obter tal codificação, ele apenas afirma que há um código com essa característica, sendo assim, devemos buscar mecanismos para medir o quão próximo ou longe estamos de encontrar essa codificação, para essa tarefa devemos usar a eficiência da codificação, que pode ser calculada usando a fórmula

$$\tau = \frac{H(S)}{L(X)}, \tag{1.17}$$

onde $H(S)$ representa a entropia da fonte de sinal e $L(X)$ o número médio de bits em cada palavra-código. Sendo assim, uma codificação é dada como eficiente quando a sua eficiência τ é igual ou muito próxima de R_{max}. Um exemplo onde isso ocorre é na codificação de Huffman [27].

Parte do que foi construído até esse momento nos restringe a canais sem ruído, contudo estes são apenas idealizações que servem como um norte, nas situações reais temos que lidar com o ruído. Para entender como podemos superar os problemas colocados pelo ruído, será necessário descrever a relação entre as variáveis aleatórias X e Y, representando respectivamente o sinal de entrada e o sinal de saída do canal. Para isso devemos utilizar distribuições conjuntas de probabilidade, que são escritas como matrizes $n \times n$ da seguinte forma

$$p(X, Y) = \sum_{i,j}^{n} p(x_i, y_j) a_{ij}, \tag{1.18}$$

onde $p(x_i, y_j)$ é a probabilidade associada a x_i e y_j, já a_{ij} representa o elemento de matriz que obedece a condição abaixo

$$a_{ij} = \begin{cases} 0, & \text{se } p(x_i, y_j) = 0 \\ 1, & \text{se } p(x_i, y_j) \neq 0 \end{cases}. \tag{1.19}$$

Uma vez que dispomos da distribuição conjunta de probabilidade, podemos determinar sua entropia através do cálculo da surpresa média, então se fizermos essa conta devemos obter a expressão

$$H(X,Y) = -\sum_{i,j} p(x_i, y_j) \log_2 p(x_i, y_j), \tag{1.20}$$

onde $H(X,Y)$ é a entropia conjunta, esta contabiliza a quantidade média de informação de Shannon fornecida pela distribuição $p(X,Y)$. O resultado acima é restrito a variáveis aleatórias discretas, entretanto podemos estendê-lo para variáveis contínuas trocando a soma por uma integral e lembrando que devemos usar o conceito de entropia diferencial para evitar problemas com infinitos.

A quantidade estatística que nos fornecerá informações sobre as relações entre as variáveis aleatórias X e Y é a informação mútua, mas para obtê-la precisaremos obter as distribuições de X e Y a partir da distribuição conjunta $p(X,Y)$. Para realizar essa tarefa é necessário marginalizar $p(X,Y)$, ou seja, somar cada valor da distribuição conjunta sobre todos os possíveis valores de uma das variáveis aleatórias, ao final desse processo devemos obter

$$p(X) = \sum_j p(X, y_j) \tag{1.21}$$

$$p(Y) = \sum_i p(x_i, Y), \tag{1.22}$$

onde $p(X)$ e $p(Y)$ representam as distribuições marginalizadas. Sendo assim, se as variáveis aleatórias X e Y são estatisticamente independentes, a seguinte igualdade será verdadeira

$$p(X,Y) = p(X)p(Y), \tag{1.23}$$

o que implica que a entropia conjunta é a soma das entropias de cada uma das variáveis, ou seja,

$$H(X,Y) = H(X) + H(Y). \tag{1.24}$$

Note que o resultado acima pode ser obtido supondo (1.23) em (1.20) e realizando algumas manipulações. Em geral, a relação entre a entropia conjunta e as entropias das variáveis X e Y é dada pela desigualdade

$$H(X,Y) \leq H(X) + H(Y). \tag{1.25}$$

A informação mútua entre duas variáveis aleatórias discretas X e Y, representa a média de informação que ganhamos sobre Y dado que observamos um único valor de X, o contrário também é verdade já que essa quantidade estatística é simétrica em relação às variáveis. Sabendo disso, podemos entender que a chave para lidar com erros está em

conhecer o quanto da entropia do sinal de saída nos diz sobre o sinal de entrada e quanto é apenas ruído. Diante disso, se queremos assegurar que possamos comunicar a maior quantidade de informação possível através de um canal ruidoso, devemos ter os seguintes ingredientes: alta entropia do sinal de entrada, alta entropia do sinal de saída e baixa entropia do ruído. Os dois primeiros ingredientes implicam que devemos ter uma alta informação mútua, diante disso, devemos quantificá-la matematicamente e isso pode ser feito através da expressão [27]

$$I(X,Y) = \sum_{i,j} p(x_i, y_j) \log_2 \left[\frac{p(x_i, y_j)}{p(x_i)p(y_j)} \right]. \tag{1.26}$$

Note que a equação acima pode ser escrita em termos da entropia conjunta e das entropias das variáveis X e Y, para isso é necessário usar da marginalização e das propriedades da função logaritmo. O resultado final das manipulações descritas deve ser igual a

$$I(X,Y) = H(X) + H(Y) - H(X,Y). \tag{1.27}$$

Há um modo alternativo de se encarar a informação mútua considerando a entropia do sinal de saída em relação ao ruído do canal. Nesta abordagem levamos em conta que se desconhecemos o sinal de entrada X, então nossa incerteza acerca de Y é dada pela entropia $H(Y)$. Mas, se conhecemos X então nossa incerteza sobre o valor de Y é reduzida por uma quantidade $H(Y|X)$ que é conhecida como entropia condicional. Para obter a expressão de $I(X,Y)$ em termos de $H(Y)$ e $H(Y|X)$ é preciso substituir a relação

$$p(y|x) = \frac{p(x,y)}{p(x)}, \tag{1.28}$$

na equação (1.26). A partir disso, com algumas manipulações utilizando as propriedades da função logaritmo e a marginalização, chegamos ao resultado desejado

$$I(X,Y) = H(Y) - H(Y|X). \tag{1.29}$$

A entropia condicional é utilizada para modelar o ruído no canal de comunicação. Para isso vamos considerar que o sinal de saída Y seja igual a

$$Y = X + \eta, \tag{1.30}$$

onde X representa o sinal de entrada e η o ruído presente no canal. Tomando essa suposição na equação (1.29) e desenvolvendo o cálculo devemos obter

$$\begin{aligned} I(X,Y) &= H(Y) - H([X+\eta]|X) && (1.31)\\ &= H(Y) - H(\eta|X);\ H(\eta|X) = H(\eta) && (1.32)\\ &= H(Y) - H(\eta). && (1.33) \end{aligned}$$

Note que $H(X|X) = 0$ dado que se conhecemos o valor de X nossa incerteza sobre X deve ser nula. Além disso assumimos que o valor do ruído η é independente do valor de X,

isso faz com que possamos considerar $H(\eta|X) = H(\eta)$. Comparando o resultado acima com a equação (1.29), vemos que podemos encarar a entropia condicional $H(Y|X)$ como o ruído presente no canal.

É possível resumir a relação entre todas as quantidades aqui discutidas através da expressão

$$H(X,Y) = I(X,Y) + H(X|Y) + H(Y|X), \tag{1.34}$$

que é facilmente visualizada através do diagrama de Venn da Figura 3. Das relações exploradas nesta seção, podemos dar uma nova definição para a capacidade de um canal, buscando adequar ao novo cenário onde lidamos com canais sujeitos a ruído. Para fazer isso, devemos usar (1.29) no lugar de $H(X)$ na fórmula (1.15)

$$C = \max_{p(X)} I(X,Y) = \max_{p(X)} [H(X) - H(X|Y)], \tag{1.35}$$

com isso temos que buscar a distribuição de probabilidade do sinal de entrada que faça $I(X,Y)$ ser máxima. Note também que quando $H(X|Y) = 0$ a expressão acima retorna ao resultado da capacidade de um canal sem ruído. Nesse novo contexto, podemos calcular também a eficiência de transmissão do canal, essa quantidade é expressa como a razão

$$\frac{I(X,Y)}{H(Y)}, \tag{1.36}$$

onde seu resultado nos revela o quanto da entropia do sinal de saída depende da entropia do sinal de entrada, além disso podemos ter uma dimensão do impacto do ruído na transmissão.

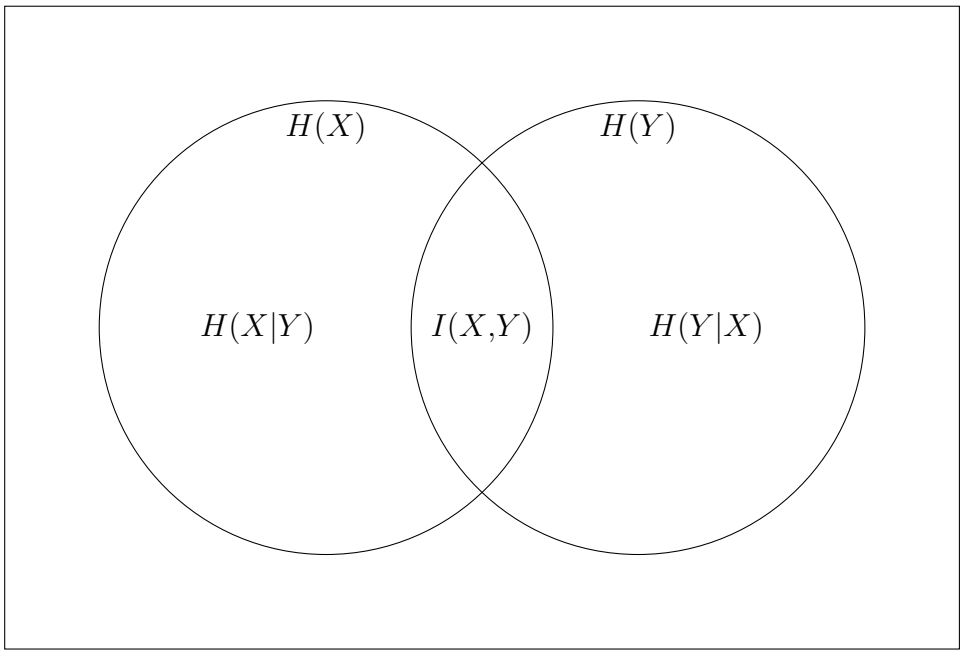


Figura 3 – Representação em diagrama de Venn da equação (1.34).

Por fim, vimos que a entropia pode ser interpretada como a quantidade de informação necessária para distinguir entre alternativas equiprováveis, levando esse fato em conta, é possível dar uma interpretação para as entropias condicionais que nos permite entender melhor a relação entre ruído e *cross-talk* entre os sinais de entrada e saída. Cada mensagem de entrada x_i pode resultar em várias diferentes mensagens de saída $(y_1, \ldots, y_{m_{y|x}})$, e cada mensagem de saída y_j pode ser resultado de diferentes mensagens de entrada $(x_1, \ldots, x_{m_{x|y}})$, para encontrar o número de mensagens de entrada relacionadas a uma de saída y, $m_{x|y}$, e o número de mensagens de saída relacionadas a uma de entrada $m_{y|x}$, devemos usar as expressões

$$m_{y|x} = 2^{H(Y|X)} \text{ e } m_{x|y} = 2^{H(X|Y)}. \tag{1.37}$$

Em um cenário ideal teríamos um mapeamento um a um, mas na presença de ruído a entropia do sinal de saída do canal é maior que a entropia do sinal de entrada, o que nos leva a um certo grau de degenerescência, podendo impedir que a mensagem original possa ser recuperada corretamente, então para que possamos lidar com esse tipo de problema é necessário explorar a ideia de criar mecanismos de redundância da mensagem através de códigos de correção de erros.

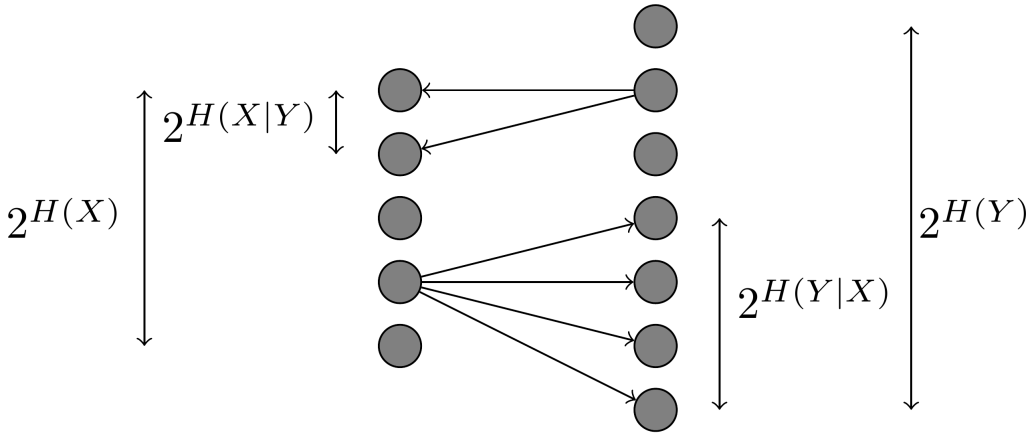


Figura 4 – Relação entre os sinais de entrada e os sinais de saída em um canal de transmissão com ruído.

1.3 Introdução a Teoria Clássica de Correção de Erros

Na seção anterior, chegamos a conclusão que para lidar com erros na mensagem, devido a presença de ruído no canal de transmissão, é preciso criar um código de correção de erros. Sendo assim, para que comunicações sejam bem sucedidas e que computadores possam produzir resultados corretos, precisamos entender os principais aspectos por trás desses códigos, como criá-los e como torná-los mais eficientes. Posteriormente, vamos ver que algumas das ideias desenvolvidas no contexto clássico não podem ser automaticamente

reaproveitadas no contexto quântico, tornando necessária a adaptação destas ao novo paradigma.

Para introduzir os conceitos básicos de correção de erros vamos considerar a seguinte situação: suponha que dois indivíduos, Alice e Bob, necessitam se comunicar para confirmarem o horário de uma reunião de trabalho, então no dia anterior à reunião Alice envia uma mensagem a Bob perguntando se ele irá comparecer a reunião, Bob deve responder apenas "Sim" (0) ou "Não" (1), ao receber a mensagem de Alice, Bob avalia que terá disponibilidade para participar da reunião e envia como resposta "Sim" (0). Contudo, o canal de comunicação que ambos estão utilizando apresenta ruído, este pode fazer com que o bit que representa a resposta de Bob possua uma probabilidade p de ter seu valor alterado, o que resultaria em Alice receber a resposta errada, o que causaria problemas, já que Bob iria aparecer para a reunião, mas Alice cancelaria o compromisso, uma vez que ela recebeu a mensagem incorreta.

A situação descrita no parágrafo anterior representa o de canal binário simétrico, cuja representação gráfica pode ser vista na Figura 5. A relação entre as probabilidades de ocorrência de uma das mensagens na entrada e na saída do canal é dada pela expressão [1]

$$P(Y=y)=\sum_{x\in X}P(Y=y|X=x)P(X=x), \tag{1.38}$$

onde as probabilidades condicionais $P(Y=y|X=x)$ são entendidas como probabilidades de transição de estados, estas podem ser escritas como uma matriz M

$$M=\begin{bmatrix} P(Y=0|X=0) & P(Y=0|X=1) \\ P(Y=1|X=0) & P(Y=1|X=1) \end{bmatrix}=\begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix}, \tag{1.39}$$

que pode ser usada para reescrever a relação entre as probabilidades como a equação matricial

$$\begin{bmatrix} P(Y=0) \\ P(Y=1) \end{bmatrix}=\begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix}\begin{bmatrix} P(X=0) \\ P(X=1) \end{bmatrix}. \tag{1.40}$$

Olhando para a matriz M fica evidente a razão de o adjetivo simétrico aparecer no nome do canal, o binário também é bem óbvio, uma vez que dispomos de apenas duas alternativas de mensagens. Entretanto, essa matriz nos dá mais informações sobre o canal em questão, para acessá-las devemos olhar para os seguintes casos de interesse

$$\text{Caso 1: } p=0 \;\Rightarrow\; M_1=\mathbb{I}=\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{1.41}$$

$$\text{Caso 2: } p=1 \;\Rightarrow\; M_2=\sigma_x=\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{1.42}$$

$$\text{Caso 3: } p=\frac{1}{2} \;\Rightarrow\; M_3=\frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}. \tag{1.43}$$

Note que o caso 1 é trivial, pois a matriz M fica igual a identidade e portanto o canal preserva as mensagens, já o caso 2 mostra a situação em que a mensagem de saída possui o valor oposto da mensagem de entrada e o terceiro caso nos revela o resultado mais interessante, nele a matriz M não apresenta inversa pois $\det(M) = 0$, uma vez que M possui duas linhas idênticas, este fato nos faz perder a informação inicial e produz um resultado completamente aleatório.

Outra forma maneira de visualizar o impacto do ruído no canal é através da determinação de sua capacidade, no caso do canal binário simétrico temos o seguinte resultado para a capacidade [29]

$$\begin{aligned} C &= \max_{p(X)} I(X,Y) \\ &= \max_{p(X)} [H(Y) - H(Y|X)];\ H(Y|X) = H(p) \\ &= 1 - H(p) \\ &= 1 + p\log_2 p + (1-p)\log_2(1-p). \end{aligned} \tag{1.44}$$

Note que a expressão acima nos indica que nos casos 1 e 2 o canal apresenta capacidade máxima, uma vez que a entropia do ruído seria nula, entretanto no caso 3 a capacidade é reduzida a zero porque o ruído nos impede de ter qualquer informação sobre a mensagem inicial. Este comportamento da capacidade do canal binário simétrico pode ser visto na Figura 6.

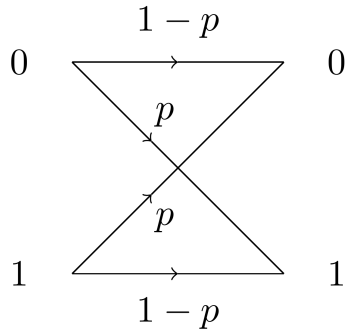


Figura 5 – Canal binário simétrico com probabilidade p de ocorrência de um *bit-flip*. À esquerda temos a entrada do canal com as mensagens enviadas e à direita temos as mensagens recebidas no final do canal.

O erro presente no canal binário simétrico é conhecido como *bit-flip*, pois troca 0 por 1 e vice-versa, uma estratégia para proteger nossas mensagens dessa classe de erro é a adição de redundância, ou seja, enviar mais de uma vez a mensagem. Entretanto aqui surge a seguinte questão: a quantidade de repetições da mensagem pode ser qualquer uma? A resposta para essa pergunta é não, pois se o número for par não é possível formar um consenso sobre qual era a mensagem original que o remetente de fato enviou. Então, para evitar esse problema adotamos um número ímpar de repetições da mensagem para formar um esquema de decodificação chamado de voto da maioria, onde declara-se como

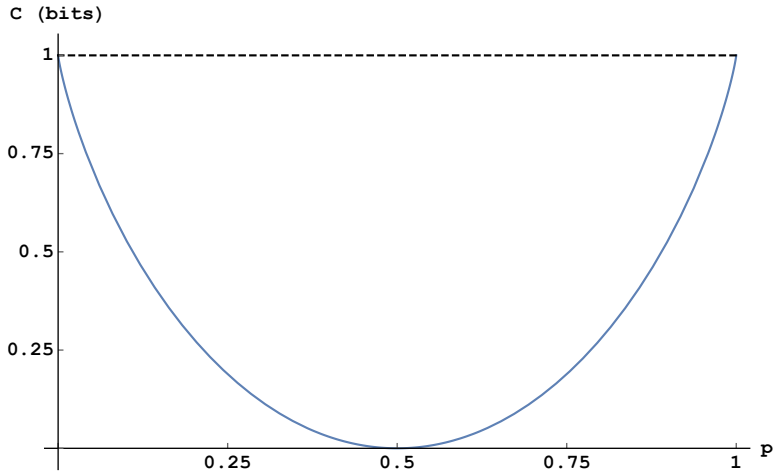


Figura 6 – Capacidade do canal binário simétrico.

mensagem final a que tiver maior frequência na *bitstring*. Considere que a mensagem enviada foi 000, mas que no processo de transmissão o segundo bit mudou de valor fazendo com que a mensagem passasse a ser 010. Então, durante o processo de decodificação o destinatário usa o esquema do voto da maioria e contabiliza que o 0 aparece duas vezes na mensagem, enquanto 1 aparece apenas uma vez, sendo assim ele deve assumir que a mensagem original seja 0. Note que o voto da maioria falha se ocorrer mais de um erro. Para visualizar este fato vamos considerar novamente a mensagem original como 000, porém agora vamos assumir que durante a transmissão ocorreram dois erros, um no primeiro bit e outro no último, nos levando a mensagem corrompida 101, então ao realizarmos a decodificação dessa mensagem iremos notar que o bit 1 aparece duas vezes, enquanto o 0 apenas uma. Então, pelo voto da maioria assumimos que a mensagem original deve ser 1, mas este resultado na verdade é errado e compromete a comunicação. Este exemplo deixa explícita a limitação do esquema do voto da maioria e nos obriga a buscar formas mais robustas de decodificar as mensagens.

O processo descrito acima define o código de repetição de 3 bits, este apresenta as seguintes *bitstrings* como palavras-código

$$0_L \ \rightarrow \ 000 \text{ e } 1_L \ \rightarrow \ 111, \tag{1.45}$$

onde 0_L e 1_L são chamados de bits lógicos e são a base do espaço do código. A matriz geradora G do código de repetição de 3 bits é escrita como

$$G = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}, \tag{1.46}$$

onde sua atuação sobre a matriz m da mensagem, produz a matriz linha x da palavra-código

$$x = mG, \tag{1.47}$$

neste caso m tem tamanho 1×1 e x é uma matriz linha 1×3. Note que se $m = 0$ ou $m = 1$, teremos como resultado (1.45) e qualquer vetor linha x que seja diferente de (1.45)

é considerado uma mensagem com erro. A estrutura do código de repetição de 3 bits é baseada em usar 3 bits para codificar 1 bit de informação com a capacidade de proteger esse 1 bit codificado de um erro, isso se deve a relação entre a distância de código e o número de erros

$$d = 2t + 1, \tag{1.48}$$

onde d representa a distância de código e t a quantidade de erros. Para o código em questão, temos $t = 1$, o que implica em $d = 3$, sendo assim podemos descrever o código de repetição de 3 bits usando a seguinte notação para códigos de correção de erros

$$[n, k, d], \tag{1.49}$$

onde n representa a quantidade de bits usados para a codificação, k é a quantidade de bits codificados e d a distância do código, dessa forma podemos representar o código em questão como $[3, 1, 3]$. Note que (1.48) implica que um código com distância d tem a capacidade de corrigir $(d-1)/2$ erros e detectar $d/2$ erros [29], sendo assim para que um código tenha a capacidade de corrigir erros ele deve ter uma distância $d \geq 3$, abaixo disso temos o cenário em que ele é capaz apenas de detectar erros e outro em que não consegue nem corrigir e nem detectar. A forma mais precisa de definir a distância do código é através da distância de Hamming, antes de definí-la precisamos entender os conceitos de peso e distância de Hamming. O peso de Hamming, denotado por $\mathrm{wt}(u)$, é definido como a quantidade de dígitos diferentes de zero em um vetor $u = u_1 \dots u_n$. Exemplos:

$$\begin{aligned} u = 101 &\Rightarrow \mathrm{wt}(101) = 2 \\ u = 000 &\Rightarrow \mathrm{wt}(000) = 0 \\ u = 001 &\Rightarrow \mathrm{wt}(001) = 1. \end{aligned}$$

A distância de Hamming, denotada por $\mathrm{dist}(u, v)$, é definida como o número de caracteres distintos entre duas *bitstrings* representadas por dois vetores $u = u_1 \dots u_n$ e $v = v_1 \dots v_n$. Exemplos:

$$\begin{aligned} u = 101 \text{ e } v = 011 &\Rightarrow \mathrm{dist}(101, 011) = 2 \\ u = 000 \text{ e } v = 111 &\Rightarrow \mathrm{dist}(000, 111) = 3. \end{aligned}$$

Note que através do peso de Hamming podemos calcular a distância de Hamming usando a seguinte igualdade

$$\mathrm{dist}(u, v) = \mathrm{wt}(u - v), \tag{1.50}$$

aqui é possível ver que ambos os lados da equação acima expressam a quantidade de caracteres distintos entre duas *bitstrings*. Conhecendo esses conceitos temos condições de definir a distância do código, para este fim vamos considerar que u e v representem

quaisquer palavras-código presentes no código $\mathcal{C}$, sendo assim nosso interesse é encontrar a menor distância possível entre as palavras-código. Para isso devemos calcular

$$\begin{aligned} d &= \min \text{dist}(u, v) \\ &= \min \text{wt}(u - v), \end{aligned} \tag{1.51}$$

onde d representa a distância mínima de Hamming do código, que por sua vez é definida como a distância do código. Para encontrar a distância mínima do código não é necessário comparar todos os pares de palavras códigos, uma vez que u e v sejam palavras-código de um código linear $\mathcal{C}$, então $u - v = w$ também é uma palavra-código, sendo assim da expressão da distância mínima temos que

$$d = \min_{w \in \mathcal{C}, w \neq 0} \text{wt}(w). \tag{1.52}$$

Uma mensagem após passar pelo processo de codificação passar a ter bits de paridade, além de seu conteúdo, para que seja possível verificar se houve algum erro durante o processo de transmissão. A fim de verificar os bits de paridade, existe uma matriz binária H que é conhecida como matriz de paridade, esta nos ajuda a definir que um código é linear com uma matriz de paridade quando todos os seus vetores x satisfazem a seguinte equação em módulo 2[3]

$$Hx^T = 0. \tag{1.53}$$

É possível obter as palavras-código a partir da equação acima, para isso é preciso resolver o sistema de equações, mas essa tarefa pode ser muito trabalhosa já que depende dos tamanhos das matrizes envolvidas. Determinar as palavras-código é importante para o problema de determinação da distância mínima do código, este é um problema que pertence a classe de complexidade NP-hard [30] e que eventualmente poderíamos tentar buscar um algoritmo quântico que fosse capaz de resolver esse problema de forma mais eficiente. No caso do código de repetição de 3 bits considerado anteriormente, temos a seguinte matriz de paridade

$$H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}. \tag{1.54}$$

Note que as palavras-código 000 e 111 satisfazem a equação (1.53) para a matriz apresentada acima. A partir de (1.47) e (1.53), podemos derivar um resultado que mostra uma relação entre a matriz geradora do código G e a matriz de paridade H, então partindo de (1.47)

[3] Para maiores detalhes sobre aritmética binária o leitor deve consultar o apêndice A.

temos

$$\begin{aligned} & x = mG;\ (AB)^T = B^T A^T \\ & x^T = G^T m^T \\ & Hx^T = HG^T m^T;\ Hx^T = 0 \\ & HG^T m^T = 0 \\ & HG^T = 0 \text{ ou } GH^T = 0. \end{aligned} \tag{1.55}$$

A relação acima é importante para verificar se de fato as matrizes G e H definem um código. Uma característica de códigos lineares é que eles podem apresentar diferentes matrizes geradoras e matrizes de paridade, em geral estas matrizes são apresentadas nas formas abaixo

$$H = [A|\mathbb{I}_{n-k}] \tag{1.56}$$

$$G = \left[\mathbb{I}_k| - A^T\right], \tag{1.57}$$

onde A é uma matriz que define os bits de paridade. Outra característica dos códigos lineares é que vetores do tipo $x' = x_1 + x_2$ e $x'' = cx_1$ satisfazem (1.53).

Podemos usar a linearidade de um código para modelar que o vetor que representa a mensagem na saída do canal de transmissão ruidoso apresente a seguinte estrutura

$$y = x + e, \tag{1.58}$$

onde x representa o vetor da palavra-código e e o vetor do erro adicionado a mensagem. Usando o vetor acima na equação (1.53) obtemos um vetor S conhecido como síndrome, que nos permite mapear os erros de forma a garantir que seja possível corrigí-los. Sendo assim, a atuação de H sobre y^T produz o seguinte resultado

$$S = Hy^T = \underbrace{Hx^T}_{=0} + He^T = He^T. \tag{1.59}$$

Note que a equação acima só resulta em um vetor nulo quando y é igual a uma palavra-código, caso contrário temos um vetor relacionado a um erro em uma dada posição da mensagem. Na Tabela 1, temos todas as possibilidades de mensagens com e sem erros para o código de repetição. Nela podemos ver que de fato que a síndrome das palavras-código é um vetor nulo, já para as mensagens corrompidas vemos que algumas apresentam síndromes iguais, o que confere um grau de degenerescência ao código, gerando complicações no processo de decodificação. Entretanto, podemos usar modelos estatísticos para nos ajudar na tarefa de decodificação de mensagens com erros que levam a mesma síndrome, este dilema pode ser resolvido determinando qual erro é o menos provável. A probabilidade de erro, P_{erro}, pode ser definida como a probabilidade de o resultado da codificação ser

diferente da mensagem original. Para fazer esse tipo de análise podemos fazer uso do seguinte modelo [29]

$$P_{erro} = \sum_i \binom{n}{i} p^i (1-p)^{n-i}, \tag{1.60}$$

onde P_{erro} pode ser caracterizada por uma distribuição binomial. Note que se usarmos o modelo acima para o código de repetição de bits 3 e considerarmos que a mensagem original era 0, então um erro irá ocorrer apenas se 2 ou 3 bits trocarem de valor, pois pelo o voto da maioria o decodificador interpretaria a mensagem como sendo 1. A probabilidade de erro no cenário descrito é igual a

$$P_{erro} = 3p^2(1-p) + p^3, \tag{1.61}$$

onde p representa a probabilidade de acontecer um *bit-flip*. Note que a probabilidade de obtermos o resultado correto ao final do processo de decodificação é igual a $1-p$, portanto para que nossa transmissão seja bem sucedida é preciso garantir que $P_{erro} < p$ e isso ocorre quando $p < 1/2$. Na Figura 8, podemos verificar que a partir de $p = 1/2$ o processo de decodificação passa a apresentar falhas.

Bitstring (y)	Síndrome (S^T)
000	[0,0]
001	[0,1]
010	[1,0]
100	[1,1]
101	[1,0]
110	[0,1]
111	[0,0]

Tabela 1 – Mensagens que podem chegar ao destinatário e suas respectivas síndromes.

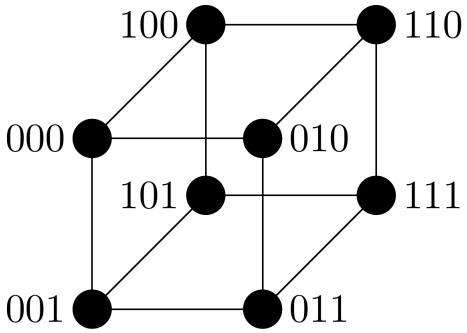


Figura 7 – Representação gráfica do código de repetição com distância $d = 3$.

Outro ponto de interesse em um código é sua eficiência de transmissão, observe que a *bitstring* do vetor x produzido através da aplicação do gerador G sobre a mensagem

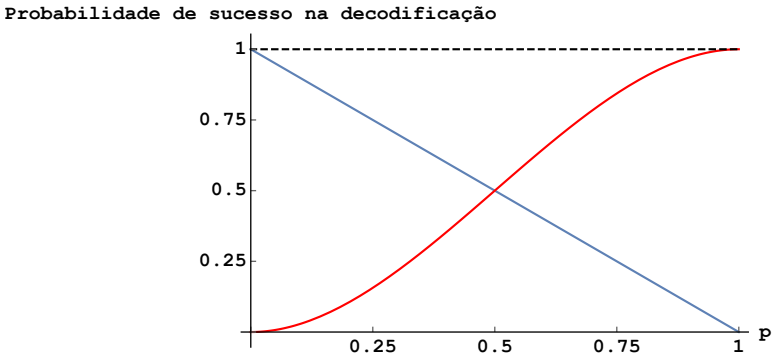


Figura 8 – Probabilidade de não ocorrer erro e de ocorrer erro em função da probabilidade de ocorrência de *bit-flips*.

m possui a seguinte estrutura

$$x_{[3,1,3]} = \underbrace{x_1}_{\text{mensagem original}} \quad \underbrace{x_2 x_3}_{\text{cópias}}. \tag{1.62}$$

Isso nos mostra que estamos usando 1 bit para a mensagem original e 2 para a redundância, analisando a razão k/n é possível ver que a eficiência de transmissão do código de repetição de 3 bits é de aproximadamente $33,3\%$. Este fato nos mostra que apesar deste código conseguir proteger nossa mensagem ele não é muito eficiente no quesito de transmissão, portanto devemos buscar por códigos que sejam mais eficientes na transmissão mantendo ou aumentando a quantidade de erros corrigidos. Um exemplo de classe de código com maior eficiência de transmissão se comparado ao código de repetição, é a dos códigos de Hamming. Estes são conhecidos por serem fáceis de codificar e decodificar. Os códigos de Hamming são fáceis de decodificar, pois possuem a característica peculiar de que as coordenadas dos vetores das síndromes são iguais ao número em binário que corresponde a posição do erro na mensagem. Além disso, são códigos cujo tamanho depende do número de bits de paridade r e apresentam distância fixa igual a $d = 3$. A forma geral dos códigos de Hamming é expressa por

$$[2^r - 1, 2^r - 1 - r, d = 3];\ r \geq 2. \tag{1.63}$$

Note que para $r = 2$ temos o código de repetição $[3, 1, 3]$. Para $r = 3$ temos um código $[7, 4, 3]$ que apresenta uma eficiência de transmissão de aproximadamente $57,1\%$, cuja estrutura das palavras-código segue o formato

$$x_{[7,4,3]} = \underbrace{x_1 x_2 x_3 x_4}_{\text{mensagem original}} \quad \underbrace{x_5 x_6 x_7}_{\text{bits de paridade}}, \tag{1.64}$$

sendo estas geradas pela matriz

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{1.65}$$

A matriz de paridade do código de Hamming $[7, 4, 3]$ é dada por

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}, \tag{1.66}$$

note que a matriz apresentada acima possui a seguinte característica particular: suas colunas são os números de 1 a 7 em binário escritos em forma crescente.

Para nos guiar na busca de bons códigos com alta eficiência de transmissão, temos o limite de Gilbert-Varshamov [1, 29]. Este estabelece que para um número grande de bits n, deve existir um código de correção de erros $[n, k]$ que protege k bits lógicos contra erros em t bits físicos, tal que sua eficiência seja igual a

$$\frac{k}{n} \geq 1 - H\left(\frac{2t}{n}\right), \tag{1.67}$$

Apesar deste resultado não nos dar o código em si, ele é importante, pois assim como o teorema da codificação de Shannon, nos diz que os códigos existem, portanto podemos buscar por eles sem ter medo de ao final não encontrar nenhum.

As ideias apresentadas nesta seção posteriormente serão adaptadas para o contexto da teoria quântica de correção de erros, veremos também que algumas classes de códigos clássicos possuem análogos quânticos, como é o caso da relação entre os códigos de Hamming e os códigos CSS [1, 31].

1.4 Computação Clássica

As principais ideias que fundamentam a teoria de algoritmos e de computação foram formuladas a partir do ano 1930 por estudiosos como Alonzo Church, Alan Turing e outros. No início do século XX, o matemático David Hilbert havia levantado a seguinte questão: existe ou não algum algoritmo que poderia ser usado para solucionar todos os problemas da matemática? Surpreendentemente, a resposta para essa pergunta é não! Para provar esse resultado, Church e Turing tiveram que ir a fundo no problema de encontrar a definição precisa de algoritmo na matemática, esta busca culminou nas fundações da teoria moderna de algoritmos e da ciência da computação [1].

Entendemos como algoritmo, uma receita precisa para a realização de uma tarefa, por exemplo o processo de preparação de um bolo, neste caso temos ingredientes bem definidos que serão usados para executar a sequência de instruções necessárias para que ao final do processo tenhamos como resultado um bolo. O modelo computacional conhecido como máquina de Turing ajudou na construção da definição de algoritmo dada no início deste parágrafo, essa máquina possui quatro elementos principais: (1) um registrador finito de estado, parte responsável por guardar o estado da máquina de Turing; (2) um

programa, parte responsável pelas instruções que serão executadas pelo instrumento de leitura e gravação; (3) uma fita, parte onde são gravadas as instruções dada pelo programa; (4) instrumento de leitura e gravação, equipamento que permite ler e alterar o conteúdo presente na fita de acordo com as instruções passadas pelo programa. O registrador de estados pode ser pensado como uma espécie de microprocessador que coordena a operação, além disso ele armazena temporariamente os vários estados que a máquina pode assumir, dentro desse conjunto de estados existem dois elementos especiais chamados de estado de início q_s e o estado de parada q_h, responsáveis por indicar quando a computação se inicia e quando chega ao seu fim. A fita é considerada um objeto unidimensional, que em princípio pode ser infinita, que consiste em uma sequência de espaços a serem preenchidos ou lidos pela máquina de Turing. Cada um desses espaços pode ser preenchido com um símbolo que pertence a um alfabeto Γ, que contém um número finito de símbolos distintos, por exemplo o conjunto $\{0, 1, b, \triangleright\}$, onde b representa um espaço em branco e $\triangleright$ é o símbolo que determina o início da fita, já 0 e 1 são usados para descrever a computação. O instrumento de leitura e gravação tem a capacidade de identificar um único espaço da fita e este é considerado o espaço que está sendo acessado pela máquina [1]. Tendo essa estrutura em mente, podemos dizer que um programa para a máquina de Turing é uma sequência de instruções que leva a máquina de um estado inicial a um estado final, através de deslocamentos na fita que são acompanhados de leituras e gravações, em geral o programa é denotado como

$$\langle q, x, q', x', s\rangle, \tag{1.68}$$

onde q e q' representam estados internos da máquina, já x e x' são símbolos presentes no alfabeto Γ que podem aparecer na fita. De forma resumida, a execução do programa acontece da seguinte maneira: primeiro a máquina verifica se na lista de programas existe alguma linha que comece com o estado q e símbolo x, caso não encontre tal linha o estado da máquina é alterado para q_h é o processo é dado como encerrado. Quando a linha do programa é encontrada, a máquina muda seu estado atual para q' e altera o símbolo x para x', depois disso ela dá uma ordem para o dispositivo de leitura e gravação baseada no comando s que pode ter valor $-1, 0$ ou 1, esses números indicam se o dispositivo deve andar para a esquerda, direita ou se manter parado na fita, caso este se encontre no início da fita e $s = -1$ não há movimento. Esse tipo de máquina permitiu Turing desenvolver o conceito de máquina universal de Turing, esta seria uma máquina capaz de simular qualquer outra máquina de Turing, além disso houve o desenvolvimento da tese de Church-Turing, que estabelece que as classes de funções que podem ser computadas por uma máquina de Turing correspondem as classes de funções que seriam consideradas como capazes de serem computadas por um algoritmo [1].

A máquina de Turing é um modelo idealizado que nos permite entender vários aspectos da teoria da computação, contudo não é muito conveniente para aplicações do

mundo real, isso fez com que fosse desenvolvido um modelo alternativo de computação mais prático e que dispusesse de uma capacidade computacional equivalente a de uma máquina de Turing. Este modelo é conhecido como modelo de circuitos e é baseado na álgebra de Boole.

Os componentes básicos de um circuito são fios e portas lógicas, unindo esses elementos podemos levar a informação de um canto para o outro e realizar algumas computações. Os fios simbolizam os bits presentes no circuitos e os transportam ao longo do circuito, já as portas lógicas são responsáveis pelas transformações dos bits. Matematicamente uma porta pode ser encarada como uma função $f : \{0,1\}^k \rightarrow \{0,1\}^l$ que mapeia um número fixo k de bits de entrada em outro número fixo l de bits de saída, por exemplo, a porta NOT pode ser encarada como a função $f(x) = 1 \oplus x$, onde x é um bit e $\oplus$ simboliza a soma módulo 2 [1]. Para realizar as tarefas desejadas, podemos combinar fios e portas, contudo existem certas combinações que devem ser evitadas, por exemplo fazer um laço em uma porta, ou seja, conectar sua entrada e saída. Outro aspecto importante a ser mencionado é que por padrão assume-se que todos os bits presentes no circuito são iniciados como um bit 0 e se queremos um estado inicial diferente devemos aplicar portas NOTs antes da computação desejada.

Na Figura 9, encontram-se algumas das portas lógicas presentes na computação clássica, vamos utilizar algumas delas para apresentar os conceitos de tabela verdade e combinação de portas. De início, temos a porta mais simples, a porta NOT, esta possui apenas um bit de entrada e um de saída, sua atuação consiste em inverter o valor do bit de entrada, ou seja, se temos um bit 0 na entrada dessa porta ele deve sair como 1 e vice-versa.

x	$\neg\, x$
0	1
1	0

Tabela 2 – Tabela verdade da porta NOT.

Além da porta NOT, temos as portas AND, OR e XOR, estas possuem dois bits de entrada e apenas um de saída. A primeira delas é inspirada no operador lógico matemático AND ($\wedge$), cuja tabela verdade 3 dita que o bit de saída só será igual a 1 quando ambos os bits de entrada forem iguais a 1, esse comportamento é equivalente ao do produto de números binários de 1 bit. A segunda porta que foi citada acima é a porta OR, esta representa o "ou" ($\vee$) da lógica matemática, cujo comportamento já é conhecido e encontra-se na Tabela 4. Em teoria de conjuntos o operador lógico "ou" representa a união entre dois conjuntos. A porta XOR trata dos casos em que temos um "ou" exclusivo, ou seja, quando uma das opções exclui a possibilidade da outra, por exemplo quando alguém nos pergunta se vamos viajar de carro ou de avião, independente de qual for a

x	y	$x \wedge y$
0	0	0
0	1	0
1	0	0
1	1	1

Tabela 3 – Tabela verdade da porta AND.

x	y	$x \vee y$
0	0	0
0	1	1
1	0	1
1	1	1

Tabela 4 – Tabela verdade da porta OR.

nossa resposta a outra opção é descartada. Além disso, esta porta representa a adição em módulo 2 como pode se observar a partir da Tabela 5.

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

Tabela 5 – Tabela verdade da porta XOR.

Existe também a possibilidade de adicionar controle nas portas, por exemplo a porta NOT controlada que é conhecida como CNOT (Figura 9 (e)), esta consiste em um bit de controle (círculo preenchido) e um bit alvo (símbolo $\oplus$) onde é aplicada uma porta NOT mediante a ter a condição do bit de controle satisfeita, no caso em questão a condição é que o bit de controle seja igual a 1, caso contrário nada ocorre. Essa porta terá um papel importante na computação quântica, pois ela que fará o papel de interação local para que os sistemas quânticos possam criar emaranhamento. A matriz que representa a porta CNOT pode ser vista abaixo

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{1.69}$$

Como foi dito anteriormente, existe a possibilidade de combinação de portas, aqui vamos dar como exemplos as portas NAND e NOR, a primeira combina as portas AND e

NOT, enquanto a segunda mistura OR com NOT, ambos os casos temos como resultado portas que possuem o comportamento oposto das originais. A porta NAND exerce um papel importante na computação clássica, pois junto com a porta COPY (como o próprio nome já diz, realiza cópias dos bits) formam um conjunto de portas para a computação universal, ou seja, as duas combinadas (com auxílio de bits auxiliares) podem reproduzir o comportamento de outras portas lógicas.

x	y	$x \wedge y$	$\neg(x \wedge y)$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Tabela 6 – Tabela verdade da porta NAND.

x	y	$x \vee y$	$\neg(x \vee y)$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Tabela 7 – Tabela verdade da porta NOR.

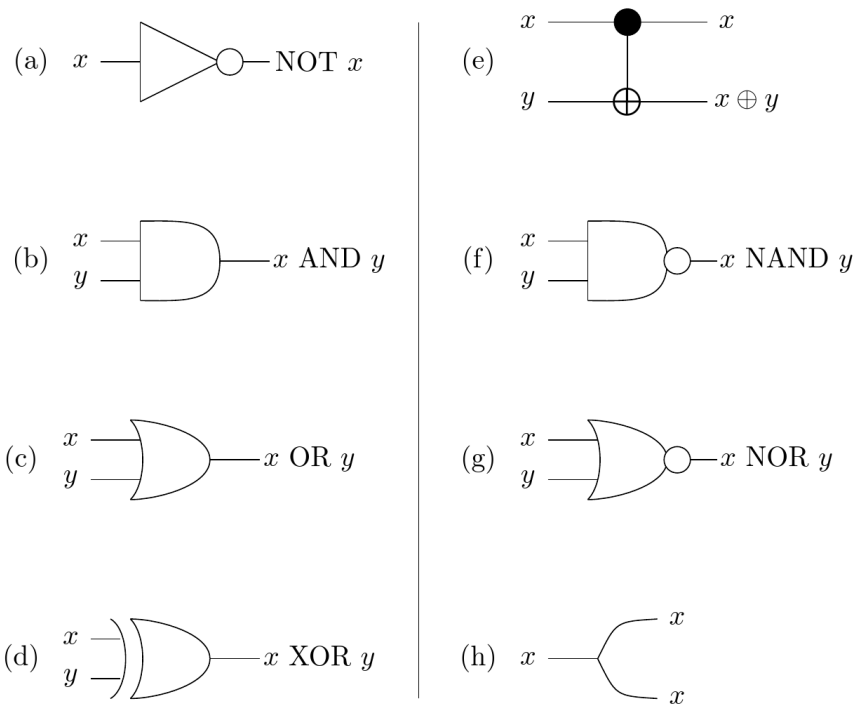


Figura 9 – Alguns exemplos de portas lógicas: (a) NOT. (b) AND. (c) OR. (d) XOR. (e) controlled-NOT (CNOT). (f) NAND. (g) NOR. (h) COPY ou FANOUT.

Um ponto que merece ser destacado, é que as portas com dois bits apresentadas possuem apenas um bit de saída, este fato faz com que elas sejam irreversíveis do ponto de vista lógico, uma vez que nos casos em que há degenerescência do bit de saída, por exemplo os pares (0,0), (0,1) e (1,0) na porta AND resultam em 0, fica impossível dizer qual foi o par de bits iniciais que o gerou, o que sugere que de alguma forma perdemos um bit de informação durante o processo lógico. Entretanto, o físico Rolf Landauer [1, 25, 27] descobriu que deletar um bit de informação dissipa pelo menos $k_B T \ln 2$ de energia, onde k_B representa a constante de Boltzmann e T a temperatura do ambiente do computador. Esta descoberta implica que existe uma ligação entre processos físicos dissipativos e processos lógicos irreversíveis, e ampliou o entendimento sobre a termodinâmica dos computadores, pois uma vez que os computadores lidam com quantidades finitas de energia, se faz necessário um uso otimizado desta energia para que seja possível realizar a maior quantidade de cálculos possíveis com a energia disponível. Posteriormente, Charles Bennett descobriu que havia uma forma de realizar algumas portas lógicas de forma reversível [1, 26], ou seja, sem dissipar energia. A descoberta de Bennett também foi importante para dar uma solução satisfatória para o paradoxo do demônio de Maxwell. Além disso, o resultado é extremamente importante para a viabilidade da computação quântica, pois uma vez que as portas lógicas podem ser feitas de forma reversível, podemos encontrar operadores da mecânica quântica que reproduzam o comportamento das portas lógicas, o que garante ao computador quântico a possibilidade de computar circuitos clássicos. Um exemplo simples é o caso da porta NOT, ela em si já é reversível e o operador quântico unitário que reproduz seu comportamento é a matriz de Pauli σ_x. Outra questão que irá surgir no contexto quântico, será a impossibilidade de encontrar um operador unitário que faça o papel de uma copiadora universal de estados, o que é conhecido como teorema da não clonagem. Veremos isso em mais detalhes posteriormente, mas podemos adiantar que o computador quântico pode realizar cópias de bits, o que não lhe é permitido fazer é a cópia de estados arbitrários não ortogonais.

Um exemplo de forma reversível de uma porta lógica é a porta Toffoli, ela é a versão reversível da porta AND, pois como podemos notar de sua tabela verdade 8, apresenta o mesmo resultado da porta AND com o auxílio de um bit extra e sem deletar nenhum dos bits de entrada, o que pode ser visto na Figura 10. Note que os bits de entrada são preservados na porta Toffoli, isso nos garante que temos condição de descobrir qual foi a entrada dada na porta a partir de seu resultado final. O resultado obtido por Tommaso Toffoli é importante, pois habilita a construção de um computador universal reversível.

Uma vez que conhecemos a atuação das portas lógicas e as regras para combiná-las, estamos aptos a construir circuitos que realizam alguma tarefa de interesse. Aqui vamos discutir dois circuitos, o primeiro que realiza a soma de dois números binários de 1 bit[4] e o

[4] Para maiores detalhes sobre aritmética binária o leitor deve consultar o apêndice A.

x	y	z	x'	y'	z'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Tabela 8 – Tabela verdade da porta Toffoli.

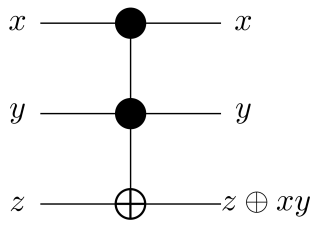


Figura 10 – Representação gráfica da porta Toffoli.

segundo que realiza o código de repetição de 3 bits. O primeiro circuito citado é conhecido como *half-adder*, este realiza a soma módulo 2 de dois bits de entrada, x e y, além disso também determina o bit de transporte c, este é igual a 1 quando as entradas x e y forem iguais a 1, caso contrário é igual a 0, conforme exibido na Tabela 9. Da tabela verdade

x	y	$x + y$
0	0	00
0	1	01
1	0	01
1	1	10

Tabela 9 – Tabela verdade do circuito *half-adder*.

da adição binária, podemos notar que o bit mais à esquerda dos resultados obedece ao comportamento da porta AND, já o bit mais à direita tem comportamento igual a da porta XOR. Sabendo disso, podemos usar essas portas para realizar o circuito *half-adder*, contudo apenas elas não completam o circuito, já que estamos usando duas portas com dois bits de entrada cada, totalizando 4 bits de entrada, sendo que só dispomos de dois bits, para contornar essa situação devemos adicionar duas portas COPY ao circuito, isso garante que vamos atender a quantidade de bits demandados. Juntando todas as peças devemos obter um circuito igual ao apresentado na Figura 11.

O segundo circuito de interesse é que o performa o código de repetição de 3 bits, aqui serão necessárias portas COPY para realizar a redundância da informação e bits

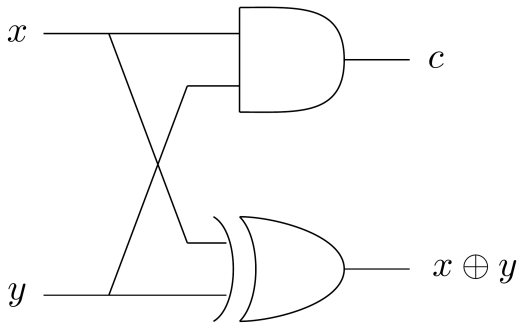


Figura 11 – Representação gráfica do circuito *half-adder*.

auxiliares para guardar informações sobre a mensagem durante o processo de decodificação. Na Figura 12 temos uma representação gráfica do circuito em questão, note que as duas CNOTs mais à esquerda representam o processo de codificação da mensagem original, ou seja, nesta etapa que é feita a redundância da informação, em seguida temos a verificação da paridade de acordo com (1.54), verificamos o bit 0 e o bit 1, depois guardamos o resultado em um bit auxiliar, fazemos o mesmo com os bits 1 e 2. Por fim, temos a etapa de decodificação e correção de erros, aqui vemos CNOTs e portas Toffoli tendo como bits de controle os bits auxiliares, isso faz com que uma correção seja aplicada em decorrência do valor guardado no bit auxiliar, por exemplo, considere que a mensagem postada na entrada do canal foi 000 e que durante o processo de transmissão o bit mais à esquerda sofreu um *bit-flip* transformando a mensagem em 100, com isso temos que a passagem da mensagem pelo teste de paridade fará com que os bits auxiliares assumam o valor 1, lembrando que por padrão inicialmente encontravam-se com valor 0, o que ativará as CNOTs e Toffoli da correção de erros, porém devemos notar que as CNOTs com alvo nos bits 1 e 2, tem suas atuações canceladas pelas Toffoli com alvo nos bits 1 e 2, o que faz com que a única porta a produzir alguma modificação na mensagem seja a Toffoli com alvo no bit 0, que é o bit com problemas, isso fará com que ele retorne ao seu valor inicial, o que garante que a mensagem entregue foi a correta.

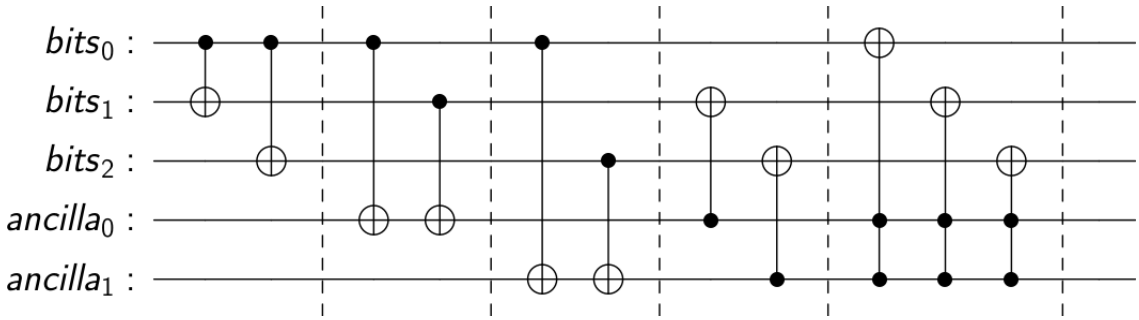


Figura 12 – Representação gráfica do circuito que realiza o código de repetição de 3 bits apresentado na seção anterior.

2 Mecânica Quântica

"So Einstein was wrong when he said, 'God does not play dice.' Consideration of black holes suggests, not only that God does play dice, but that he sometimes confuses us by throwing them where they can't be seen."
- Stephen Hawking

"Anybody who is not shocked by quantum theory has not understood it."
- Niels Bohr

2.1 Conceitos Básicos de Álgebra Linear

Antes de iniciar a apresentação dos conceitos básicos de álgebra linear que serão utilizados durante o texto, devemos estabelecer uma convenção a respeito da notação. Como estamos interessados em explorar a mecânica quântica, então desde o início adotaremos a notação de Dirac, portanto os vetores serão denotados como "*kets*" $|v\rangle$ e os vetores duais como "*bras*" $\langle w|$, estes são considerados como matrizes do tipo coluna e linha

$$|v\rangle = \begin{bmatrix} v_0 \\ \vdots \\ v_n \end{bmatrix} \text{ e } \langle w| = \begin{bmatrix} w_0 & \dots & w_n \end{bmatrix}, \tag{2.1}$$

onde $v_0, \dots, v_n$ e $w_0, \dots, w_n$ são as coordenadas dos vetores. É importante ressaltar que vamos adotar $\vec{0}$ para o vetor nulo em vez de $|0\rangle$, pois posteriormente utilizaremos $|0\rangle$ como um dos vetores da base computacional da computação quântica. Uma vez estabelecida a notação, podemos prosseguir para as definições de espaço vetorial e em seguida explorar as ferramentas que esse tipo de espaço tem a nos oferecer.

O espaço vetorial que serve de "plano de fundo" da mecânica quântica é conhecido como espaço de Hilbert $\mathcal{H}$, este é um espaço vetorial linear complexo dotado de um produto interno. Este espaço apresenta uma regra de adição e uma de multiplicação por escalar, sendo que a primeira deve apresentar as seguintes propriedades: sejam $|\psi\rangle$ e $|\phi\rangle$ vetores pertencentes ao espaço $\mathcal{H}$, então a soma $|\psi\rangle + |\phi\rangle$ resulta em um vetor que também pertence a este espaço; comutatividade ($|\psi\rangle + |\phi\rangle = |\phi\rangle + |\psi\rangle$); associatividade ($(|\psi\rangle + |\phi\rangle) + |\chi\rangle = |\psi\rangle + (|\phi\rangle + |\chi\rangle)$); existência de elemento neutro ($\vec{0} + |\psi\rangle = |\psi\rangle + \vec{0} = |\psi\rangle$) e a existência de elemento inverso ($|\psi\rangle + (-|\psi\rangle) = \vec{0}$). A regra da multiplicação de vetores por um escalar obedece às propriedades: seja $|\psi\rangle$ um vetor de $\mathcal{H}$ e α uma constante complexa, então o vetor $\alpha|\psi\rangle$ também faz parte de $\mathcal{H}$; distributividade ($\alpha(|\psi\rangle + |\phi\rangle) =$

$\alpha|\psi\rangle+\alpha|\phi\rangle$ e $(\alpha+\beta)|\psi\rangle = \alpha|\psi\rangle+\beta|\psi\rangle$); associatividade $(\alpha(\beta|\psi\rangle) = (\alpha\beta)|\psi\rangle)$ e a existência do escalar unitário e do zero escalar $(1|\psi\rangle = |\psi\rangle 1 = |\psi\rangle$ e $0|\psi\rangle = |\psi\rangle 0 = \vec{0})$.

Existe uma operação no espaço que relaciona os "*kets*" e "*bras*", esta é conhecida como o Hermitiano conjugado, cujo símbolo é uma adaga †, sua aplicação sobre um ket produz o resultado

$$\langle\psi| = (|\psi\rangle)^\dagger \;\equiv\; \langle\psi| = [(|\psi\rangle)^T]^*. \tag{2.2}$$

Note que operacionalmente o Hermitiano conjugado consiste em transpor a matriz coluna do vetor, transformando-a em uma matriz linha, e por fim tomar o complexo conjugado das coordenadas do vetor. Este operador também pode atuar sobre uma combinação linear de vetores, nesse cenário sua atuação produz o seguinte resultado

$$(\alpha|\psi\rangle + \beta|\phi\rangle)^\dagger = \langle\psi|\alpha^* + \langle\phi|\beta^*. \tag{2.3}$$

O produto interno, $\langle\psi|\phi\rangle$ ("*braket*"), é uma função que mapeia dois vetores, $|\psi\rangle$ e $|\phi\rangle$, do espaço em um número complexo. Considere que $|\psi\rangle$ e $|\phi\rangle$ são os seguintes vetores coluna

$$|\psi\rangle = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \text{ e } |\phi\rangle = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}, \tag{2.4}$$

então temos que o produto interno entre eles é igual a

$$\langle\psi|\phi\rangle = \begin{bmatrix} a_1^* & \dots & a_n^* \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} = \sum_{i=1}^{n} a_i^* b_i \;\in \mathbb{C}. \tag{2.5}$$

Observe que como o espaço de Hilbert é um espaço vetorial complexo, a ordem dos vetores no produto interno importa para o resultado final, dessa forma temos que quando há mudança na ordem dos vetores é preciso tomar o complexo conjugado do resultado para que os resultados sejam iguais

$$\langle\psi|\phi\rangle = \langle\phi|\psi\rangle^*. \tag{2.6}$$

Além disso, o produto interno satisfazer as seguintes propriedades

$$\langle\phi|(\alpha|\psi_1\rangle + \beta|\psi_2\rangle) = \alpha\langle\phi|\psi_1\rangle + \beta\langle\phi|\psi_2\rangle; \tag{2.7}$$

$$(\langle\phi_1|\alpha^* + \langle\phi_2|\beta^*)|\psi\rangle = \alpha^*\langle\phi_1|\psi\rangle + \beta^*\langle\phi_2|\psi\rangle; \tag{2.8}$$

$$\langle\psi|\psi\rangle = \|\psi\|^2 \geq 0;\; \|\psi\|^2 = 0 \iff |\psi\rangle = \vec{0}, \tag{2.9}$$

se tomarmos a raiz quadrada em ambos os lados da equação (2.9), temos uma expressão para calcular a norma de um vetor no espaço de Hilbert

$$\|\psi\| = \sqrt{\langle\psi|\psi\rangle}. \tag{2.10}$$

Na mecânica quântica, trabalhamos com vetores de norma igual a 1 para satisfazer a condição de normalização da teoria de probabilidade, portanto se um vetor de interesse possui norma maior que 1, podemos transformá-lo em um vetor unitário multiplicando-o pelo inverso de sua norma

$$|\psi'\rangle = \frac{1}{\|\psi\|}|\psi\rangle. \tag{2.11}$$

Através do produto interno também é possível dizer se dois vetores são ortogonais ou não, então se considerarmos dois vetores, $|\psi\rangle$ e $|\phi\rangle$, podemos dizer que estes são ortogonais se

$$\langle\psi|\phi\rangle = 0. \tag{2.12}$$

Uma base de um espaço vetorial consiste em um conjunto de vetores linearmente independentes, cuja dimensão da base é igual ao número de vetores presentes no conjunto, esse número pode ser finito ou não, mas para os temas que serão abordados neste trabalho nos limitaremos a usar apenas bases com número finito de vetores. Dado um conjunto com n vetores não nulos $\{\phi_i\}$, temos que o conjunto é dito linearmente independente, se e somente se a solução de

$$\sum_{i=1}^{n} \alpha_i|\phi_i\rangle = 0, \tag{2.13}$$

for a solução trivial, ou seja, todas as constantes $\alpha_i = 0$. A prova dessa condição pode ser feita da seguinte maneira, primeiro vamos considerar que os vetores do conjunto são ortonormais e que o produto interno entre eles é igual a

$$\langle\phi_i|\phi_j\rangle = \delta_{ij} = \begin{cases} 0, \text{ se } i \neq j \\ 1, \text{ se } i = j \end{cases}, \tag{2.14}$$

onde δ_{ij} representa a delta de Kronecker. Sendo assim, se escrevermos (2.13) explicitamente e tomarmos o produto interno de em relação ao vetor $|\phi_1\rangle$ chegaremos ao seguinte resultado

$$\begin{aligned} &\alpha_1\langle\phi_1|\phi_1\rangle + \alpha_2\underbrace{\langle\phi_1|\phi_2\rangle}_{=0} + \cdots + \alpha_n\underbrace{\langle\phi_1|\phi_n\rangle}_{=0} = 0 \\ &\alpha_1\underbrace{\|\phi_1\|^2}_{\neq 0} = 0;\ |\phi_1\rangle \neq \vec{0} \\ &\therefore \alpha_1 = 0. \end{aligned} \tag{2.15}$$

Então se repetirmos o procedimento acima para os demais vetores, chegaremos à conclusão de que as demais constantes α_i devem ser todas iguais a zero. Entretanto, se uma das constantes α_i for diferente de zero, temos que o conjunto de vetores $\{\phi_i\}$ é um conjunto linearmente dependente, ou seja, pelo menos um dos vetores pode ser escrito como combinação linear de outros vetores da base. Neste caso, é possível transformar o conjunto

de vetores $\{\phi_i\}$ em um conjunto linearmente independente através do método de Gram-Schmidt [1]

$$|\phi'_k\rangle = \frac{|\phi_k\rangle - \sum_{i=1}^{k-1}\langle\phi_i|\phi_k\rangle|\phi_i\rangle}{\||\phi_k\rangle - \sum_{i=1}^{k-1}\langle\phi_i|\phi_k\rangle|\phi_i\rangle\|}, \tag{2.16}$$

onde $|\phi_k\rangle$ representa o vetor que pode ser escrito como combinação linear de outros vetores da base e $|\phi'_k\rangle$ é o vetor ortogonal aos vetores do conjunto $\{\phi_i\}$.

Outro produto presente no espaço de Hilbert é o produto externo, $|\psi\rangle\langle\phi|$, também conhecido como "*ketbra*", este leva dois vetores do espaço em uma matriz e é muito usado para representar os operadores da mecânica quântica na forma de matrizes. Considerando novamente os vetores $|\psi\rangle$ e $|\phi\rangle$ usados na definição do produto interno, temos que o produto externo desses vetores resulta na matriz

$$|\psi\rangle\langle\phi| = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \begin{bmatrix} b_1^* & \dots & b_n^* \end{bmatrix} = \begin{bmatrix} a_1b_1^* & a_1b_2^* & \dots & a_1b_n^* \\ a_2b_1^* & a_2b_2^* & \dots & a_2b_n^* \\ \vdots & \vdots & \ddots & \vdots \\ a_nb_1^* & a_nb_2^* & \dots & a_nb_n^* \end{bmatrix}. \tag{2.17}$$

Podemos usar o produto externo para obter a relação de completeza dos vetores ortonormais, sendo assim vamos considerar o seguinte cenário: dada uma base $\{|i\rangle\}$ para o espaço vetorial $\mathcal{H}$, é possível escrever qualquer vetor $|\psi\rangle$, que também pertença a $\mathcal{H}$ em termos desses vetores base da seguinte forma

$$|\psi\rangle = \sum_i c_i|i\rangle;\ c_i \in \mathbb{C}, \tag{2.18}$$

onde c_i representam as coordenadas de $|\psi\rangle$ na base $\{|i\rangle\}$ e podem ser determinadas através do produto interno entre $|\psi\rangle$ e os vetores $|i\rangle$, ou seja,

$$c_i = \langle i|\psi\rangle. \tag{2.19}$$

Sendo assim, tomando o produto externo dos vetores $|i\rangle$ e somando as matrizes resultantes, vamos obter uma matriz que pode ser aplicada sobre o vetor $|\psi\rangle$ da seguinte forma

$$\left(\sum_{i=1}^{n}|i\rangle\langle i|\right)|\psi\rangle = \sum_i\langle i|\psi\rangle|i\rangle = \sum_i c_i|i\rangle = |\psi\rangle. \tag{2.20}$$

Note que o resultado final foi o próprio vetor $|\psi\rangle$, o que nos diz que a matriz em questão não altera o vetor, mas este comportamento é característico da matriz identidade $\mathbb{I}$, portanto podemos chegar a conclusão que

$$\sum_{i=1}^{n}|i\rangle\langle i| = \mathbb{I} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}_{n\times n}. \tag{2.21}$$

O resultado acima é conhecido como a relação de completeza. Uma das aplicações dessa relação é na obtenção da forma matricial de operadores lineares, por exemplo considere um operador O que conecta duas bases ortonormais $|i\rangle$ e $|j\rangle$. Então, se queremos obter a matriz desse operador devemos usar a relação de completeza e realizar os seguintes cálculos

$$\begin{aligned} O &= \mathbb{I}O\mathbb{I} \\ &= \left(\sum_{i=1}^{n} |i\rangle\langle i|\right) O \left(\sum_{j=1}^{n} |j\rangle\langle j|\right) \\ &= \sum_{i,j=1}^{n} |i\rangle\underbrace{\langle i|O|j\rangle}_{O_{ij}}\langle j| = \sum_{i,j=1}^{n} O_{ij}|i\rangle\langle j|, \end{aligned} \tag{2.22}$$

onde nosso resultado final é a representação de O em termos do produto externo, sendo que O_{ij} representam os elementos de matriz. Podemos visualizar a equação acima como a seguinte matriz:

$$O = \begin{bmatrix} O_{11} & O_{12} & \dots & O_{1n} \\ O_{21} & O_{22} & \dots & O_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ O_{n1} & O_{n2} & \dots & O_{nn} \end{bmatrix} = \begin{bmatrix} \langle 1|O|1\rangle & \langle 1|O|2\rangle & \dots & \langle 1|O|n\rangle \\ \langle 2|O|1\rangle & \langle 2|O|2\rangle & \dots & \langle 2|O|n\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle n|O|1\rangle & \langle n|O|2\rangle & \dots & \langle n|O|n\rangle \end{bmatrix}. \tag{2.23}$$

Dois elementos importantes das matrizes são seus autovalores e autovetores, estes nos dão informações sobre a matriz. Dito isso, temos que um autovetor da matriz de um operador linear O é um vetor não nulo que satisfaz a equação

$$O|\lambda\rangle = \lambda|\lambda\rangle \tag{2.24}$$

onde λ representa um autovalor da matriz O. O conjunto de autovalores $\{\lambda_n\}$ pode ser obtido resolvendo a equação característica

$$\det(O - \lambda\mathbb{I}) = 0, \tag{2.25}$$

e tendo todos os autovalores em mãos podemos retornar a (2.24) e determinar o conjunto de autovetores $\{|\lambda_n\rangle\}$ de O. É importante ressaltar que se os autovalores de uma matriz não apresentarem degenerescência, então os autovetores dela serão ortogonais entre si e formam uma base em que a matriz pode ser escrita na forma diagonal. Este procedimento é conhecido como a diagonalização de uma matriz. Por exemplo, considerando que os autovalores λ_n de O não apresentam degenerescência, então os autovetores $|\lambda_n\rangle$ associados a estes autovalores são ortogonais entre si e podem ser usados para escrever O na forma diagonal

$$O = \sum_{n} \lambda_n |\lambda_n\rangle\langle\lambda_n|. \tag{2.26}$$

Quando a matriz apresenta autovalores degenerados os autovetores não são automaticamente ortogonais, para transformá-los em um conjunto de vetores ortogonais é necessário fazer uso do método de Gram-Schmidt (2.16).

Os operadores atuam no espaço vetorial transformando os vetores $|\psi\rangle$ em outros vetores $|\psi'\rangle$ que pertencem ao mesmo espaço, ou seja, a atuação deles pode ser entendida como

$$|\psi'\rangle = O|\psi\rangle. \tag{2.27}$$

Uma vez que estamos lidando com espaços vetoriais lineares, temos que os operadores presentes no espaço devem obedecer uma lei de distributividade e de comutação com constantes, ou seja,

$$O(\alpha|\psi\rangle + \beta|\phi\rangle) = \alpha O|\psi\rangle + \beta O|\phi\rangle. \tag{2.28}$$

Além disso, temos que o produto de operadores é uma operação definida, mas como estes podem ser representados por matrizes devemos tomar o cuidado com a ordem do produto, já que em geral o produto de matrizes é não comutativo. A operação produto satisfaz as propriedades:

$$\text{Distributividade: } A(B+C) = AB + AC \text{ e } (B+C)A = BC + CA; \tag{2.29}$$

$$\text{Associatividade: } ABC = A(BC) = (AB)C; \tag{2.30}$$

$$A^n A^m = A^{n+m}; \tag{2.31}$$

$$\langle\phi|A|\psi\rangle \in \mathbb{C}. \tag{2.32}$$

Da última propriedade apresentada acima, podemos definir o que vamos chamar de valor médio do operador na mecânica quântica, para isso basta trocar o $\langle\phi|$ por um $\langle\psi|$ e teremos a definição matemática da quantidade de interesse

$$\langle A\rangle_\psi = \langle\psi|A|\psi\rangle, \tag{2.33}$$

onde $\langle A\rangle_\psi$ representa o valor médio de A calculado a partir de $|\psi\rangle$.

Anteriormente foi mostrado como o operador † atuava sobre as constantes complexas e sobre os vetores do espaço vetorial, contudo agora estamos interessados em como ele atua sobre as matrizes dos operadores, pois isso vai nos ajudar a definir uma classe de operadores que é muito utilizada na mecânica quântica. Sendo assim, temos que o Hermitiano conjugado de um operador O é denotado como $O^\dagger$ e é definido pela relação [32]

$$\langle\psi|O^\dagger|\phi\rangle = (\langle\phi|O|\psi\rangle)^*, \tag{2.34}$$

além disso a operação Hermitiano conjugado deve satisfazer as regras:

$$(O^\dagger)^\dagger = O; \tag{2.35}$$

$$(\alpha O)^\dagger = \alpha^* O^\dagger; \tag{2.36}$$

$$(O^n)^\dagger = (O^\dagger)^n; \tag{2.37}$$

$$(O_1 + O_2)^\dagger = O_1^\dagger + O_2^\dagger; \tag{2.38}$$

$$(O_1 O_2)^\dagger = O_2^\dagger O_1^\dagger. \tag{2.39}$$

Quando tomamos Hermitiano conjugado de um operador e ele apresenta o comportamento abaixo

$$\langle\psi|O|\phi\rangle = (\langle\phi|O|\psi\rangle)^* \;\Rightarrow\; O = O^\dagger, \tag{2.40}$$

o classificamos como um operador Hermitiano. Essa classe de operadores é muito importante para a mecânica quântica, pois representam os observáveis físicos uma vez que estes possuem autovalores reais e autovetores ortogonais, na ausência de degenerescência. Este fato pode ser demonstrado considerando o seguinte cenário: considere um operador A cujos autovetores pertencem ao conjunto de vetores $\{\phi_i\}$, então para os autovetores de número n e m temos que

$$A|\phi_n\rangle = \lambda_n|\phi_n\rangle \;\Rightarrow\; \langle\phi_m|A|\phi_n\rangle = \lambda_n\langle\phi_m|\phi_n\rangle \tag{2.41}$$

$$\langle\phi_m A^\dagger = \langle\phi_m|\lambda_m^* \;\Rightarrow\; \langle\phi_n|A^\dagger|\phi_m\rangle = \lambda_m^*\langle\phi_m|\phi_n\rangle, \tag{2.42}$$

então subtraindo as expressões acima e assumindo que $A = A^\dagger$, temos

$$(\lambda_n - \lambda_m^*)\langle\phi_m|\phi_n\rangle = 0. \tag{2.43}$$

Neste ponto devemos analisar a expressão acima em dois casos, o primeiro quando $n = m$, sendo assim o produto interno passa a ser $\langle\phi_n|\phi_n\rangle = \|\phi_n\|^2 \geq 0$, e considerando que $|\phi_n\rangle \neq \vec{0}$, chegamos a

$$\lambda_n - \lambda_n^* = 0 \;\Rightarrow\; \lambda_n = \lambda_n^* \;\therefore\; \lambda_n \in \mathbb{R}, \tag{2.44}$$

o que nos faz concluir que λ_n deve ser real para que a equação (2.43) seja satisfeita. O segundo caso de interesse é quando $n \neq m$, sendo assim se considerarmos que em geral $\lambda_n \neq \lambda_m^*$, então devemos ter $\langle\phi_m|\phi_n\rangle = 0$ para que (2.43) seja satisfeita, ou seja, os autovetores devem ser ortogonais e assim fica demonstrado essa propriedade particular dos operadores Hermitianos.

Outra classe de operadores que é importante no contexto da mecânica quântica é a dos operadores unitários, pois eles representam os operadores de evolução temporal devido a sua propriedade especial de preservar o produto interno entre vetores e por consequência a norma dos vetores. Uma matriz U é dita unitária quando satisfaz a condição

$$U^\dagger U = UU^\dagger = \mathbb{I}, \tag{2.45}$$

ou seja, o Hermitiano conjugado de U deve ser igual a sua inversa U^{-1}. Da definição acima, é possível ver como os operadores unitários mantém o produto interno invariante, para isso considere dois vetores, $|\psi\rangle$ e $|\phi\rangle$, cujo produto interno é igual a $\langle\psi|\phi\rangle$, sabendo que a atuação de U sobre os vetores produz

$$|\psi'\rangle = U|\psi\rangle \text{ e } |\phi'\rangle = U|\phi\rangle, \tag{2.46}$$

então tomando o produto interno entre $|\psi'\rangle$ e $|\phi'\rangle$, temos que

$$\langle\psi'|\phi'\rangle = \langle\psi|U^\dagger U|\phi\rangle = \langle\psi|\mathbb{I}|\phi\rangle = \langle\psi|\phi\rangle. \tag{2.47}$$

Um tipo especial de operadores que vamos discutir aqui são os projetores, estes são usados como operadores de medida e representam matrizes de densidade de estados puros. Um operador é dito um projetor P quando satisfaz as condições

$$P^\dagger = P \text{ e } P^2 = P, \tag{2.48}$$

além disso operadores projeção possuem autovalores iguais a 0 e 1. Um exemplo trivial de operador que satisfaz essas condições é o operador identidade, pois note que

$$\mathbb{I}^\dagger = (\sum_{i=1}^{n} |i\rangle\langle i|)^\dagger = \mathbb{I} \tag{2.49}$$

$$\mathbb{I}^2 = \sum_{i,j=1}^{n} |i\rangle\underbrace{\langle j|i\rangle}_{\delta_{ij}}\langle j| = \mathbb{I}. \tag{2.50}$$

Também precisamos definir o comutador entre dois operadores, pois estes serão usados mais à frente na definição dos estabilizadores. Para isso, considere A e B dois operadores e que $[A, B]$ denota o comutador de A e B, cuja definição é

$$[A, B] \equiv AB - BA. \tag{2.51}$$

Caso $[A, B] = 0$, dizemos que A comuta com B, este fato implica que a ordem do produto dos operadores não importa ($AB = BA$) e que ambos podem ser diagonalizados na mesma base. A definição de comutador dada acima satisfaz as seguintes propriedades [32]:

$$[\alpha, A] = 0,\ \alpha \in \mathbb{C}; \tag{2.52}$$

$$[A, B] = -[B, A]; \tag{2.53}$$

$$[A, B + C] = [A, B] + [A, C]; \tag{2.54}$$

$$[A, BC] = [A, B]C + B[A, C]; \tag{2.55}$$

$$[AB, C] = [A, C]B + A[B, C]; \tag{2.56}$$

$$[A, [B, C]] + [B, [C, A]] + [C, [A, B]] = 0. \text{ (Identidade de Jacobi)} \tag{2.57}$$

A relação de comutação entre dois operadores é importante para definir a relação de incerteza entre dois operadores. Essa informação é muito importante na mecânica quântica, pois ela nos diz se dois operadores podem ser medidos simultaneamente sem que exista uma incerteza associada a medida de ambos ($[A, B] = 0$) e também estabelece as limitações que teremos ao medir dois operadores incompatíveis ($[A, B] \neq 0$), pois não importa o quão precisos sejam nossos aparelhos de medida, neste caso sempre haverá uma incerteza associada. Podemos calcular essa quantidade através da fórmula [32]

$$\Delta A \Delta B \geq \frac{1}{2}|\langle[A, B]\rangle|, \tag{2.58}$$

onde ΔA e ΔB representam o desvio padrão dos operadores A e B que por definição são iguais a

$$\Delta A = \sqrt{\langle A^2 \rangle - \langle A \rangle^2} \tag{2.59}$$

$$\Delta B = \sqrt{\langle B^2 \rangle - \langle B \rangle^2}. \tag{2.60}$$

Através de (2.58) que podemos obter a famosa relação de incerteza de Heisenberg

$$\Delta x \Delta p \geq \frac{\hbar}{2}, \tag{2.61}$$

que nos diz que os operadores posição x e momento linear p são incompatíveis e que portanto em uma medida simultânea de ambos, sempre deve haver no mínimo uma incerteza de $\hbar/2$, isto ocorre devido ao fato de $[x, p] = i\hbar$ e nos impõe um limite na precisão que podemos obter informações sobre ambos. De maneira similar podemos definir o anticomutador dos operadores A e B como

$$\{A, B\}_+ \equiv AB + BA. \tag{2.62}$$

Essas duas operações serão muito utilizadas no contexto de teoria quântica de correção de erros para definir o grupo de estabilizadores que definirão o código de correção de erros.

Existem funções conhecidas para trabalhar com números reais e complexos que também podem ser definidas para matrize: dada uma função f que leva números complexos a outros números complexos, pode ser definida como uma função matricial f. Para utilizar esse tipo de recurso é necessário realizar a decomposição espectral de um operador normal O de interesse, dessa forma podemos definir a atuação da função f sobre O da seguinte forma [1, 32]

$$f(O) = \sum_n f(\lambda_n) |\lambda_n\rangle\langle\lambda_n|. \tag{2.63}$$

A fórmula acima é importante, pois como veremos adiante, algumas quantidades essenciais em computação e informação quântica utilizam funções sobre operadores na forma matricial, por exemplo a entropia de von Neumann e a fidelidade de estados quânticos. Uma função matricial que merece destaque é a função traço, esta determina a soma dos elementos da diagonal principal da matriz do operador, para defini-la vamos considerar um operador O e uma base ortonormal $|i\rangle$, sendo assim temos que

$$\mathrm{tr}(O) = \sum_{i=1}^{n} \langle i|O|i\rangle = \sum_{i=1}^{n} O_{ii}. \tag{2.64}$$

Da definição acima é possível mostrar que a função traço satisfaz as propriedades

$$\mathrm{tr}(A^\dagger) = [\mathrm{tr}(A)]^*; \tag{2.65}$$

$$\mathrm{tr}(\alpha A + \beta B) = \alpha \mathrm{tr}(A) + \beta \mathrm{tr}(B); \tag{2.66}$$

$$\mathrm{tr}(A^T) = \mathrm{tr}(A); \tag{2.67}$$

$$\mathrm{tr}(AB) = \mathrm{tr}(BA); \tag{2.68}$$

$$\mathrm{tr}(ABC) = \mathrm{tr}(CAB) = \mathrm{tr}(BCA); \tag{2.69}$$

$$\mathrm{tr}([A, B]) = 0. \tag{2.70}$$

Além disso, essa função apresenta uma característica muito interessante, ela independe da base em que é expressada.

Por fim, precisamos definir uma operação que nos permita trabalhar com vários sistemas físicos, que matematicamente corresponde a expandir o espaço vetorial. Para entender como esse processo funciona considere dois espaços vetoriais $\mathcal{H}_A$ e $\mathcal{H}_B$, cujas respectivas dimensões são iguais a n e m, sendo assim temos que o espaço vetorial resultante do produto tensorial entre $\mathcal{H}_A$ e $\mathcal{H}_B$ é igual a

$$\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B. \tag{2.71}$$

A dimensão deste novo espaço pode ser calculada através do produto das dimensões dos espaços considerados no produto tensorial, ou seja,

$$\dim(\mathcal{H}_{AB}) = \dim(\mathcal{H}_A)\dim(\mathcal{H}_B). \tag{2.72}$$

Os vetores presentes no novo espaço são escritos como o produto tensorial dos vetores $|\psi_A\rangle$ e $|\psi_B\rangle$ pertencentes aos espaços $\mathcal{H}_A$ e $\mathcal{H}_B$

$$|\psi_{AB}\rangle = |\psi_A\rangle \otimes |\psi_B\rangle = |\psi_A\psi_B\rangle. \tag{2.73}$$

Usando a forma matricial dos vetores envolvidos no produto tensorial, podemos reescrever a expressão acima da seguinte forma

$$|\psi_{AB}\rangle = \begin{bmatrix} \alpha_1 \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_m \end{bmatrix} \\ \vdots \\ \alpha_n \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_m \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \alpha_1\beta_1 \\ \vdots \\ \alpha_1\beta_m \\ \vdots \\ \alpha_n\beta_1 \\ \vdots \\ \alpha_n\beta_m \end{bmatrix}_{1\times\dim(\mathcal{H}_{AB})} . \tag{2.74}$$

Uma vez que o espaço resultante do produto tensorial $\mathcal{H}_A \otimes \mathcal{H}_B$ também é um espaço vetorial linear, então temos que a soma de dois vetores do novo espaço $\mathcal{H}_{AB}$ também

pertence ao espaço. Além disso o produto tensorial por definição satisfaz as propriedades [1]:

$$\alpha(|\psi_A\rangle \otimes |\psi_B\rangle) = (\alpha|\psi_A\rangle) \otimes |\psi_B\rangle = |\psi_A\rangle \otimes (\alpha|\psi_B\rangle),\ \alpha \in \mathbb{C}; \tag{2.75}$$

$$(|\psi_A\rangle + |\phi_A\rangle) \otimes |\psi_B\rangle = |\psi_A\rangle \otimes |\psi_B\rangle + |\phi_A\rangle \otimes |\psi_B\rangle; \tag{2.76}$$

$$|\psi_A\rangle \otimes (|\psi_B\rangle + |\phi_B\rangle) = |\psi_A\rangle \otimes |\psi_B\rangle + |\psi_A\rangle \otimes |\phi_B\rangle. \tag{2.77}$$

O produto interno entre dois vetores de $\mathcal{H}_{AB}$ também está definido nesse novo espaço, para defini-lo vamos considerar os vetores

$$|\psi_{AB}\rangle = \sum_{i=1}^{\dim(\mathcal{H}_{AB})} \alpha_i |\psi_A^{(i)}\rangle \otimes |\psi_B^{(i)}\rangle \tag{2.78}$$

$$|\phi_{AB}\rangle = \sum_{i=1}^{\dim(\mathcal{H}_{AB})} \beta_i |\phi_A^{(i)}\rangle \otimes |\phi_B^{(i)}\rangle, \tag{2.79}$$

sendo assim o produto interno $\langle\psi_{AB}|\phi_{AB}\rangle$ pode ser definido como

$$\langle\psi_{AB}|\phi_{AB}\rangle = \sum_{i,j=1}^{\dim(\mathcal{H}_{AB})} \alpha_i^* \beta_j \langle\psi_A^{(i)}|\phi_A^{(j)}\rangle \langle\psi_B^{(i)}|\phi_B^{(j)}\rangle\ \in \mathbb{C}. \tag{2.80}$$

Além dos vetores, precisamos adequar os operadores de ambos os espaços envolvidos no produto tensorial, considerando que temos um operador O_A que atua sobre os vetores de $\mathcal{H}_A$, cuja matriz possui dimensões $n \times m$ e que temos um operador O_B que atua sobre os vetores de $\mathcal{H}_B$, cuja matriz possui dimensões $p \times q$, então a matriz do operador $O_A \otimes O_B$ que atua nos vetores de $\mathcal{H}_{AB}$, é obtida da seguinte forma

$$O_A \otimes O_B = \begin{bmatrix} o_{11}^A \begin{bmatrix} o_{11}^B & o_{12}^B & \dots & o_{1q}^B \\ \vdots & \ddots & \vdots & \vdots \\ o_{p1}^B & o_{p2}^B & \dots & o_{pq}^B \end{bmatrix}_{p\times q} & \dots & o_{1m}^A \begin{bmatrix} o_{11}^B & o_{12}^B & \dots & o_{1q}^B \\ \vdots & \ddots & \vdots & \vdots \\ o_{p1}^B & o_{p2}^B & \dots & o_{pq}^B \end{bmatrix}_{p\times q} \\ \vdots & \ddots & \vdots \\ o_{n1}^A \begin{bmatrix} o_{11}^B & o_{12}^B & \dots & o_{1q}^B \\ \vdots & \ddots & \vdots & \vdots \\ o_{p1}^B & o_{p2}^B & \dots & o_{pq}^B \end{bmatrix}_{p\times q} & \dots & o_{nm}^A \begin{bmatrix} o_{11}^B & o_{12}^B & \dots & o_{1q}^B \\ \vdots & \ddots & \vdots & \vdots \\ o_{p1}^B & o_{p2}^B & \dots & o_{pq}^B \end{bmatrix}_{p\times q} \end{bmatrix}_{n\times m} \tag{2.81}$$

Assim como no caso de vetores, as matrizes também apresentam aumento das dimensões, neste caso a matriz resultante teria dimensões $np \times mq$. Quando o operador atua em apenas um dos subespaços que fazem parte do espaço conjunto, os operadores devem apresentar a forma

$$O_A \otimes \mathbb{I}_B \tag{2.82}$$

$$\mathbb{I}_A \otimes O_B, \tag{2.83}$$

pois isso garante que estes só transformarão os vetores do subespaço de interesse. Esse tipo de operador será muito usado quando estivermos falando do modelo de circuitos quânticos na computação quântica, pois em alguns casos estamos interessados em modificar o estado de apenas um ou de alguns dos qubits disponíveis.

2.2 Postulados da Mecânica Quântica

Na seção anterior, construímos as ferramentas necessárias para construir o palco em que a mecânica quântica se apresenta e para descrever como seus atores atuam, aqui daremos a interpretação física da matemática desenvolvida para representar sistemas quânticos.

O primeiro postulado da mecânica quântica, estabelece que associado a qualquer sistema físico isolado existe um vetor de estado que descreve completamente o sistema, este vetor possui norma igual a um e pertence a um espaço vetorial complexo com norma hermitiana, conhecido como espaço de Hilbert $\mathcal{H}$. Para visualizarmos a descrição matemática deste postulado, devemos considerar que um vetor que não pertença a base do espaço pode ser escrito como combinação linear dos vetores da base, sendo assim, considerando que $\{i\}$ formam uma base do espaço de Hilbert podemos escrever o vetor de estado de um sistema quântico $|\psi\rangle$ como

$$|\psi\rangle = \sum_i c_i |i\rangle;\ c_i \in \mathbb{C}, \tag{2.84}$$

onde os coeficientes c_i são números complexos que representam as coordenadas do vetor na base $\{|i\rangle\}$, mas que no contexto da física quântica são chamados de amplitude de probabilidade. No postulado é dito que os vetores de estado possuem norma um, então para verificarmos quais restrições esta condição coloca sobre as amplitudes de probabilidade c_i, devemos calcular a norma de $|\psi\rangle$ usando (2.5) e igualando a 1, fazendo isso obtemos

$$\begin{aligned} \langle\psi|\psi\rangle &= \sum_{i,j} c_j^* c_i \langle j|i\rangle;\ \delta_{ij} = \langle j|i\rangle \\ &= \sum_i |c_i|^2 = 1. \end{aligned} \tag{2.85}$$

O resultado acima é conhecido como condição de normalização dos vetores de estado. Uma informação importante sobre estados quânticos é que estes podem ter dois tipos de fase: global e relativa, a primeira não apresenta efeitos físicos e pode ser desconsiderada na normalização do estado. Um exemplo de estado com fase global pode ser visto abaixo

$$|\psi\rangle = e^{i\gamma}\left(\frac{1}{2}|\phi_1\rangle + \frac{\sqrt{3}}{2}|\phi_2\rangle\right), \tag{2.86}$$

onde $e^{i\gamma}$ representa a fase global. A fase relativa pode ser observada e possui importância para efeitos físicos, portanto esta não pode ser ignorada. Este tipo de fase é utilizada como recurso na computação quântica, um exemplo é o algoritmo de Grover [8], onde utiliza-se a ideia de um operador chamado oráculo que marca os itens que desejamos encontrar com uma fase relativa $e^{i\pi}$. Abaixo encontra-se um exemplo de estado que apresenta fase relativa

$$|\psi'\rangle = \frac{1}{\sqrt{2}}|\phi_1\rangle - \frac{i}{\sqrt{2}}|\phi_2\rangle, \tag{2.87}$$

onde a fase relativa é igual a $e^{-i\frac{\pi}{2}} = -i$.

O segundo postulado estabelece que a todo observável físico A, existe um operador linear Hermitiano cujos autovetores formam uma base completa. O postulado 3 estabelece uma relação entre as medidas e os autovalores do operador que representa o observável, sendo assim temos que: a medida de um observável A pode ser representada formalmente como a ação do operador que o representa, sobre o vetor de estado do sistema. O resultado dessa operação deve ser um único autovalor real a_n e a medida causará um colapso na função de onda do sistema, fazendo com que o seu vetor de estado seja, imediatamente a medida, igual ao autovetor $|\phi_n\rangle$ [32]. Os postulados 2 e 3 podem ser resumidos na seguinte equação

$$A|\phi\rangle = a_n|\phi_n\rangle, \tag{2.88}$$

onde $|\phi\rangle$ representa o estado do sistema físico antes da medida, $a_n = \langle\phi_n|\phi\rangle$ o valor medido do observável A e $|\phi_n\rangle$ o estado do sistema imediatamente após o colapso provocado pela medida.

O quarto postulado nos diz que a probabilidade de se obter o autovalor não degenerado a_n quando se mede o observável A de um estado $|\phi\rangle$ é igual a

$$p(a_n) = |\langle\phi|\phi_n\rangle|^2 = |a_n|^2, \tag{2.89}$$

mas se o autovalor a_n for degenerado, deve-se calcular a probabilidade a partir da expressão

$$p(a_n) = \sum_{i=1}^{k} |\langle\phi|\phi_n^k\rangle|^2 = \sum_{i=1}^{k} |a_n^{(k)}|^2. \tag{2.90}$$

O quinto e último postulado, estabelece como se dá a evolução do vetor de estado de um sistema quântico ao longo do tempo, este diz que a evolução de um sistema quântico fechado é governada pela equação de Schrödinger

$$i\hbar\frac{\partial|\psi(t)\rangle}{\partial t} = H(t)|\psi(t)\rangle, \tag{2.91}$$

onde $\hbar$ representa a constante de Planck dividida por 2π e $H(t)$ a Hamiltoniana[1] do sistema, esta é uma função que nos dá informações acerca das características do objeto em estudo. Este postulado também nos diz que a evolução de sistemas quânticos deve ser realizada por um operador unitário, pois assim temos a garantia que a norma dos vetores de estado é preservada, sendo assim o operador $U(t)$ deve apresentar a forma

$$U(t) = \mathcal{T}_{\leftarrow} e^{-\frac{i}{\hbar}\int_0^t H(t')dt'}, \tag{2.92}$$

[1] A princípio a Hamiltoniana de um sistema não necessita ter dependência temporal, mas para tornar a equação mais geral optamos por escrevê-la com uma Hamiltoniana que apresenta dependência do tempo.

onde $\mathcal{T}_{\leftarrow}$ representa o operador de ordenamento temporal. Para o caso especial onde a Hamiltoniana independe do tempo, o operador $U(t)$ se reduz a

$$U(t) = e^{-\frac{i}{\hbar}Ht}. \tag{2.93}$$

A evolução de um estado através da atuação do operador de evolução temporal é representada pela equação abaixo

$$|\psi(t)\rangle = U(t)|\psi(0)\rangle. \tag{2.94}$$

2.3 Spin - 1/2

Para fins de computação, buscamos por sistemas físicos que representem o sistema binário, um exemplo no contexto da mecânica quântica é o de spin-1/2. Neste caso além de ter a representação clássica binária através de *spin up* e *spin down*, temos de bônus a superposição de *spin up* e *spin down*, o que torna um sistema desse tipo um candidato natural a qubit. O objetivo aqui é apresentar os conceitos que envolvem esse assunto e deixar aberto o caminho para o desenvolvimento dessas ideias na computação quântica.

O spin é uma característica quântica de sistemas físicos descoberta em 1922 através do experimento de Stern-Gerlach [32, 33], o spin é algo intrinsecamente quântico e não possui análogo clássico. Eventualmente usamos analogias com objetos clássicos, como bússolas, para facilitar a visualização dos conceitos, esta analogia funciona bem, pois o spin também responde a aplicação de campos magnéticos externos. Representamos o spin s de uma partícula como o vetor $\vec{S}$

$$\vec{S} = S_x\hat{x} + S_y\hat{y} + S_z\hat{z}, \tag{2.95}$$

cujas componentes S_x, S_y e S_z obedecem a relação de comutação abaixo

$$[S_i, S_j] = i\hbar\varepsilon_{ijk}S_k;\ i, j \text{ e } k = x, y \text{ e } z, \tag{2.96}$$

onde ε_{ijk} representa o tensor de Levi-Civita

$$\varepsilon_{ijk} = \begin{cases} 1, \text{ se } ijk \text{ for uma permutação par de } x,\ y\ e\ z; \\ -1, \text{ se } ijk \text{ for uma permutação ímpar de } x,\ y\ e\ z; \\ 0, \text{ se qualquer um dos índices } i,\ j\ e\ k \text{ forem iguais.} \end{cases} \tag{2.97}$$

Os operadores S^2 e S_z obedecem a seguinte relação de comutação

$$[S^2, S_z] = 0, \tag{2.98}$$

e portanto podem ser diagonalizados na mesma base $\{|s, m_s\rangle\}$, onde s representa o spin da partícula e m_s o número quântico de spin, o segundo pode assumir os valores

$m_s = -s, -s+1, \ldots, s-1, s$. As atuações de S^2 e S_z sobre os vetores da base $\{|s, m_s\rangle\}$ produzem os resultados abaixo

$$S^2|s, m_s\rangle = \hbar^2 s(s+1)|s, m_s\rangle \tag{2.99}$$

$$S_z|s, m_s\rangle = \hbar m_s|s, m_s\rangle. \tag{2.100}$$

Também é possível definir os operadores de abaixamento S_- e levantamento de spin S_+, cujas atuações sobre os vetores da base produzem

$$S_\pm|s, m_s\rangle = \hbar\sqrt{s(s+1) - m_s(m_s \pm 1)}|s, m_s \pm 1\rangle, \tag{2.101}$$

sendo importante ressaltar que existem limites para aplicação desses operadores que é dado por

$$S_+|s, m_s = s\rangle = 0 \text{ e } S_-|s, m_s = -s\rangle = 0. \tag{2.102}$$

É possível escrever os operadores S_- e S_+ a partir de uma combinação linear entre as componentes S_x e S_y do vetor de spin $\vec{S}$, esta representação encontra-se abaixo

$$S_\pm = S_x \pm iS_y. \tag{2.103}$$

As definições dadas acima servem para um contexto mais geral, aqui estamos interessados em estudar apenas o caso em que $s = 1/2$, sendo assim temos a base de vetores igual a

$$\{|1/2, -1/2\rangle, |1/2, 1/2\rangle\}. \tag{2.104}$$

Dado que os vetores acima diagonalizam S^2 e S_z, temos que a atuação destes operadores sobre os estados da base nos remete a uma equação de autovalor, cujos resultados são iguais a

$$S^2|1/2, \pm 1/2\rangle = \frac{3\hbar^2}{4}|1/2, \pm 1/2\rangle \tag{2.105}$$

$$S_z|1/2, \pm 1/2\rangle = \pm\frac{\hbar}{2}|1/2, 1/2\rangle. \tag{2.106}$$

A partir desse ponto do texto vamos abandonar a notação $|s, m_s\rangle$ para os autovetores de S_z e vamos passar a adotar

$$|0\rangle \equiv |1/2, 1/2\rangle \ = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{2.107}$$

$$|1\rangle \equiv |1/2, -1/2\rangle \ = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{2.108}$$

pois essa nova notação é mais conveniente para a computação quântica, uma vez que representa os estados de spin para cima e para baixo como 0 e 1 do sistema binário, sendo assim passaremos a usar $\{|0\rangle, |1\rangle\}$ como base do espaço de spin $s = 1/2$.

Os principais operadores de spin podem ser escritos em termos da matriz identidade e de um conjunto de matrizes conhecido como matrizes de Pauli e possuem relação com o grupo de simetria SU(2) [34]. Posteriormente voltaremos a falar deste grupo quando definirmos a forma geral das portas da computação quântica

$$S^2 = \frac{3\hbar^2}{4}\mathbb{I},\ S_x = \frac{\hbar}{2}\sigma_x,\ S_y = \frac{\hbar}{2}\sigma_y \ e\ S_z = \frac{\hbar}{2}\sigma_z, \tag{2.109}$$

onde σ_x, σ_y e σ_z representam as seguintes matrizes

$$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \text{ e } \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \tag{2.110}$$

estas apresentam as seguintes propriedades:

$$\sigma_i^2 = \mathbb{I}; \tag{2.111}$$

$$\sigma_i^\dagger = \sigma_i; \tag{2.112}$$

$$\mathrm{tr}(\sigma_i) = 0; \tag{2.113}$$

$$[\sigma_i, \sigma_j] = 2i\varepsilon_{ijk}\sigma_k; \tag{2.114}$$

$$\{\sigma_i, \sigma_j\}_+ = 2\delta_{ij}\mathbb{I}. \tag{2.115}$$

De (2.114) e (2.115) podemos ver que é possível escrever qualquer uma das matrizes de Pauli como produto de outras duas, como por exemplo

$$\sigma_y = -i\sigma_x\sigma_z. \tag{2.116}$$

Esse tipo de relação será muito útil no contexto de computação quântica e de teoria quântica de correção de erros, pois vai nos permitir escrever tudo em função das bases de σ_x e σ_z. Os autovetores de σ_x ($|+\rangle$ e $|-\rangle$) e σ_y ($|+y\rangle$ e $|-y\rangle$) podem ser escritas como

$$|+\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} \text{ e } |-\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{2.117}$$

$$|+y\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ i \end{bmatrix} \text{ e } |-y\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -i \end{bmatrix}. \tag{2.118}$$

Podemos reescrever os vetores apresentados acima em função dos vetores da base computacional $|0\rangle, |1\rangle$ da seguinte forma

$$|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle) \tag{2.119}$$

$$|\pm y\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm i|1\rangle). \tag{2.120}$$

Posteriormente vamos apresentar a matriz que realiza a transferência da base computacional para a base conjugada (σ_x), ela é uma das portas mais importantes da computação quântica, pois nos permite criar superposições de estados e transformar matrizes de Pauli.

2.4 Matrizes de Densidade

As matrizes de densidade são usadas na mecânica quântica para descrever o estado de um sistema quântico que não é conhecido completamente, onde este pode ser descrito a partir de *ensembles* de estados quânticos que possam descrever o sistema. Podemos considerar a preparação de um estado quântico em laboratório como exemplo de onde isto se aplica, pois não temos garantia que vamos produzir apenas o estado de interesse, pode ser que por erro experimental ou por fatores externos outros estados sejam preparados, o que nos leva a ter certo grau de ignorância sobre as características do sistema e obter resultados diferentes do previsto. O exemplo dado não é algo distante, na verdade é bem comum no campo da computação quântica, pois para executar os algoritmos é preciso preparar os estados puros que serão usados na computação, mas pode ser que nesse processo ocorram falhas que façam com que outros estados também sejam preparados, configurando uma mistura de estados, o que no contexto da computação quântica pode nos levar a resultados incorretos.

Dada essa introdução e motivação do uso de matrizes de densidade no contexto da mecânica quântica, vamos buscar definir esses objetos. Para isso vamos começar considerando que temos o seguinte estado quântico

$$|\psi\rangle = \sum_i c_i |i\rangle, \tag{2.121}$$

e que queremos calcular o valor médio de um operador A através do estado acima, ou seja $\langle A\rangle_\psi = \langle\psi|A|\psi\rangle$. Em seguida, devemos introduzir relações de completeza na expressão, com isso podemos obter uma expressão para matrizes de densidade

$$\begin{aligned}
\langle A\rangle_\psi &= \langle\psi|A|\psi\rangle \\
&= \sum_{i,j} c_i c_j^* \langle i|\mathbb{I}A\mathbb{I}|j\rangle \\
&= \sum_{i,j} c_i c_j^* \langle i|\left(\sum_k |k\rangle\langle k|\right) A \left(\sum_l |l\rangle\langle l|\right)|j\rangle \\
&= \sum_{i,j,k,l} c_i c_j^* \langle i|k\rangle\langle l|j\rangle\langle k|A|l\rangle \\
&= \sum_{k,l} \langle l| \underbrace{\left(\sum_{i,j} c_i c_j^* |i\rangle\langle j|\right)}_{\equiv\rho} |k\rangle\langle k|A|l\rangle \\
&= \sum_l \langle l|\rho \underbrace{\left(\sum_k |k\rangle\langle k|\right)}_{=\mathbb{I}} A|l\rangle \\
&= \sum_l \langle l|\rho A|l\rangle = \mathrm{tr}(\rho A).
\end{aligned} \tag{2.122}$$

Note que usando matrizes de densidade o valor médio de A passa a ser calculado pelo traço do produto entre A e ρ. A matriz ρ obtida no cálculo acima define a forma geral

para matriz de densidade de estados puros

$$\rho = \sum_{i,j} c_i c_j^* |i\rangle\langle j|. \tag{2.123}$$

Quando estamos trabalhando com *ensembles* de estados quânticos, temos uma matriz de densidade com a seguinte forma

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|, \tag{2.124}$$

onde $|\psi_i\rangle\langle\psi_i|$ representa a matriz de densidade de estado puro relacionada ao estado $|\psi_i\rangle$ que possui uma probabilidade p_i de representar o sistema físico de interesse. Para que um operador ρ seja caracterizado como uma matriz de densidade é preciso que ele satisfaça as seguintes condições [1]

$$\mathrm{tr}(\rho) = 1 \quad \text{(Condição do traço)} \tag{2.125}$$

$$\rho \geq 0 \text{ (Operador positivo).} \tag{2.126}$$

É possível mostrar que as matrizes (2.123) e (2.124) satisfazem as condições acima. No caso de (2.123), mostra-se que a condição do traço é satisfeita calculando o valor médio do operador identidade

$$\begin{aligned} \langle \mathbb{I} \rangle_\psi &= \langle\psi|\mathbb{I}|\psi\rangle \\ &= \langle\psi|\psi\rangle = \mathrm{tr}(\rho\mathbb{I}) = \sum_i |c_i| = 1. \end{aligned} \tag{2.127}$$

Já para o caso de (2.124) verificamos a condição do traço realizando os seguintes cálculos

$$\begin{aligned} &\underbrace{\mathrm{tr}(\rho)}_{=1} = \mathrm{tr}\left(\sum_i p_i |\psi_i\rangle\langle\psi_i|\right) \\ &\sum_i p_i \underbrace{\mathrm{tr}(|\psi_i\rangle\langle\psi_i|)}_{=1} = 1 \\ &\sum_i p_i = 1. \end{aligned} \tag{2.128}$$

Supondo um estado arbitrário $|\phi\rangle$ podemos provar que a condição de ρ ser um operador positivo da seguinte maneira [1]

$$\begin{aligned} \langle\phi|\rho|\phi\rangle &= \sum_i p_i \langle\phi|\psi_i\rangle\langle\psi_i|\phi\rangle \\ &= \sum_i p_i |\langle\phi|\psi_i\rangle|^2 \\ &> 0, \end{aligned} \tag{2.129}$$

onde entendemos por operador positivo um operador cujos autovalores são não negativos.

As matrizes de densidade podem ser divididas em dois grupos, o primeiro daquelas que representam os estados puros, ou seja, estados que podem ser representados por (2.123)

e o segundo sendo o grupo dos estados mistos (2.124). O critério para dizer se uma matriz de densidade ρ representa um estado puro ou misto é dado pelo traço de ρ^2, se este for igual a 1 dizemos que o estado é puro e se for menor que 1 dizemos que o estado é misto, como representado abaixo

$$\mathrm{tr}(\rho^2)\begin{cases} = 1, & \text{estado puro} \\ < 1, & \text{estado misto} \end{cases}. \tag{2.130}$$

As matrizes de estados puros apresentam comportamento de um projetor, pois seu quadrado é igual a ela mesma, para visualizarmos isso podemos fazer a conta para (2.123), com isso vamos ver que de fato $\rho^2 = \rho$

$$\begin{aligned} \rho^2 &= \left(\sum_{i,j} c_i c_j^* |i\rangle\langle j|\right)\left(\sum_{k,l} c_k c_l^* |k\rangle\langle l|\right) \\ &= \sum_{i,j,k,l} c_i c_j^* c_k c_l^* |i\rangle\underbrace{\langle j|k\rangle}_{=\delta_{jk}}\langle l| \\ &= \underbrace{\sum_j |c_j|^2}_{=1} \sum_{i,l} c_i c_l^* |i\rangle\langle l|;\ l \ \to \ j \\ &= \sum_{i,j} c_i c_j^* |i\rangle\langle j| = \rho. \end{aligned} \tag{2.131}$$

É possível trabalhar com sistemas compostos usando o formalismo de matrizes densidade, para isso basta utilizar o produto tensorial para criar a nova matriz de densidade que irá representar o espaço expandido, por exemplo como a ρ_{AB} apresentada abaixo

$$\rho_{AB} = \rho_A \otimes \rho_B. \tag{2.132}$$

Nesse contexto pode ser que não tenhamos acesso aos graus de liberdade de um dos sistemas ou que não estejamos interessados neles, sendo assim é possível utilizar a operação conhecida como traço parcial para obter a matriz de densidade reduzida sem os graus de liberdade traçados, no caso de (2.132) os traços parciais ficam definidos como [1]

$$\rho_A = \mathrm{tr}_B(\rho_{AB}) \tag{2.133}$$

$$\rho_B = \mathrm{tr}_A(\rho_{AB}). \tag{2.134}$$

É importante ressaltar que o uso do traço parcial não é restrito a espaços de estados bipartidos, ou seja, essa função também está definida em espaços compostos por 3 subespaços ou mais.

Para calcularmos a entropia associada às matrizes de densidade de estados quânticos usamos a entropia de von Neumann, cuja definição é dada por [1, 38]

$$S(\rho) \equiv -\mathrm{tr}(\rho \log \rho), \tag{2.135}$$

onde log pode ser tomado em qualquer base, contudo para fins de computação o mais comum é usar o logaritmo na base 2. A expressão apresentada acima pode ser reescrita em termos dos autovalores λ_i de ρ

$$S(\rho) = -\sum_i \lambda_i \log \lambda_i. \tag{2.136}$$

Esta forma se assemelha a entropia de Shannon e é a mais utilizada para fins de cálculo da entropia.

2.4.1 Evolução de Matrizes de Densidade e Operações Quânticas

A evolução temporal de matrizes de densidade que representam sistemas quânticos fechados é descrita pela equação de Liouville-von Neumann [39]

$$\frac{d\rho(t)}{dt} = -\frac{i}{\hbar}[H(t), \rho(t)], \tag{2.137}$$

contudo os processos quânticos em geral são mais complexos e consideram interações com outros sistemas. Para descrevermos essas evoluções mais gerais precisamos definir as operações quânticas $\mathcal{E}$

$$\rho' = \mathcal{E}(\rho), \tag{2.138}$$

estas são mapas CPTP (*complete positive trace preserving*), ou seja, são mapas que levam uma matriz de densidade em outra, além disso preservam o traço e o caráter positivo do operador original. Considerando um sistema quântico fechado que é representado por uma matriz de densidade de estados mistos

$$\rho = \sum_i p_i \left|\psi_i\right\rangle\!\left\langle\psi_i\right|, \tag{2.139}$$

temos que a evolução de ρ pode ser dada em termos da atuação do operador de evolução U sobre os estados puros $|\psi_i\rangle$ presentes no *ensemble* que define o sistema em questão, sendo assim a evolução seria dada por

$$\sum_i p_i U \left|\psi_i\right\rangle\!\left\langle\psi_i\right| U^\dagger = U\rho U^\dagger = \rho', \tag{2.140}$$

onde neste caso a operação quântica $\mathcal{E}$ seria equivalente a

$$\mathcal{E}(\rho) = U\rho U^\dagger. \tag{2.141}$$

Modelamos como sistema quântico aberto um sistema quântico S acoplado a outro sistema quântico que em geral é chamado de ambiente, este pode ser modelado de várias formas a depender do contexto de interesse. Independente do modelo de ambiente a ser considerado, em geral supomos que o sistema e o ambiente juntos formam um sistema fechado. No caso em questão, a evolução de S dependerá da dinâmica interna entre o sistema e o ambiente,

como mostrado na Figura 13 (b), dependendo da interação U eles podem criar correlações quânticas fazendo com que se não possamos ter acesso a certas informações sobre S. A operação quântica que representa a evolução da matriz de densidade dos estados de S é dada por [1, 38]

$$\mathcal{E}(\rho) = \mathrm{tr}_{amb}\left[U(\rho_S \otimes \rho_{amb})U^\dagger\right], \tag{2.142}$$

onde assumimos que inicialmente S e o ambiente são um estado produto, ou seja, não possuem correlações quânticas. Além disso, também é assumido que após a operação U não ocorram mais interações entre o sistema e o ambiente, sendo assim tomamos o traço parcial em relação aos graus de liberdade do ambiente pois só queremos acompanhar a evolução de S. Considerando que a matriz de densidade relacionada ao ambiente é

$$\rho_{amb} = \sum_\mu \lambda_\mu \left|\mu\right\rangle\!\left\langle\mu\right|, \tag{2.143}$$

ou seja, um estado misto, então temos que a substituição dela em (2.142) produz

$$\begin{aligned}
\mathcal{E}(\rho) &= \sum_{\mu,\nu} \lambda_\mu \left\langle\nu\right| U \rho_S \otimes \left|\mu\right\rangle\!\left\langle\mu\right| U^\dagger \left|\nu\right\rangle \\
&= \sum_{\mu,\nu} \underbrace{\sqrt{\lambda_\mu}\left\langle\nu\right| U \left|\mu\right\rangle}_{\equiv K_{\mu\nu}} \rho_S \underbrace{\sqrt{\lambda_\mu}\left\langle\mu\right| U^\dagger \left|\nu\right\rangle}_{\equiv K^\dagger_{\mu\nu}} \\
&= \sum_{\mu,\nu} K_{\mu\nu}\rho_S K^\dagger_{\mu\nu},
\end{aligned} \tag{2.144}$$

onde $K_{\mu\nu}$ são os chamados operadores de Kraus que descrevem a dinâmica de S. Os operadores de Kraus contém toda a informação disponível sobre o estado inicial do ambiente e sobre a dinâmica da combinação sistema-ambiente, eles encapsulam todo o efeito do ambiente na matriz de densidade reduzida do sistema. Observe que estes operadores dependem da base escolhida para performar o traço parcial, mas são determinados univocamente pelo estado inicial do ambiente e pela Hamiltoniana do sistema conjunto sistema-ambiente [38]. Dado que a operação quântica $\mathcal{E}$ deve ser um mapa CPTP, então temos que

$$\begin{aligned}
&\mathrm{tr}\left[\mathcal{E}(\rho)\right] = 1 \\
&\mathrm{tr}\left[\sum_{\mu,\nu} K_{\mu\nu}\rho_S K^\dagger_{\mu\nu}\right] = 1,
\end{aligned} \tag{2.145}$$

e utilizando a propriedade cíclica do traço podemos reescrever a expressão acima como

$$\mathrm{tr}\left[\sum_{\mu,\nu} K^\dagger_{\mu\nu} K_{\mu\nu}\rho_S\right] = 1, \tag{2.146}$$

dessa forma podemos identificar que

$$\sum_{\mu,\nu} K^\dagger_{\mu\nu} K_{\mu\nu} = \mathbb{I}. \tag{2.147}$$

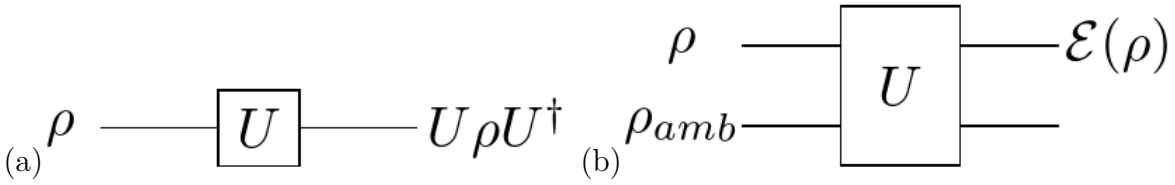


Figura 13 – (a) Modelo de evolução das matrizes densidade em sistemas quânticos fechados. (b) Modelo de evolução das matrizes de densidade em sistemas quânticos abertos.

O resultado acima está de acordo com a suposição de que o conjunto sistema-ambiente é um sistema fechado, pois os operadores de Kraus satisfazem a relação de completeza e portanto resultam em uma evolução unitária.

Outra abordagem possível para o estudo da dinâmica de sistemas quânticos abertos é através das equações mestras, estão são muito utilizadas no campo da óptica quântica. Neste cenário descrevemos a evolução temporal de um sistema aberto através de uma equação diferencial que descreve apropriadamente o comportamento não unitário da dinâmica. Em geral utilizamos a equação mestra conhecida como equação de Lindblad cuja forma é igual a [1, 38, 39]

$$\frac{d\rho}{dt} = -\frac{i}{\hbar}[H,\rho] + \sum_j \left(2L_j\rho L_j^\dagger - \{L_j^\dagger L_j,\rho\}_+\right), \tag{2.148}$$

onde L_j representam os operadores de Lindblad, H a Hamiltoniana do sistema e $\{.\}_+$ o anticomutador. Através da solução da equação mestra é possível representar o resultado como uma operação quântica, mas a recíproca não é verdade sempre, ou seja, em geral operações quânticas não podem ser realizadas necessariamente como uma equação mestra [1].

2.4.2 Fidelidade

2.4.2.1 Clássica

A fidelidade pode ser entendida como um tipo de medida da distância, não no sentido euclidiano, entre duas distribuições de probabilidade. No caso clássico temos que dada as distribuições de probabilidade $\{p_x\}$ e $\{q_x\}$, a fidelidade entre elas é definida como [1]:

$$F(p_x,q_x) \equiv \sum_x \sqrt{p_x q_x}. \tag{2.149}$$

A fidelidade é uma forma muito diferente de se medir a distância entre duas distribuições de probabilidade, pois primeiro ela não é uma métrica em si. Para ver este fato basta considerar o caso em que as distribuições de probabilidade $\{p_x\}$ e $\{q_x\}$ sejam idênticas, então de acordo com (2.149) temos

$$F(p_x,p_x) = \sum_x p_x = 1, \tag{2.150}$$

uma vez que os axiomas de Kolmogorov exigem que a distribuição de probabilidade seja normalizada a um [35]. Para uma métrica, esperamos que o resultado obtido para o caso discutido acima fosse nulo, o que não é verdade para (2.149). Entretanto, podemos dar uma interpretação geométrica para (2.149), nesse cenário ela pode ser entendida como o produto interno entre vetores com componentes $\sqrt{p_x}$ e $\sqrt{q_x}$ que estão sob uma esfera de raio unitário.

$$F(p,q) = \sqrt{p} \cdot \sqrt{q} = \cos\sigma, \tag{2.151}$$

onde σ é o ângulo entre os vetores.

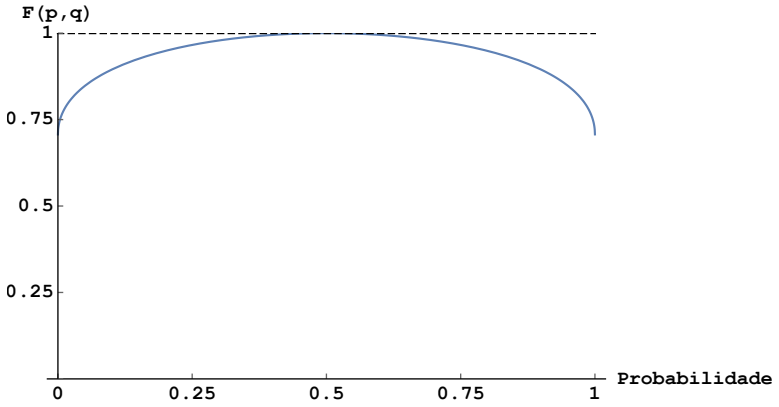


Figura 14 – Exemplo da fidelidade clássica entre as distribuições de probabilidade clássicas $\{1/2, 1/2\}$ e $\{p, 1-p\}$.

Para compreendermos melhor como usar essa ferramenta estatística, devemos discutir o seguinte exemplo: considere as distribuições de probabilidade (a) $\{1, 0\}$ e $\{1/2, 1/2\}$; (b)$\{1/2, 1/3, 1/6\}$ e $\{3/4, 1/8, 1/8\}$ e use (2.149) para determinar a distância entre elas, sendo assim substituindo os valores de (a) em (2.149)

$$F\left(\{1,0\},\{\frac{1}{2},\frac{1}{2}\}\right) = \sqrt{1\times\frac{1}{2}} + \sqrt{0\times\frac{1}{2}} = \sqrt{\frac{1}{2}} = 0,707106781.$$

O resultado acima nos mostra que a distância entre as distribuições de probabilidade é igual a $\sqrt{1/2}$. Do ponto de vista estatístico, podemos interpretar esse resultado como uma comparação entre as distribuições, de forma que as que apresentam um grau de fidelidade maior devem possuir maior grau de semelhança. Para as distribuições de (b) temos o seguinte resultado

$$F\left(\{\frac{1}{2},\frac{1}{3},\frac{1}{6}\},\{\frac{3}{4},\frac{1}{8},\frac{1}{8}\}\right) = \sqrt{\frac{1}{2}\times\frac{3}{4}} + \sqrt{\frac{1}{3}\times\frac{1}{8}} + \sqrt{\frac{1}{6}\times\frac{1}{8}} = 0,653529905.$$

Note que apesar das distribuições nos casos (a) e (b) serem diferentes, podemos afirmar que as do caso (b) estão mais distantes uma da outra do que as de (a), uma vez que apresentam um valor mais distante de um. O exemplo considerado aqui é o exercício 9.3 do capítulo 9 de [1].

2.4.2.2 Quântica

Na teoria quântica usamos a fidelidade como uma ferramenta para medir a distância entre dois estados quânticos. Assim como no caso clássico ela não é uma métrica de fato, contudo ela continua sendo uma ferramenta útil e importante para mensuração de qualidade de preparação de estados quânticos e de preservação da informação quântica em canais quânticos. A fidelidade dos estados ρ e σ é definida como [1]

$$F(\rho,\sigma) \equiv \mathrm{tr}\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}. \tag{2.152}$$

Existem dois casos especiais onde temos formas mais específicas para a expressão apresentada acima. O primeiro caso é quando as matrizes de densidade ρ e σ comutam, ou seja, quando são diagonalizadas na mesma base

$$\rho = \sum_i r_i |i\rangle\langle i| \text{ e } \sigma = \sum_i s_i |i\rangle\langle i|, \tag{2.153}$$

para alguma base ortonormal $|i\rangle$. Note que $|i\rangle\langle i| = (|i\rangle\langle i|)^2$, logo $(|i\rangle\langle i|)^{\frac{1}{2}} = |i\rangle\langle i|$, sendo assim nesse caso temos que

$$\begin{aligned} F(\rho,\sigma) &= \mathrm{tr}\sqrt{\sum_i r_i s_i |i\rangle\langle i|} \\ &= \mathrm{tr}\left(\sum_i \sqrt{r_i s_i}|i\rangle\langle i|\right) \\ &= \sum_i \sqrt{r_i s_i} = F(r_i, s_i). \end{aligned} \tag{2.154}$$

O resultado obtido acima nos diz que quando ρ e σ comutam, a fidelidade quântica se reduz ao resultado clássico.

O segundo exemplo que possui uma fórmula bem definida, é o caso em que estamos interessados em descobrir fidelidade entre um estado puro $|\psi\rangle$ e uma matriz de densidade de um estado arbitrário σ. Note que $|\psi\rangle$ foi dito um estado puro, portanto a sua matriz de densidade $\rho = |\psi\rangle\langle\psi|$ satisfaz a condição de pureza $\rho = \rho^2$, logo usando a definição (2.152) para o caso aqui descrito, temos que $\rho^{\frac{1}{2}}$ será igual a ρ, o que vai facilitar nossos cálculos. Sendo assim, temos que

$$\begin{aligned} F(|\psi\rangle, \sigma) &= \mathrm{tr}\sqrt{\langle\psi|\sigma|\psi\rangle|\psi\rangle\langle\psi|};\ \rho^2 = |\psi\rangle\langle\psi| \\ &= \mathrm{tr}\left[\sqrt{\langle\psi|\sigma|\psi\rangle}\,|\psi\rangle\langle\psi|\right] \\ &= \sqrt{\langle\psi|\sigma|\psi\rangle}. \end{aligned} \tag{2.155}$$

Isto é, neste caso a fidelidade é igual a raiz quadrada da sobreposição entre ρ e σ.

Assim como outros tipos de medidas de distância entre distribuições de probabilidade, a fidelidade também tem um conjunto de propriedades importantes que ela satisfaz,

vamos apresentar algumas no restante de nossa discussão sobre o tema. A primeira que deve ser comentada é que transformações unitárias deixam a fidelidade invariante, em termos matemáticos, temos

$$F\left(U\rho U^{\dagger}, U\sigma U^{\dagger}\right) = F(\rho,\sigma). \tag{2.156}$$

Além da propriedade discutida acima temos outras que valem para (2.152), contudo são complicadas de serem provadas nessa forma, então para visualizar que de fato são válidas é preciso fazer uso do teorema de Uhlmann [1]

$$F(\rho,\sigma) = \max_{|\psi\rangle,|\phi\rangle} |\langle\psi|\phi\rangle|, \tag{2.157}$$

que nos permite escrever a fidelidade como exposto acima, assim facilitando verificar que as seguintes propriedades são válidas para (2.152):

$$\text{Simetria} : F(\rho,\sigma) = F(\sigma,\rho); \tag{2.158}$$

$$\text{Se} \quad \rho = \sigma, \quad \text{então} \quad F(\rho,\sigma) = 1; \tag{2.159}$$

$$\text{Se} \quad \rho \neq \sigma, \quad \text{então} \quad F(\rho,\sigma) < 1; \tag{2.160}$$

$$\text{Se} \quad |\psi\rangle \quad \text{e} \quad |\phi\rangle \quad \text{são estados ortogonais, então} \quad F(\rho,\sigma) = 0. \tag{2.161}$$

Com as propriedades acima, fica mais fácil de ver que a fidelidade é um número que satisfaz a desigualdade

$$0 \leq F(\rho,\sigma) \leq 1. \tag{2.162}$$

2.4.2.3 Exemplo: Sistema de Dois Níveis Evoluindo no Tempo

Considere um sistema de dois níveis cuja Hamiltoniana não apresenta dependência explícita do tempo, sendo assim o operador evolução temporal desse sistema é dado por

$$U(t) = e^{-\frac{i}{\hbar}H_0 t}, \tag{2.163}$$

os estados que representam os níveis de energia são $|\psi_a\rangle$ e $|\psi_b\rangle$, estes são ortonormais ($\langle\psi_a|\psi_b\rangle = \delta_{ab}$). Esses estados são autoestados da hamiltoniana H_0, dessa maneira temos que a atuação da hamiltoniana produz o seguinte resultado

$$H_0 |\psi_a\rangle = E_a |\psi_a\rangle \text{ e } H_0 |\psi_b\rangle = E_b |\psi_b\rangle . \tag{2.164}$$

No instante de tempo $t = 0$, o sistema em questão encontra-se no seguinte estado

$$|\psi(0)\rangle = a |\psi_a\rangle + b |\psi_b\rangle , \tag{2.165}$$

depois ele passar a evoluir de acordo com o operador $U(t)$, tornando-se o estado abaixo

$$\begin{aligned} |\psi(t)\rangle &= U(t) |\psi(0)\rangle \\ &= a e^{-\frac{i}{\hbar}E_a t} |\psi_a\rangle + b e^{-\frac{i}{\hbar}E_b t} |\psi_b\rangle . \end{aligned} \tag{2.166}$$

O estado inicial $|\psi(0)\rangle$ e o estado que evolui no tempo $|\psi(t)\rangle$, devem ser normalizados e é fácil ver que satisfazem a condição de normalização

$$\langle\psi(0)|\psi(0)\rangle = \langle\psi(t)|\psi(t)\rangle = |a|^2 + |b|^2 = 1, \tag{2.167}$$

sendo assim, queremos descobrir como que a evolução temporal afeta a distribuição de probabilidade do sistema. Para descobrirmos o que ocorre vamos fazer uso da fidelidade, mas antes devemos determinar a matriz de densidade do estado inicial, pois a distribuição de probabilidade dele será a nossa referência. Realizando o produto externo de $|\psi(0)\rangle$ com ele mesmo, obtemos

$$\rho = |\psi(0)\rangle\langle\psi(0)| = |a|^2 |\psi_a\rangle\langle\psi_a| + ab^* |\psi_a\rangle\langle\psi_b| + ba^* |\psi_b\rangle|\psi_a\rangle + |b|^2 |\psi_b\rangle\langle\psi_b|. \tag{2.168}$$

Sabemos que a fidelidade entre dois estados puros pode ser escrita da seguinte forma

$$F\left(|\psi(t)\rangle, \rho\right) = \sqrt{\langle\psi(t)|\,\rho\,|\psi(t)\rangle}, \tag{2.169}$$

usando ρ calculada anteriormente e o estado $|\psi(t)\rangle$ na expressão acima, temos que realizar uma série de manipulações que nos levarão a

$$F(|\psi(t)\rangle, \rho) = \sqrt{|a|^4 + |a|^2|b|^2 e^{-\frac{i}{\hbar}(E_b - E_a)t} + |a|^2|b|^2 e^{-\frac{i}{\hbar}(E_a - E_b)t} + |b|^4}. \tag{2.170}$$

A expressão acima não nos ajuda muito, portanto devemos utilizar o truque de soma e subtração de um mesmo termo, aqui faremos isso com o termo $2|a|^2|b|^2$, pois com o termo em questão com sinal positivo seremos capazes de completar um produto notável, cuja soma de seus termos é igual a um. Fazendo as contas indicadas obtemos

$$\begin{aligned}
F(|\psi(t)\rangle, \rho) &= \sqrt{|a|^4 + |a|^2|b|^2 e^{-\frac{i}{\hbar}(E_b - E_a)t} + |a|^2|b|^2 e^{-\frac{i}{\hbar}(E_a - E_b)t} + |b|^4 \pm 2|a|^2|b|^2} \\
&= \sqrt{(|a|^4 + 2|a|^2|b|^2 + |b|^4) - 2|a|^2|b|^2 + |a|^2|b|^2 \left(e^{-\frac{i}{\hbar}(E_b - E_a)t} + e^{-\frac{i}{\hbar}(E_a - E_b)t}\right)} \\
&= \sqrt{(\underbrace{|a|^2 + |b|^2}_{=1})^2 - 2|a|^2|b|^2 + |a|^2|b|^2 (e^{\frac{i}{\hbar}(E_b - E_a)t} + e^{-\frac{i}{\hbar}(E_b - E_a)t})} \\
&= \sqrt{1 - 2|a|^2|b|^2 \left[1 - \cos\left[\frac{(E_b - E_a)t}{\hbar}\right]\right]};
\end{aligned} \tag{2.171}$$

O resultado acima já melhorou a situação, pois agora temos o número um e dado que a função cosseno em algum instante de tempo t irá assumir o valor um, temos garantia que o limite superior da fidelidade está garantido, contudo é possível realizar mais algumas manipulações no resultado obtido. Usando cosseno do arco duplo, ou seja, a expressão abaixo

$$\cos(2\alpha) = \cos^2(\alpha) - \sin^2(\alpha) \;\Rightarrow\; 2\sin^2(\alpha) = 1 - \cos(2\alpha), \tag{2.172}$$

finalmente chegamos a expressão final para a fidelidade entre os estados $|\psi(0)\rangle$ e $|\psi(t)\rangle$, note que o resultado abaixo é mais elegante e deixa mais claro que vamos ter os limites

superior e inferior respeitados

$$F(|\psi(t)\rangle, \rho) = \sqrt{1 - 4|a|^2|b|^2 \sin^2\left[\frac{(E_b - E_a)t}{2\hbar}\right]}. \tag{2.173}$$

Agora vamos testar o resultado obtido, primeiro testaremos a fidelidade no instante de tempo inicial, para isso temos que tomar $t = 0$, assim temos

$$F(|\psi(0)\rangle, \rho) = 1, \tag{2.174}$$

o que era esperado, pois em $t = 0$ temos $U(t = 0) = \mathbb{I}$. Portanto as distribuições devem ser idênticas no início, o curioso é que como a expressão depende de uma função seno ela será periódica, logo a fidelidade vai diminuir e em algum momento vai zerar, depois irá subir e retornar a um. Abaixo seguem todos os instantes de tempo em que a fidelidade assume o valor um

$$\sin^2\left[\frac{(E_b - E_a)t}{2\hbar}\right] = 0 \;\Rightarrow\; t_1 = \frac{2n\pi\hbar}{E_b - E_a};\; E_b \neq E_a,\; n \in \mathbb{N}. \tag{2.175}$$

Os instantes de tempo t em que a fidelidade se anula (quando os estados são ortogonais) ou quando atinge um valor mínimo diferente de zero, são dados por

$$t_{min} = \frac{2\hbar}{E_b - E_a} \arcsin\left(\pm\frac{1}{2|a||b|}\right), \tag{2.176}$$

com isso verificamos que os limites inferior e superior são satisfeitos. Retornando a questão do resultado ser uma função periódica, nos levanta um questionamento sobre a física do sistema, pois de alguma forma ele afasta os estados e depois os aproxima, ou seja, a evolução leva a um estado distinto do original e depois retorna ao estado original, tal comportamento pode ser observado na Figura 15. Alguns sistemas físicos podem se encaixar nesse cenário, mas lembrando que estamos considerando um sistema de dois níveis, o comportamento que estamos observando é dado justamente pela transição entre os dois níveis, pois em algum momento o sistema vai retornar ao seu nível de origem.

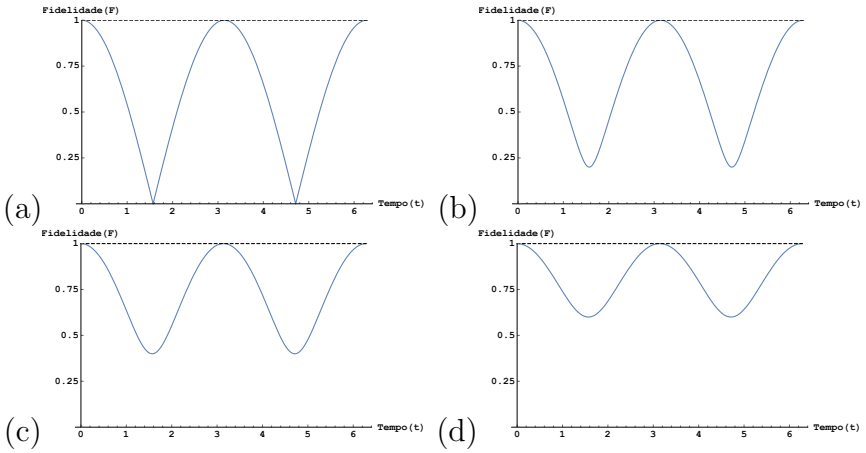


Figura 15 – Gráficos de (2.173) em função do tempo para diferentes valores de $|a|^2$ e $|b|^2$. Aqui é possível ver que só ocorre transição entre os estados no cenário do primeiro gráfico. (a) $|a|^2 = |b|^2 = 0.5$ (b) $|a|^2 = 0.6$ e $|b|^2 = 0.4$ (c) $|a|^2 = 0.7$ e $|b|^2 = 0.3$ (d) $|a|^2 = 0.8$ e $|b|^2 = 0.2$.

2.5 Estados Emaranhados

Os estados emaranhados passaram a virar tema de interesse da comunidade científica após a publicação do artigo de Einstein, Podolsky e Rosen em 1935 [40], desde então o assunto gerou diversos debates e nos fez questionar as concepções e ideias mais profundas acerca da realidade que experienciamos. Para Einstein, Podolsky e Rosen, a presença desse tipo de estado na mecânica quântica indicava que esta deveria ser uma teoria incompleta da realidade e que deveria existir algum tipo de variável oculta para corrigir esse problema. Em 1964, John Stewart Bell provou que teorias de variáveis ocultas locais devem ser incompatíveis com a mecânica quântica [41]. Em 1981, Alain Aspect e colaboradores verificaram experimentalmente as ideias propostas por Bell em seu artigo [42]. Este resultado nos mostrou que a mecânica quântica realmente desafia nossa visão clássica de mundo e provocou na comunidade científica um grande debate sobre as premissas de realismo (as propriedades físicas possuem valores definidos que existem independente de uma observação) e localidade (a medida em um sistema não deve apresentar nenhuma influência sobre o resultado da medida do outro) nas teorias físicas.

Na seção 2.1 vimos que é possível descrever o estado de um sistema físico composto por várias partes através do produto tensorial, sendo assim vamos considerar que queremos trabalhar com um sistema composto por duas cujos estados quânticos são

$$|\psi_A\rangle = \alpha|0\rangle + \beta|1\rangle \tag{2.177}$$

$$|\psi_B\rangle = \delta|0\rangle + \gamma|1\rangle. \tag{2.178}$$

Então realizando o produto tensorial desses estados para obter o estado do espaço de Hilbert do sistema como um todo, temos

$$|\psi_{AB}\rangle = |\psi_A\rangle \otimes |\psi_B\rangle = \alpha\delta|00\rangle + \alpha\gamma|01\rangle + \beta\delta|10\rangle + \beta\gamma|11\rangle. \tag{2.179}$$

A princípio podemos achar que neste espaço expandido temos apenas estados os estados da base $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ e a superposição destes, contudo o espaço resultante do produto tensorial é possui mais estados do que imaginamos inicialmente, pois existem estados que não podem ser descritos como produto tensorial, por exemplo os estados apresentados abaixo

$$|\psi_+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{2.180}$$

$$|\phi_+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \tag{2.181}$$

$$|\psi_-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \tag{2.182}$$

$$|\phi_-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle). \tag{2.183}$$

Vamos tomar como exemplo (2.180), observe que não é possível escrever este estado como (2.179), pois precisamos que $\alpha\delta$ e $\beta\gamma$ sejam iguais a $1/\sqrt{2}$, mas também é preciso que $\alpha\gamma$ e $\beta\delta$ sejam iguais a zero, porém isso nos leva a contradições, uma vez que se $\alpha\gamma = \beta\delta = 0$ então temos que nos dois casos uma das constantes é igual a 0 ou ambas, mas isso implicaria que $\alpha\delta$ e $\beta\gamma$ não seriam iguais a $1/\sqrt{2}$. Isso nos dá evidências de que existem estados em nosso novo espaço vetorial que não podem ser produzido por produtos tensoriais, esses estados são conhecidos como estados emaranhados e apresentam correlações quânticas que são mais fortes do que as conhecidas correlações clássicas que em geral possuem origem nas leis de conservação [38]. No formalismo de matrizes de densidade dizemos que um estado é emaranhado quando este não pode ser descrito como (2.132), por exemplo as matrizes de densidade dos estados emaranhados (2.180), (2.181), (2.182) e (2.183) são descritas pelas matrizes listadas abaixo

$$\rho_{\psi_+} = \frac{1}{2}(|00\rangle\langle 00| + |00\rangle\langle 11| + |11\rangle\langle 00| + |11\rangle\langle 11|) \tag{2.184}$$

$$\rho_{\phi_+} = \frac{1}{2}(|01\rangle\langle 01| + |01\rangle\langle 10| + |10\rangle\langle 01| + |10\rangle\langle 10|) \tag{2.185}$$

$$\rho_{\psi_-} = \frac{1}{2}(|00\rangle\langle 00| - |00\rangle\langle 11| - |11\rangle\langle 00| + |11\rangle\langle 11|) \tag{2.186}$$

$$\rho_{\phi_-} = \frac{1}{2}(|01\rangle\langle 01| - |01\rangle\langle 10| - |10\rangle\langle 01| + |10\rangle\langle 10|). \tag{2.187}$$

Um ponto importante sobre as matrizes apresentadas acima que merece destaque é que as matrizes densidade reduzidas resultantes da aplicação da operação traço parcial nelas são todas iguais a

$$\rho_A = \rho_B = \frac{1}{2}\mathbb{I}_{2\times 2}. \tag{2.188}$$

Um fato curioso é que todas as matrizes de densidade relacionadas aos estados emaranhados mencionados aqui representam estados puros, por exemplo $\mathrm{tr}(\rho^2_{\psi_+}) = 1$, o que indica que quando temos acesso a ambos os graus de liberdade não temos incerteza na descrição do

sistema. Contudo quando só possuímos acesso aos graus de liberdade de um dos sistemas, por exemplo a matriz de densidade reduzida apresentada acima, temos um estado misto ($\mathrm{tr}(\rho^2) < 1$), o que nos indica que sempre teremos algumas incerteza associada na descrição deste sistema, pois parte da informação que precisamos para descrevê-lo completamente está presente na outra parte do sistema composto. Outra forma de visualizarmos este fato é através da entropia de von Neumann (2.135). Se calcularmos ela para as matrizes de densidade dos estados emaranhados (2.180), (2.181), (2.182) e (2.183) vamos obter para todas elas o resultado $S = 0$, o que corrobora a ideia de não termos incerteza na descrição do sistema, mas quando tomamos o traço parcial em qualquer uma delas a entropia da matriz de densidade é igual a $S = 1$, o que neste caso simboliza máxima entropia e está relacionada a alto grau de incerteza na descrição do sistema. Aqui devemos separar as correlações quânticas e clássicas, as clássicas em geral surgem de leis de conservação em física e permitem realizar inferências. Por exemplo, considere que inicialmente temos uma partícula em repouso que decai em duas partículas, considerando que neste caso o momento linear se conserva, então ao medirmos o momento linear de uma das partículas produzidas no decaimento automaticamente saberemos as informações sobre o momento linear da outra [38]. Entretanto, no caso de estados emaranhados não podemos fazer esse tipo de inferência , pois neste caso sempre há um certo grau de ignorância sobre as informações de um dos sistemas de interesse que compõem o estado emaranhado, uma vez que parte da informação para descrevê-lo encontra-se no outro sistema. Apenas quando uma medida é realizada no estado quântico é que somos capazes de fazer inferência sobre o estado da outra parte, pois após a medida o sistema colapsa para um estado clássico.

3 Computação Quântica

"Nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy."
- Richard Phillips Feynman

"The theory of computation has traditionally been studied almost entirely in the abstract, as a topic in pure mathematics. This is to miss the point of it. Computers are physical objects, and computations are physical processes. What computers can or cannot compute is determined by the laws of physics alone, and not by pure mathematics."
- David Deutsch

3.1 Introdução

A computação quântica é um novo paradigma de computação que utiliza os efeitos quânticos como superposição de estados, emaranhamento, interferência de amplitudes de probabilidade e outros, para buscar ampliar o horizonte de classes de complexidade tratáveis. Esta abordagem surgiu naturalmente em um contexto em que o processo de miniaturização do transístor nos levou a ter que contabilizar efeitos quânticos para garantir pleno funcionamento dos computadores atuais. Além disso, existiam debates sobre a física da computação, ou seja, como processos físicos produziam os resultados da álgebra de Boole utilizada nas máquinas e também o inverso, como processos lógicos refletiam em processos dissipativos ou não dissipativos. Aliado a isso, existia uma discussão sobre os limites da computação clássica na simulação da natureza, uma vez que esta é produto da mecânica quântica e que devido a este fato torna-se um objeto complicado de ser simulado por máquinas clássicas sem recorrer a aproximações ou consumir muito tempo e recursos energéticos. Foi neste contexto que Richard Feynman e Paul Benioff escreveram as primeiras propostas de construção de computadores quânticos e suas possíveis aplicações.

3.2 Modelo de Circuitos Quânticos

A unidade fundamental da computação quântica é o qubit, estes podem ser representados por sistemas quânticos de dois níveis (por exemplo: uma partícula de spin 1/2), pois devem ter a capacidade de representar a base computacional clássica, $\{|0\rangle, |1\rangle\}$, e as superposições desses vetores, lembrando que a base computacional é representada por (2.107) e (2.108). Em geral, um estado que representa um qubit pode ser escrito em função

de dois ângulos θ e ϕ, como exposto abaixo:

$$|\psi(\theta,\phi)\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle;\ 0 \leq \theta \leq \pi;\ 0 \leq \phi < 2\pi. \tag{3.1}$$

A descrição dada pela a expressão acima nos permite mapear o estado do qubit sobre a superfície da conhecida esfera de Bloch (Figura 16), esta apresenta raio unitário, o que faz com que os estados dos qubits representados por (3.1) fiquem posicionados sobre a superfície da esfera. Se queremos encontrar o local da superfície em que o estado do qubit se encontra, devemos recorrer as coordenadas do vetor de Bloch, que podem ser calculadas usando os ângulos θ e ϕ a partir da seguinte expressão

$$\vec{n} = (\cos\phi\sin\theta, \sin\phi\sin\theta, \cos\theta), \tag{3.2}$$

onde as coordenadas representam os valores esperados das matrizes de Pauli, calculados usando o vetor (3.1) ou pelas expressões

$$\langle\sigma_x\rangle = \mathrm{tr}(\sigma_x\rho) \tag{3.3}$$

$$\langle\sigma_y\rangle = \mathrm{tr}(\sigma_y\rho) \tag{3.4}$$

$$\langle\sigma_z\rangle = \mathrm{tr}(\sigma_z\rho), \tag{3.5}$$

onde ρ representa a matriz densidade de estados mistos. Vale ressaltar que essas quantidades podem ser usadas para realizar uma tomografia de estado quântico de um qubit. Um aspecto importante da esfera de Bloch que vale a pena ser salientado, é que podem surgir erros de interpretação dos resultados devido ao fato de que estados ortogonais encontrarem-se sobre um mesmo eixo, dando a impressão de que seriam paralelos ou antiparalelos. No contexto de estados puros, essa representação é útil para visualizar como o vetor de estado evolui com as aplicações das portas quânticas, mas seu uso não se restringe a apenas isso, pois podemos utilizá-la para entender se temos um estado puro ou misto, uma vez que os estados puros estão espalhados sobre a superfície da esfera, já os mistos se encontram em pontos internos da esfera, indicando um aumento da entropia do sistema.

Os postulados da mecânica quântica de sistemas fechados, estabelecem que os operadores que atuam sobre os estados quânticos devem ser unitários, para que a norma dos vetores de estado sejam preservadas, portanto para executarmos as portas da computação quântica, precisamos garantir que os operadores que as representam satisfaçam essa condição. Tendo isso em vista e que um qubit é representado por um sistema de dois níveis, sabemos que qualquer porta que atue apenas sobre um qubit, pode ser representada pela matriz geral do grupo de simetria SU(2) com valores específicos para os parâmetros θ, ϕ e λ

$$U(\theta,\phi,\lambda) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{i\lambda}\sin\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) & e^{i(\phi+\lambda)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}. \tag{3.6}$$

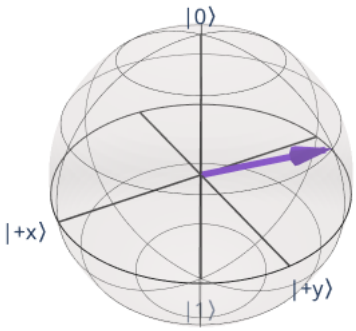


Figura 16 – Representação do estado de um qubit na esfera de Bloch, com $\theta = \pi/3$ e $\phi = 2\pi/3$.

Com isso fica garantido que os operadores que performarão as portas quânticas satisfazem as condições necessárias para garantir a unitariedade. É válido ressaltar que a equação (3.1), pode ser facilmente obtida através da atuação de (3.6) sobre o estado $|0\rangle$. As seguintes escolhas de parâmetros para (3.6) nos levam as matrizes de Pauli $\sigma_x \equiv X$, $\sigma_y \equiv Y$ e $\sigma_z \equiv Z$[1]

$$X = U(\pi, 0, \pi),\ Y = U(\pi, \pi/2, \pi/2) \text{ e } Z = U(0, \pi, 0). \tag{3.7}$$

A porta X é equivalente ao NOT da computação clássica, ou seja, ela realiza um *bit-flip*, levando isso em conta temos que a atuação dessa porta sobre os estados da base computacional produz o resultado clássico

$$X|0\rangle = |1\rangle \text{ e } X|1\rangle = |0\rangle, \tag{3.8}$$

mas como os computadores quânticos podem trabalhar com superposições dos estados $|0\rangle$ e $|1\rangle$, então a ação dessa porta sobre (3.1) produz

$$X|\psi(\theta, \phi)\rangle = \cos\left(\frac{\theta}{2}\right)|1\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|0\rangle. \tag{3.9}$$

Note que o resultado foi uma inversão das amplitudes de probabilidade dos $|0\rangle$ e $|1\rangle$, isto implicaria em uma mudança no resultado das medições feitas para finalizar o processo de computação quântica.

Outras portas aparecem exclusivamente no contexto quântico, como por exemplo a porta Z, cuja atuação sobre os estados da base computacional produz

$$Z|0\rangle = |0\rangle \text{ e } Z|1\rangle = -|1\rangle. \tag{3.10}$$

[1] A partir deste ponto do texto vamos usar a notação alternativa introduzida aqui, pois esta é mais conveniente para as contas e mais usada na construção de códigos quânticos de correção de erros.

A princípio pode parecer que essa porta não fornece nada de novo e útil, pois ela apenas adiciona uma fase global ao estado $|1\rangle$, que sabemos que não possui relevância física, contudo se pensarmos na aplicação de Z sobre um estado que é uma superposição dos estados da base computacional, neste caso ela adiciona uma fase relativa que modifica o estado, por exemplo

$$\begin{aligned} |\phi\rangle &= \alpha|0\rangle + \beta|1\rangle \\ Z|\phi\rangle &= \alpha Z|0\rangle + \beta Z|1\rangle = \alpha|0\rangle - \beta|1\rangle = |\phi'\rangle. \end{aligned} \tag{3.11}$$

Note que os estados $|\phi\rangle$ e $|\phi'\rangle$ são ortogonais. Ainda considerando as matrizes de Pauli, temos uma terceira porta baseada nelas e que só aparece no contexto quântico, esta é a porta Y que apresenta a seguinte atuação sobre os estados da base computacional

$$Y|0\rangle = -i|1\rangle \text{ e } Y|1\rangle = i|0\rangle. \tag{3.12}$$

Perceba que Y pode ser implementada usando as portas X e Z usando a relação (2.116) aliada a uma porta P que será apresentada mais à frente, esta com o parâmetro adequado adiciona a fase que aparece em (2.116). Existem outras relações que reproduzem o comportamento de Y, o importante é saber que mesmo sem contar com esta porta no hardware é possível reproduzir os resultados produzidos por ela se dispusermos das portas presentes na relação. É importante dizer que Y e Z são estritamente quânticas, pois trabalham diretamente com fases globais e relativas que são características que só aparecem nos qubits.

Um computador quântico precisa produzir estados em superposição, caso contrário estaria restrito ao que já conhecemos da computação clássica, além disso em alguns processos quânticos a superposição pode desaparecer propositalmente ou não, mas uma vez que isso pode acontecer devemos dispor de mecanismos para produzir essa importante característica da mecânica quântica. Usando (3.6) com os parâmetros $\theta = \pi/2$, $\phi = 0$ e $\lambda = \pi$, definimos uma porta que é conhecida como porta Hadamard e é denotada por H

$$U(\pi/2, 0, \pi) = H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{3.13}$$

Esta porta apresenta a característica desejada que foi descrita no início do parágrafo. A porta Hadamard também apresenta algumas características interessantes, por exemplo, ela pode ser vista como a combinação linear entre as portas X e Z

$$H = \frac{1}{\sqrt{2}}(X + Z), \tag{3.14}$$

além disso ela é um operador Hermitiano

$$H^\dagger = H. \tag{3.15}$$

A atuação da porta Hadamard sobre os estados $|0\rangle$ e $|1\rangle$ produz superposições onde ambos os estados possuem a mesma probabilidade de serem medidos, a diferença é que os estados obtidos apresentam fases relativas distintas

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle \tag{3.16}$$

$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle. \tag{3.17}$$

Outro ponto importante das relações apresentadas acima, é que ela demonstra que a porta Hadamard leva os estados da base computacional para os estados da base conjugada e uma vez que H é sua própria inversa, temos que a atuação desta porta sobre os estados $|+\rangle$ e $|-\rangle$ recupera a base computacional

$$H|+\rangle = |0\rangle \tag{3.18}$$

$$H|-\rangle = |1\rangle. \tag{3.19}$$

Esse comportamento de transferência de base, faz com que seja possível produzir a porta X combinando portas Z e H e vice-versa da seguinte forma

$$X = HZH \tag{3.20}$$

$$Z = HXH. \tag{3.21}$$

Essas relações conferem comportamento similares das portas em suas respectivas bases, pois note que a atuação de Z sobre os estados $|+\rangle$ e $|-\rangle$

$$Z|+\rangle = |-\rangle \tag{3.22}$$

$$Z|-\rangle = |+\rangle, \tag{3.23}$$

nada mais é do que uma espécie de *bit-flip* na base conjugada. Através das portas X, Y e Z podemos definir portas que realizam rotações nos vetores de estados dos qubits. Para isso é necessário levar em consideração a exponencial dessas matrizes, a expansão da função exponencial em série de Taylor e as propriedades das matrizes de Pauli, dessa forma devemos obter os seguintes operadores

$$R_X(\theta) = e^{-i\theta\frac{X}{2}} = \cos\left(\frac{\theta}{2}\right)\mathbb{I} - i\sin\left(\frac{\theta}{2}\right)X = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix} \tag{3.24}$$

$$R_Y(\theta) = e^{-i\theta\frac{Y}{2}} = \cos\left(\frac{\theta}{2}\right)\mathbb{I} - i\sin\left(\frac{\theta}{2}\right)Y = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix} \tag{3.25}$$

$$R_Z(\theta) = e^{-i\theta\frac{Z}{2}} = \cos\left(\frac{\theta}{2}\right)\mathbb{I} - i\sin\left(\frac{\theta}{2}\right)Z = \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}. \tag{3.26}$$

Note que quando o argumento dos operadores de rotação apresentados acima é igual a π, recuperamos os operadores X, Y e Z multiplicados por uma fase, que não afeta a

fidelidade da porta quando comparada com a porta sem fase, portanto a atuação deve resultar no mesmo resultado que a forma original da porta.

Uma vez que o qubit apresenta fase relativa além do comportamento de dualidade presente nos bits, podemos utilizá-la como recurso computacional e para isso necessitamos de portas que possam adicionar e transformar fases relativas. A primeira a ser apresentada aqui é a porta fase, cuja matriz é igual a

$$P(\varphi) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}. \tag{3.27}$$

Note que a forma da matriz acima indica que ela produz efeito apenas sobre o estado $|1\rangle$. É válido ressaltar que o Hermitiano conjugado da porta P é equivalente a usá-la com argumento negativo, ou seja, $P^\dagger(\varphi) = P(-\varphi)$. Outra porta importante relacionada a fases é a porta S, esta nada mais é do que a raiz quadrada da porta Z

$$S = \sqrt{Z} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \tag{3.28}$$

este fato faz com que a porta S adicione uma fase relativa $e^{i\frac{\pi}{2}} = i$ ao estado $|1\rangle$. O Hermitiano conjugado de S também é uma porta de fase e é dada pela matriz

$$S^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}, \tag{3.29}$$

onde $SS^\dagger = \mathbb{I}$. As raízes quadradas de S e $S^\dagger$ também definem portas que adicionam fase, estas são conhecidas como T e $T^\dagger$ e suas matrizes são iguais a

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{bmatrix} \tag{3.30}$$

$$T^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-i\frac{\pi}{4}} \end{bmatrix}. \tag{3.31}$$

A princípio pode não parecer necessário usar S e T uma vez que dispomos de P, pois bastaria usá-la com $\varphi = \pi/2$ e $\varphi = \pi/4$, mas como será apresentado mais adiante as portas S e T fazem parte de um conjunto que define um conjunto de computação quântica universal.

Os computadores quânticos da IBM utilizam o conjunto $\{U_{CNOT},\ \mathbb{I},\ R_Z(\theta),\ \sqrt{X},\ X\}$ como portas de base para execução de algoritmos quânticos através de pulsos de microondas. A matriz que representa a porta $\sqrt{X}$ é igual a

$$\sqrt{X} = \frac{1}{2}\begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix}. \tag{3.32}$$

As portas apresentadas até agora se restringiam a atuar apenas sobre um qubit, mas sabemos que é possível trabalhar com mais de um qubit e para isso devemos usar o

produto tensorial. Na computação clássica temos algumas portas que atuam sobre mais de um bit que podem ser aproveitadas no contexto quântico pois são operadores unitários, por exemplo a porta CNOT, cuja representação na notação de Dirac é dada por

$$U_{CNOT} = |0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes X. \tag{3.33}$$

Os operadores de projeção $|0\rangle\langle 0|$ e $|1\rangle\langle 1|$ atuam sobre o qubit de controle, já os operadores $\mathbb{I}$ e X atuam sobre o qubit alvo. Uma representação alternativa para a NOT controlada é dada por [36]

$$\Lambda_{(c,t)}(X) = U_{CNOT}, \tag{3.34}$$

onde fica mais evidente quem é o qubit de controle c e o qubit alvo t. Existem diversos tipos de portas controladas além da NOT na computação quântica, um exemplo simples que aparece de forma natural é a porta Z controlada, que é mais comumente denominada como CZ e sua representação na notação de Dirac é similar a da CNOT

$$U_{CZ} = |0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes Z, \tag{3.35}$$

a expressão acima é igual a seguinte matriz

$$U_{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}. \tag{3.36}$$

Assim como no caso da CNOT, podemos usar a representação Λ para a porta CZ

$$\Lambda_{(c,t)}(Z) = U_{CZ}. \tag{3.37}$$

A representação Λ nos auxilia a provar as seguintes identidades de circuito [36]

$$\Lambda_{(c,t)}(X) X_c \Lambda_{(c,t)}(X) = X_c X_t \tag{3.38}$$

$$\Lambda_{(c,t)}(X) X_t \Lambda_{(c,t)}(X) = X_t \tag{3.39}$$

$$\Lambda_{(c,t)}(X) Z_c \Lambda_{(c,t)}(X) = Z_c \tag{3.40}$$

$$\Lambda_{(c,t)}(X) Z_t \Lambda_{(c,t)}(X) = Z_c Z_t, \tag{3.41}$$

estas são úteis para realizar a conversão dos estabilizadores da teoria quântica de correção de erros para o conjunto de portas que definem o grupo de Clifford

$$\{H, S, \text{CNOT}\}, \tag{3.42}$$

que é importante, pois é um exemplo de conjunto de portas que podem ser simuladas de maneira eficiente em um computador clássico de acordo com o que é estabelecido no teorema de Knill-Gottesman [1, 36]. O grupo de Clifford, não forma um conjunto universal

de portas para a computação quântica, entretanto se adicionarmos a porta T ao conjunto do grupo de Clifford, este se tornará universal, mas a um custo de deixar de ser simulado de maneira eficiente em máquinas clássicas. Sendo assim, temos que um conjunto possível para a computação quântica universal é igual a

$$\{H, S, \text{CNOT}, T\}. \tag{3.43}$$

Com esse conjunto de portas, também é possível reproduzir todas as portas clássicas reversíveis, com isso podemos tomar a computação clássica como um limite da computação quântica. Na Figura 17 temos alguns exemplos de implementações de portas clássicas na computação quântica. Através das relações (3.20) e (3.21) podemos converter uma porta

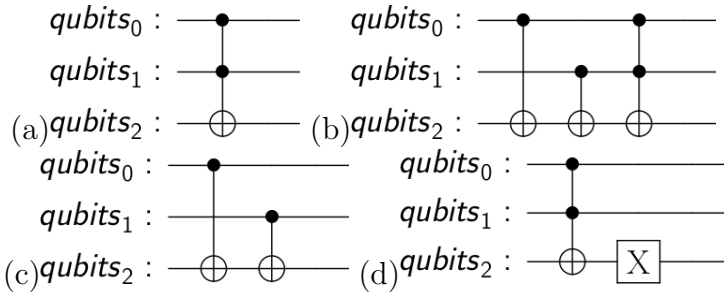


Figura 17 – Circuitos que representam as portas clássicas: (a) AND; (b) OR; (c) XOR e (d) NAND.

CNOT em CZ e vice-versa, a equação exposta abaixo demonstra como realizar a conversão

$$U_{CZ} = (\mathbb{I} \otimes H) U_{CNOT} (\mathbb{I} \otimes H). \tag{3.44}$$

Dependendo do tipo de qubit e arquitetura utilizados na construção do *hardware*, pode acontecer de um qubit não apresentar comunicação direta com outro, isso poderia ser um impedimento à execução de alguns algoritmos, porém existe um mecanismo de troca de estado de qubits através de uma combinação de três portas CNOT. Isto permite com que qubits que não "conversam" diretamente possam se comunicar através da troca de estado com um qubit intermediário, este arranjo de portas CNOT é chamado de porta SWAP e seu operador é descrito como

$$U_{SWAP} = \Lambda_{(c,t)}(X) \Lambda_{(t,c)}(X) \Lambda_{(c,t)}(X). \tag{3.45}$$

Note que a primeira porta CNOT está no sentido original de controle para alvo, mas a segunda realiza uma inversão do sentido, este procedimento é necessário para que os estados do controle e alvo sejam trocados, para finalizar o SWAP precisamos realizar outra CNOT no sentido original. A matriz que representa o operador U_{SWAP} é igual a

$$U_{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.46}$$

As representações gráficas da porta SWAP podem ser vistas na Figura 18. É importante ressaltar que na atual era da computação quântica, ou seja, dispositivos NISQ, a profundidade do circuito (maior caminho percorrido por um qubit dentro de um circuito) e o número de portas CNOT presente nele possuem um alto impacto qualidade do resultado final da computação. Portanto, quando estamos desenvolvendo algoritmos que vão ser executados nos *hardwares* atuais é preciso se atentar ao mapa de acoplamento dos qubits presentes no dispositivo, pois se buscarmos utilizar os qubits da melhor forma em termos de conexões, podemos minimizar a introdução de erros durante o processo de computação.

Assim como a porta CNOT, a porta Toffoli pode ser importada diretamente da computação clássica. Isto nos confere a habilidade de reconstruir todas as portas clássicas no contexto da computação quântica, como mostrado na Figura 17 e também possibilita a utilização em novos cenários, por exemplo em oráculos e amplificadores de amplitude do circuito que performa o algoritmo de Grover [8]. A forma de operador da porta Toffoli na notação de Dirac é igual a

$$U_{Toffoli} = |00\rangle\langle 00| \otimes \mathbb{I} + |11\rangle\langle 11| \otimes X, \tag{3.47}$$

cuja matriz é expressa da seguinte forma

$$U_{Toffoli} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{3.48}$$

É importante dizer que é possível expandir a quantidade de qubits de controle da porta Toffoli caso o circuito de interesse exija uma quantidade maior que três. Apesar de podermos representar a porta Toffoli de maneira simples usando operadores unitários, isto não significa que ela possui uma implementação direta no *hardware*, portanto pode ser que sua forma final se diferencie da forma apresentada na Figura 10, por exemplo nos dispositivos da IBM a porta Toffoli é igual a sequência de portas apresentadas na Figura 19.

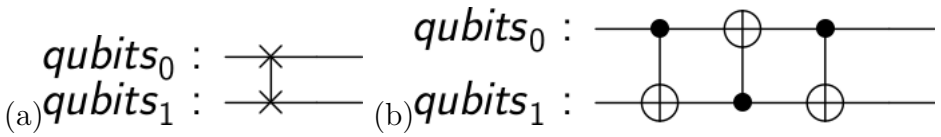


Figura 18 – Representações gráficas da porta SWAP: (a) convencional e (b) alternativa.

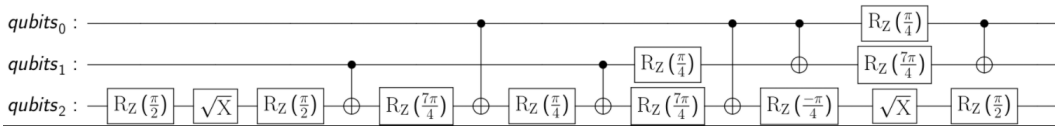


Figura 19 – Circuito quântico que performa a porta Toffoli nos computadores quânticos da IBM.

Dando continuidade a linha de portas que atuam sobre mais de um qubit, temos também as rotações em mais de um qubit, como exemplificado abaixo

$$R_{X_1X_2}(\theta) = e^{-i\theta\frac{X_1\otimes X_2}{2}} = \cos\left(\frac{\theta}{2}\right)\mathbb{I}_1\otimes\mathbb{I}_2 - i\sin\left(\frac{\theta}{2}\right)X_1\otimes X_2. \tag{3.49}$$

Outras combinações de matrizes de Pauli podem ser usadas nas rotações mais gerais, a escolha de $X_1 \otimes X_2$ foi apenas para exemplificação. A importância de portas como CNOT, CZ, R_{XX} e outras, é que elas realizam o papel de estabelecer uma interação local entre os qubits, de maneira a permitir que em algum momento estes possam criar emaranhamento que pode ser utilizado para diversas tarefas no campo da informação quântica, mais à frente no texto veremos como o emaranhamento pode ser usado para transmitir bits clássicos e estados quânticos arbitrários.

Agora que conhecemos os principais componentes de um circuito quântico, ou seja, qubits e portas, estamos aptos a entender o processo de como montar um circuito. Inicialmente, devemos preparar os estados quânticos que serão usados na computação, lembrando que todos os qubits presentes em um circuito quântico são inicializados como um $|0\rangle$, por exemplo: considere que estamos trabalhando com um circuito com n qubits, portanto o estado inicial que representa esses qubits é igual a

$$|\psi_i\rangle = |\underbrace{000\ldots0}_{n \text{ qubits}}> . \tag{3.50}$$

Os qubits são representados por linhas no circuitos como mostrado na Figura 20, a estas linhas devemos ligar as portas para transformar o estado de computação que será utilizado. Nesta seção, vimos algumas das principais portas da computação quântica que atuam sobre um qubit ou em mais de um, nas Figuras 21 e 22 temos as representações gráficas das portas aqui apresentadas. Combinando vários operadores unitários, podemos criar circuitos como o retratado na Figura 23. Os circuitos nada mais são do que uma representação gráfica da aplicação de operadores sobre um estado quântico de interesse, ou seja, considerando que o estado inicial seja $|\psi_i\rangle = |00\rangle$, então o circuito da Figura 23 representa a seguinte expressão

$$|\psi_f\rangle = U_nU_{n-1}\ldots U_3U_2U_1|\psi_i\rangle. \tag{3.51}$$

Note que há uma diferença entre a expressão acima e a representação em circuito quântico, pois lê-se o circuito da esquerda para a direita, enquanto que em (3.51) a ordem de

aplicação dos operadores é da direita para à esquerda. Em um cenário ideal, no final de um processo de computação devemos obter o estado quântico que possui a informação de interesse, sendo assim precisamos medí-lo, mas uma única medição não basta, dado o caráter probabilístico intrínseco da computação quântica. Sendo assim, se faz necessário realizar várias medidas para obter a distribuição de probabilidade dos estados e por consequência a informação de interesse. É válido ressaltar que por padrão a porta que representa a medida na base de autovetores de σ_z, ou seja, a base computacional $\{|0\rangle, |1\rangle\}$, contudo é possível realizar medições em outras bases. Por exemplo se quisermos medir o estado na base de autovetores de σ_x é preciso adicionar uma porta Hadamard antes da medida, mas se quisermos medir na base de autovetores de σ_y, precisamos adicionar uma porta $S^\dagger$ antes da porta de medida. Outro ponto relevante, é que a porta de medida é ligada a bits clássicos, estes são representados por duas linhas paralelas e exercem o papel de memória responsável por armazenar o resultado da medida realizada no estado quântico.

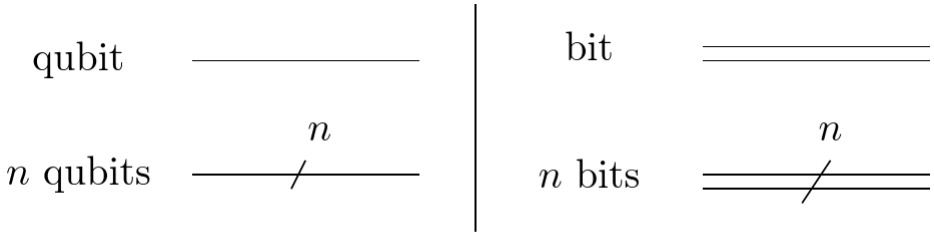


Figura 20 – Convenção das representações de qubits e bits em um circuito quântico.

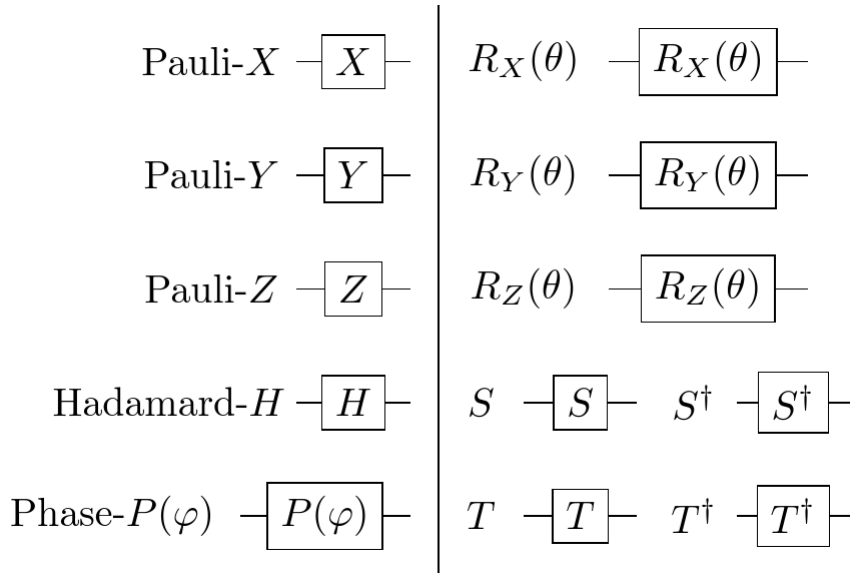


Figura 21 – Representações gráficas de algumas das portas que atuam sobre um qubit em um circuito quântico.

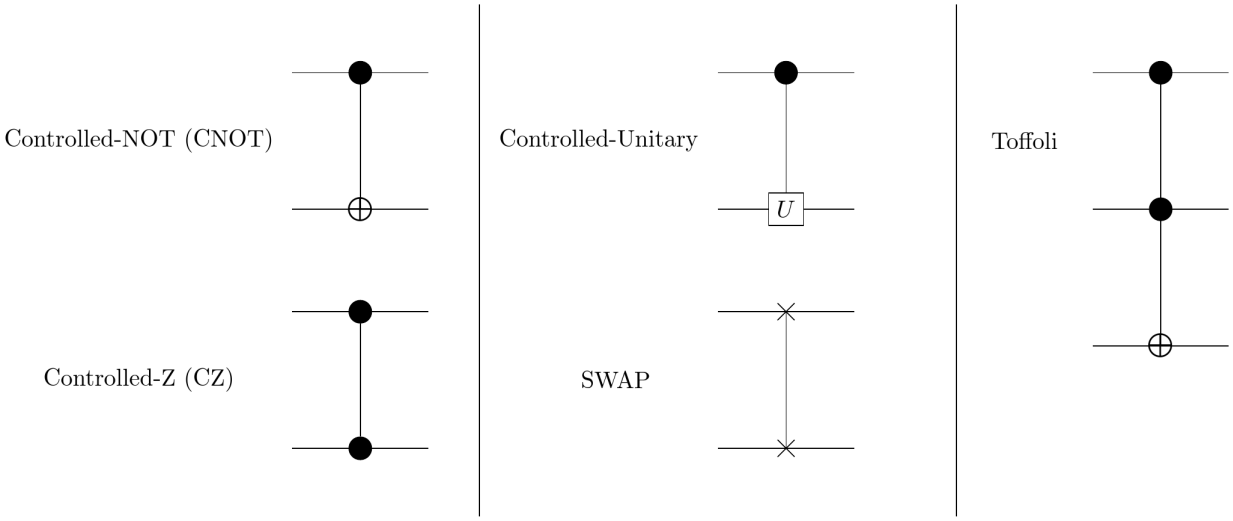


Figura 22 – Representações gráficas de algumas das portas de controle que aparecem em circuitos quânticos.

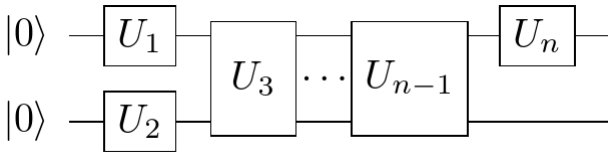


Figura 23 – Exemplo de uma sequência de aplicações de operadores que formam um circuito.

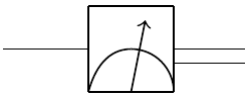


Figura 24 – Representação gráfica da porta de medida em um circuito quântico.

3.3 Teorema da Não Clonagem

Em nosso mundo clássico estamos acostumados a fazer cópias de arquivos, músicas, fotos e etc, ou seja, conseguimos copiar a forma que estão codificados na base binária para serem armazenados ou reproduzidos em outros dispositivos. A teoria quântica estabelece limites para a criação de uma copiadora universal de estados quânticos, entenda copiadora universal como um dispositivo que poderia copiar qualquer estado quântico arbitrário que é dado como entrada. Essa restrição afeta diretamente a construção de códigos quânticos de correção de erros, pois nos impede de usar a ideia clássica de utilizar cópias para criar redundância da mensagem visando protegê-la contra erros. Dessa forma, é preciso encontrar outros meios de tratar problemas que usavam cópias como parte da solução.

Aqui devemos mostrar como os postulados da mecânica quântica nos levam ao que chamamos de teorema da não clonagem, a prova a ser dada se baseia em contradição[2],

[2] A prova feita aqui usa como argumento a linearidade dos operadores, mas existem outras abordagens,

ou seja, vamos partir do princípio que existe um operador U que atue como copiador universal e que a atuação dele contradiz o que entendemos por cópia de estado quântico. Para realizar essa demonstração vamos considerar os seguintes estados que vão sofrer a ação de U

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \;\text{ e }\; |0\rangle\,, \tag{3.52}$$

com $|\psi\rangle$ sendo um estado qualquer que queremos copiar e $|0\rangle$ o estado que irá receber a informação do estado $|\psi\rangle$. O operador U cria uma cópia da informação do primeiro qubit e cola no segundo, mantendo a informação do primeiro intacta, podemos visualizar esse tipo de operação através do exemplos abaixo

$$U\,|0\rangle \otimes |0\rangle = |0\rangle \otimes |0\rangle = |00\rangle \tag{3.53}$$

$$U\,|1\rangle \otimes |0\rangle = |1\rangle \otimes |1\rangle = |11\rangle\,. \tag{3.54}$$

Estes representam a porta COPY da computação clássica e pode ser reproduzida através de uma porta CNOT considerando que o bit de controle é o que desejamos copiar e o alvo é a cópia, o que indica que a cópia clássica pode ser realizada dado que a porta CNOT pode ser implementada em um computador quântico, uma vez que esta pode ser representada por um operador unitário. Esse comportamento era esperado pois de certo modo a computação clássica é um limite da computação quântica, mas aqui surge o questionamento, por quê perdemos essa habilidade no contexto quântico? A resposta está na linearidade dos operadores da mecânica quântica e para entendermos como este problema surge devemos considerar a atuação de U sobre o qubit $|\psi\rangle$ de duas formas, a primeira é realizar a conta pensando em termos de produto tensorial, como está feito abaixo

$$\begin{aligned} U\,|\psi\rangle \otimes |0\rangle &= |\psi\rangle \otimes |\psi\rangle = |\psi\psi\rangle \\ &= (\alpha\,|0\rangle + \beta\,|1\rangle) \otimes (\alpha\,|0\rangle + \beta\,|1\rangle) \\ &= \alpha^2\,|00\rangle + \alpha\beta\,|01\rangle + \beta\alpha\,|10\rangle + \beta^2\,|11\rangle\,. \end{aligned} \tag{3.55}$$

Na segunda abordagem, consideramos a linearidade da teoria quântica, esta implica que se o operador U deve atuar na superposição $\alpha\,|0\rangle + \beta\,|1\rangle$ da seguinte maneira

$$U(\alpha\,|0\rangle + \beta\,|1\rangle) \otimes |0\rangle = \alpha\,|00\rangle + \beta\,|11\rangle\,. \tag{3.56}$$

Note que a equação (3.56) tem como resultado um estado emaranhado e não somente isso, exibe também a contradição existente entre os resultados (3.55) e (3.56) de cada abordagem de aplicação do operador. Então se igualarmos os resultados obtidos e analisarmos se existem valores de α e β que façam que as expressões (3.55) e (3.56) serem sempre iguais, veremos que isso não é verdade, assim podemos concluir que existem α e β tal que

$$\alpha^2\,|00\rangle + \alpha\beta\,|01\rangle + \beta\alpha\,|10\rangle + \beta^2\,|11\rangle \neq \alpha\,|00\rangle + \beta\,|11\rangle\,. \tag{3.57}$$

como exemplo a da referência [37], esta usa a evolução unitária para provar o teorema.

Assim, temos que a linearidade dos operadores da mecânica quântica impede a existência de uma copiadora universal, ou seja, existem certos tipos de estados que transformações unitárias não podem copiar, nos limitando a ter cópias apenas de estados ortogonais. Nestes é garantido que a fidelidade seja máxima, este é o caso das cópias de bits clássicos, pois note que para $\alpha = 1$ e $\beta = 0$ (ou $\alpha = 0$ e $\beta = 1$) a expressão (3.57) não é satisfeita e temos o caso (3.53).

3.4 Spin de um Elétron como um Qubit

Para termos uma ideia de como os gates da computação quântica são realizados fisicamente, vamos assumir um modelo bem simplificado de um elétron em um campo magnético. Esse caso não engloba descoerência e outros aspectos de controle do qubit mais avançados, mas serve para fins didáticos. Em modelos mais realísticos, que veremos mais à frente, temos átomos artificiais feitos de junções de Josephson, que são desenhados para terem dois níveis de energia mais distantes dos demais, de forma na aplicação de um pulso de microondas sobre a junção, possamos garantir que não serão acessados níveis de energia mais altos. Essa é uma preocupação importante do ponto de vista da engenharia do computador quântico, pois se ele fosse construído com um sistema de osciladores harmônicos ele teria níveis de energia equidistantes, o que poderia gerar transições indesejadas que poderiam atrapalhar o processo de computação.

De início vamos discutir como seria a implementação da porta NOT através da aplicação de um campo magnético constante na direção x. Neste caso podemos escrever a Hamiltoniana do nosso sistema de interesse como [33]

$$H = -\vec{\mu} \cdot \vec{B}, \tag{3.58}$$

onde $\vec{B}$ representa o campo magnético e $\vec{\mu}$ é o momento magnético do elétron, que é expresso por

$$\vec{\mu} = \gamma \vec{S} \tag{3.59}$$

$$\vec{S} = \frac{\hbar}{2}\vec{\sigma}, \tag{3.60}$$

onde γ representa a razão giromagnética e $\vec{S}$ o vetor do spin. Substituindo o vetor momento magnético, pelo vetor de spin multiplicado por γ, na Hamiltoniana, temos

$$H = -\gamma \vec{B} \cdot \vec{S}. \tag{3.61}$$

Neste caso o sistema pode apresentar dois níveis de energia que podem ser determinados através de

$$E_{\pm} = \pm \frac{\hbar \gamma B_0}{2}, \tag{3.62}$$

onde E_+ representa o elétron cujo spin tem orientação paralela ao campo magnético e E_- para o elétron cujo spin possui orientação antiparalela. Considerando que se aplica um campo magnético constante na direção x e que o spin do elétron tem componentes em todas as direções

$$\vec{B} = B_0\hat{x} \tag{3.63}$$

$$\vec{\sigma} = \sigma_x\hat{x} + \sigma_y\hat{y} + \sigma_z\hat{z}, \tag{3.64}$$

temos que o único termo $\vec{\sigma}$ que irá ter contribuição é σ_x, com isso a Hamiltoniana fica expressa como

$$H = -\frac{\hbar\gamma B_0}{2}\sigma_x. \tag{3.65}$$

O estado do elétron, que estamos usando como qubit, vai evoluir de acordo com o seguinte operador de evolução temporal

$$U(t) = \exp\left(\frac{i\gamma B_0 t}{2}\sigma_x\right), \tag{3.66}$$

e supondo que o elétron inicialmente encontra-se com spin $+\hbar/2$, ou seja, no estado

$$|\psi(0)\rangle = |0\rangle\,, \tag{3.67}$$

podemos ver que a aplicação do operador de evolução temporal ao estado inicial, produz

$$\begin{aligned} |\psi(t)\rangle &= U(t)\,|0\rangle \\ &= \exp\left(\frac{i\gamma B_0 t}{2}\sigma_x\right)|0\rangle \\ &= \cos\left(\frac{\gamma B_0 t}{2}\right)|0\rangle + i\sin\left(\frac{\gamma B_0 t}{2}\right)\sigma_x\,|0\rangle \\ &= \cos\left(\frac{\gamma B_0 t}{2}\right)|0\rangle + i\sin\left(\frac{\gamma B_0 t}{2}\right)|1\rangle\,. \end{aligned} \tag{3.68}$$

Uma vez que nosso objetivo é implementar a porta NOT, desejamos conhecer o instante de tempo t em que o campo magnético deve ser cortardo para que nosso estado assuma a forma

$$|\psi(t^0_{NOT})\rangle = \pm i\,|1\rangle. \tag{3.69}$$

Note que o aparecimento da fase complexa $\pm i$ não afeta o resultado, pois trata-se de uma fase global e portanto não possui efeito físico. O instante de tempo para que obtenhamos o estado desejado deve ser igual as raízes da função cosseno presenta na amplitude de probabilidade, portanto

$$\frac{\gamma B_0 t^0_{NOT}}{2} = \frac{n\pi}{2} \;\Rightarrow\; t^0_{NOT} = \frac{n\pi}{\gamma B_0};\; n \in \mathbb{Z}. \tag{3.70}$$

Com o resultado acima, temos a quantidade de tempo que devemos manter campo magnético ligado para que o estado inicial $|0\rangle$ torne-se $|1\rangle$, dessa forma performando uma porta NOT. Agora vamos fazer o caminho contrário, ou seja, partir de um estado inicial

$$|\phi(0)\rangle = |1\rangle\,, \tag{3.71}$$

deixando esse estado evoluir com o operador de evolução

$$\begin{aligned} |\phi(t)\rangle &= \exp\left(-\frac{i\gamma B_0 t}{2}\sigma_x\right)|1\rangle \\ &= \cos\left(\frac{\gamma B_0 t}{2}\right)|1\rangle - i\sin\left(\frac{\gamma B_0 t}{2}\right)\sigma_x\,|1\rangle \\ &= \cos\left(\frac{\gamma B_0 t}{2}\right)|1\rangle - i\sin\left(\frac{\gamma B_0 t}{2}\right)|0\rangle\,. \end{aligned} \tag{3.72}$$

Note que aqui o sinal da exponencial mudou, isso se deve porque o estado $|1\rangle$ está associado a energia E_-. Nosso objetivo é transformar o estado $|1\rangle$ em $|0\rangle$, então temos que descobrir a quantidade de tempo que precisamos deixar o estado evoluir para que isso ocorra

$$|\phi(t^1_{NOT})\rangle = \pm i\,|0\rangle, \tag{3.73}$$

novamente teremos que buscar o instante de tempo que anula a função cosseno

$$\frac{\gamma B_0 t^1_{NOT}}{2} = \frac{n\pi}{2} \;\Rightarrow\; t^1_{NOT} = \frac{n\pi}{\gamma B_0};\; n \in \mathbb{Z}. \tag{3.74}$$

No caso de campo magnético constante, temos que os tempos para transformar $|0\rangle$ em $|1\rangle$ e vice-versa, são exatamente os mesmos

$$t^0_{NOT} = t^1_{NOT} = t_{NOT}. \tag{3.75}$$

Na Figura 25, podemos ver o comportamento dos estados com o passar do tempo, assim podemos visualizar como a porta NOT é implementada dentro desse contexto de elétron em campo magnético.

A porta NOT também pode ser implementada através de um campo magnético oscilante, sendo assim considere o seguinte campo magnético

$$\vec{B} = B_0\cos(\omega t)\,\hat{x}, \tag{3.76}$$

neste caso a Hamiltoniana passa a apresenta dependência temporal e assume a forma

$$H(t) = -\frac{\hbar\gamma B_0\cos(\omega t)}{2}\sigma_x. \tag{3.77}$$

Com a Hamiltoniana em mãos podemos determinar o operador de evolução temporal, para isso vamos precisar resolver uma integral, neste caso não é complicada, pois trata-se uma

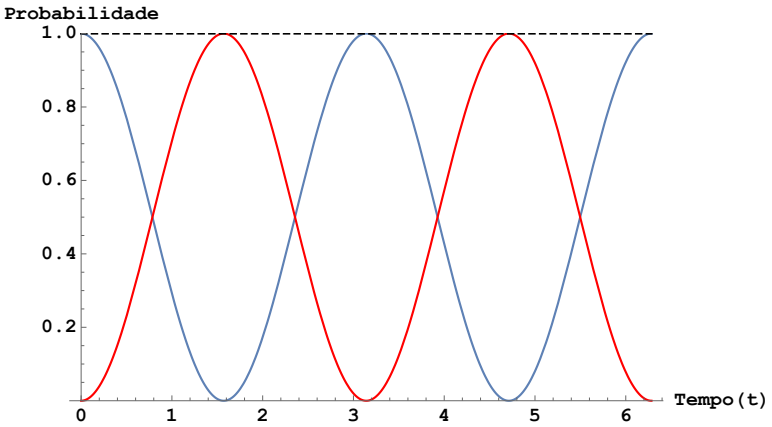


Figura 25 – Probabilidade de se medir os estados $|0\rangle$(curva azul) e $|1\rangle$(curva vermelha) em um instante de tempo t dado os parâmetros $\gamma B_0 = 2$ e $n = 1$.

função cosseno, então fazendo as contas chegaremos ao resultado

$$\begin{aligned} U(t) &= \exp\left(-\frac{i}{\hbar}\int_0^t H(t')dt'\right) \\ &= \exp\left(\frac{i\gamma B_0}{2}\int_0^t \cos(\omega t)\right) \\ &= \exp\left(\frac{i\gamma B_0}{2\omega}\sin(\omega t)\right) \\ &= \cos\left[\frac{\gamma B_0 \sin(\omega t)}{2\omega}\right]\mathbb{I} + i\sin\left[\frac{\gamma B_0 \sin(\omega t)}{2\omega}\right]\sigma_x. \end{aligned} \tag{3.78}$$

Assumindo que o estado inicial seja

$$|\psi'(0)\rangle = |0\rangle\,, \tag{3.79}$$

e que este evolui de acordo com o operador evolução temporal calculado anteriormente, temos que

$$|\psi'(t)\rangle = \cos\left[\frac{\gamma B_0 \sin(\omega t)}{2\omega}\right]|0\rangle + i\sin\left[\frac{\gamma B_0 \sin(\omega t)}{2\omega}\right]|1\rangle\,. \tag{3.80}$$

Uma vez que buscamos implementar a porta NOT, precisamos determinar o instante de tempo que o campo magnético precisa ser cortado e que garante que o estado $|0\rangle$ tornou-se

$$|\psi'(t'_{NOT})\rangle = \pm i\,|1\rangle, \tag{3.81}$$

para fazer isso devemos buscar os instantes de tempo que anulam o cosseno presente na amplitude de probabilidade que acompanha o $|1\rangle$, então fazendo as contas chegamos em

$$\frac{\gamma B_0 \sin(\omega t'_{NOT})}{2\omega} = \frac{n\pi}{2} \;\Rightarrow\; t'_{NOT} = \arcsin\left(\frac{n\pi\omega}{\gamma B_0}\right);\; n \in \mathbb{Z}. \tag{3.82}$$

Agora fazendo o caminho inverso, devemos começar com o seguinte estado

$$|\phi'(0)\rangle = |1\rangle\,, \tag{3.83}$$

novamente teremos que determinar o operador de evolução temporal, mas aqui temos que tomar o cuidado de mudar o sinal, pois estamos trabalhando com um elétron com outra orientação de spin, então levando este fato em conta o operador deve ser igual a

$$\begin{aligned} U(t) &= \exp\left(-\frac{i}{\hbar}\int_0^t H(t')dt'\right) \\ &= \exp\left(\frac{-i\gamma B_0}{2}\int_0^t \cos(\omega t)\right) \\ &= \exp\left(\frac{-i\gamma B_0}{2\omega}\sin(\omega t)\right) \\ &= \cos\left[\frac{\gamma B_0 \sin(\omega t)}{2\omega}\right]\mathbb{I} - i\sin\left[\frac{\gamma B_0 \sin(\omega t)}{2\omega}\right]\sigma_x. \end{aligned} \tag{3.84}$$

A aplicação do operador determinado acima sobre o estado inicial $|\phi'(0)\rangle$ produz o resultado

$$|\phi'(t)\rangle = \cos\left[\frac{\gamma B_0 \sin(\omega t)}{2\omega}\right]|1\rangle - i\sin\left[\frac{\gamma B_0 \sin(\omega t)}{2\omega}\right]|0\rangle. \tag{3.85}$$

Como nosso objetivo é implementar o comportamento da porta NOT, vamos precisar novamente determinar o instante de tempo t que o campo precisa ser cortado para que o comportamento desejado seja produzido, ou seja, o estado

$$|\psi'(t''_{NOT})\rangle = \pm i\,|0\rangle, \tag{3.86}$$

sendo assim devemos buscar os t que anulam o cosseno, dessa forma garantimos que o estado acima seja produzido, então fazendo as contas devemos obter

$$\frac{\gamma B_0 \sin(\omega t''_{NOT})}{2\omega} = \frac{n\pi}{2} \;\Rightarrow\; t''_{NOT} = \arcsin\left(\frac{n\pi\omega}{\gamma B_0}\right);\; n \in \mathbb{Z}. \tag{3.87}$$

Analisando os tempos obtidos para os dois estados iniciais distintos, podemos ver que novamente os tempos são iguais para mudar os estados, já o comportamento dos estados nesse cenário com campo magnético oscilante e dependente do tempo, pode ser visto na Figura 26.

Outro exemplo interessante de ser explorado é a implementação física da porta Hadamard, esta pode ter seu comportamento reproduzido por uma aplicação de um campo magnético constante na direção y sobre um elétron. Para entender como isto é feito devemos considerar a seguinte Hamiltoniana

$$H = -\frac{\hbar\gamma B_0}{2}\sigma_y, \tag{3.88}$$

que produz o operador de evolução temporal

$$U(t) = \exp\left(\frac{i\gamma B_0}{2}\sigma_y t\right). \tag{3.89}$$

Adotando como estado inicial

$$|\psi(0)\rangle = |0\rangle, \tag{3.90}$$

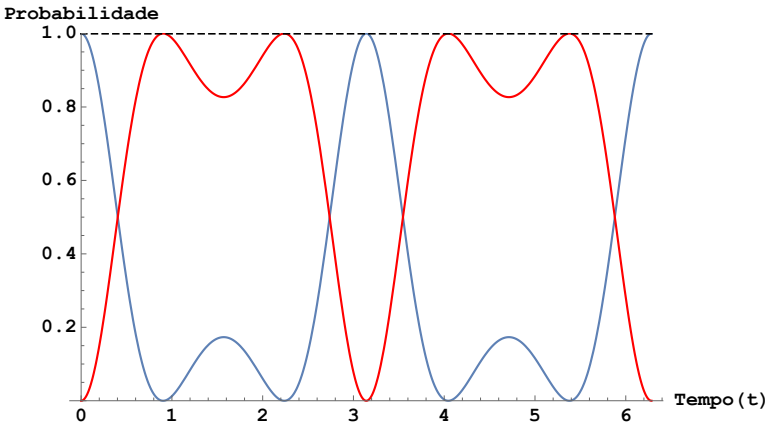


Figura 26 – Probabilidade de se medir os estados $|0\rangle$(curva azul) e $|1\rangle$(curva vermelha) em um instante de tempo t dado os parâmetros $\gamma B_0 = 1$, $\omega = 4$ e $n = 1$.

e deixando que este evolua no tempo de acordo com o operador determinado por (3.88), temos como resultado o seguinte estado $|\psi(t)\rangle$

$$\begin{aligned} |\psi(t)\rangle &= \exp\left(\frac{i\gamma B_0}{2}\sigma_y t\right)|0\rangle \\ &= \left[\cos\left(\frac{\gamma B_0 t}{2}\right)\mathbb{I} + i\sin\left(\frac{\gamma B_0 t}{2}\right)\sigma_y\right]|0\rangle\,;\ \sigma_y|0\rangle = i|1\rangle \\ &= \cos\left(\frac{\gamma B_0 t}{2}\right)|0\rangle - \sin\left(\frac{\gamma B_0 t}{2}\right)|1\rangle\,. \end{aligned} \tag{3.91}$$

Lembrando que a porta Hadamard leva os estados da base de σ_z para a base de σ_x e vice-versa, e que a sua atuação sobre o estado $|0\rangle$ produz

$$|\psi(t_H)\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \tag{3.92}$$

então devemos encontrar o instante de tempo t_H, que faz com que nosso estado inicial torne-se o estado alvo. Para isso, temos que o argumento das funções seno e cosseno presentes nas amplitudes de probabilidade devem ser um múltiplo de $\pi/4$ e que troque o sinal negativo a frente do $|1\rangle$, uma vez que Hadamard aplicava em $|0\rangle$ produz $|+\rangle$ e este não apresenta fase relativa, sendo assim temos que os instantes de tempo que satisfazem as condições necessárias são iguais a

$$\frac{\gamma B_0 t_H}{2} = \frac{315n\pi}{180} \ \Rightarrow\ t_H^0 = \frac{315n\pi}{90\gamma B_0};\ n \in \mathbb{Z}. \tag{3.93}$$

Tomando o caminho inverso, ou seja, vamos transformar um estado da base de σ_x em um estado da base computacional, sendo assim devemos partir de

$$|\phi(0)\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \tag{3.94}$$

e obter o estado $|0\rangle$. Então, aplicando o operador de evolução temporal na superposição em questão, vamos obter

$$\begin{aligned} |\phi(t)\rangle &= \frac{1}{\sqrt{2}}\left[\exp\left(\frac{i\gamma B_0 t}{2}\sigma_y\right)|0\rangle + \exp\left(-\frac{i\gamma B_0 t}{2}\sigma_y\right)|1\rangle\right] \\ &= \frac{1}{\sqrt{2}}\left[\left[\cos\left(\frac{\gamma B_0 t}{2}\right) + \sin\left(\frac{\gamma B_0 t}{2}\right)\right]|0\rangle + \left[\cos\left(\frac{\gamma B_0 t}{2}\right) - \sin\left(\frac{\gamma B_0 t}{2}\right)\right]|1\rangle\right]. \end{aligned} \tag{3.95}$$

Agora nossa tarefa é encontrar um instante de tempo que faça que os valores das funções seno e cosseno serem iguais, para que o termo com o $|1\rangle$ não apareça no estado final como desejado, sendo assim é preciso que o campo magnético seja cortado nos instantes

$$t_H^1 = \frac{n\pi}{2\gamma B_0} \;\Rightarrow\; |\phi(t_H^1)\rangle = |0\rangle\,;\; n \in \mathbb{Z}. \tag{3.96}$$

Assim, fica demonstrado que é possível reproduzir o comportamento esperado de um porta Hadamard através da aplicação de um campo magnético constante na direção y sobre um elétron. Isso mostra que os processos abstratos por trás dos circuitos quânticos apresentam fundamentos físicos para realizar as operações, assim como acontece na computação clássica com circuitos elétricos reproduzindo os resultados da aplicação da álgebra de Boole.

3.5 Transmon Qubit

Os computadores quânticos supercondutores da IBM e do Google utilizam como qubits os transmons, este são da classe de qubits de carga da família dos supercondutores e se utilizam da presença ou ausência de pares de Cooper nas ilhas supercondutoras acopladas por uma junção de Josephson para representar o comportamento de um qubit. O transmon foi proposto nos anos 2000 por Michel Devoret, Steven M. Girvin, Alexandre Blais e outros colaboradores [45] e desde então vem permitindo o avanço da computação quântica, além disso auxiliou ao desenvolvimento de uma área conhecida como *circuit QED*. Visando a introdução dessas ideias, devemos começar pela quantização de um circuito LC clássico, este caso é interessante pois apresenta a energia do sistema oscila entre a energia elétrica no capacitor C e a energia magnética no indutor L, aqui faremos um paralelo entre essas energias com a energia cinética e a energia potencial adotando o padrão da maioria dos artigos na área, ou seja, a convenção de que a energia elétrica do capacitor corresponderia a cinética e a magnética a energia potencial.

A energia instantânea, em cada elemento do circuito, pode ser calculada através da integral

$$E(t) = \int_{-\infty}^{t} V(t')I(t')dt', \tag{3.97}$$

onde $V(t')$ e $I(t')$ representam a voltagem e a corrente do capacitor/indutor. Para construirmos a Hamiltoniana clássica do sistema, vamos seguir pelo caminho padrão da mecânica

clássica, ou seja, vamos obter as expressões para a energia cinética e potencial, para então escrever a Lagrangiana do sistema, para depois, através de um transformação de Legendre, obtermos a Hamiltoniana. Essa abordagem nos faz ter que escolher entre duas opções de variáveis, podemos escrever tudo em termos da carga ou do fluxo de campo magnético, aqui vamos optar pela segunda opção que é definido como a seguinte integral

$$\Phi(t) = \int_{-\infty}^{t} V(t')dt' \;\Rightarrow\; \dot{\Phi} = \frac{d\Phi(t)}{dt} = V(t). \tag{3.98}$$

Lembrando que na teoria de circuitos elétricos temos as seguintes relações envolvendo voltagem, indutância, capacitância e corrente

$$V = L\frac{dI}{dt} \text{ e } I = C\frac{dV}{dt}, \tag{3.99}$$

temos então que a expressão da energia magnética no induto, V_L,é dada pela integral abaixo

$$V_L = \int_{-\infty}^{t} L\frac{dI(t')}{dt'}I(t')dt' = L\frac{I^2(t)}{2} = \frac{\Phi^2(t)}{2L}, \tag{3.100}$$

já a energia elétrica no capacitor, T_C,pode ser calculada através de

$$T_C = \int_{-\infty}^{t} V(t')C\frac{dV(t')}{dt'}dt' = \frac{CV^2(t)}{2} = \frac{C\dot{\Phi}^2}{2}. \tag{3.101}$$

Subtraindo (3.100) de (3.101), temos a definição da Lagrangiana do sistema, que em termos apenas do fluxo de campo magnético e sua primeira derivada toma a forma

$$\begin{aligned} L(\Phi,\dot{\Phi}) &= T_C - V_L \\ &= \frac{C\dot{\Phi}^2}{2} - \frac{\Phi^2(t)}{2L}, \end{aligned} \tag{3.102}$$

substituindo o resultado acima na equação de Euler-Lagrange

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\Phi}}\right) - \frac{\partial L}{\partial \Phi} = 0, \tag{3.103}$$

obtemos a equação de movimento

$$\ddot{\Phi} + \omega^2\Phi = 0;\; \omega = \frac{1}{\sqrt{LC}}, \tag{3.104}$$

esta caracteriza um oscilador harmônico, neste caso com uma frequência dada em termos da capacitância e da indutância. O momento conjugado do sistema em questão é igual a

$$p_\Phi = \frac{\partial L}{\partial \dot{\Phi}} = C\dot{\Phi} = CV = Q, \tag{3.105}$$

ou seja, nessa analogia que adotamos temos que a carga total faz o papel de momento conjugado. Com o momento conjugado p_Φ em mãos, finalmente podemos chegar na

Hamiltoniana do sistema, pois vamos poder descrever o sistema apenas em termos de seu "momento" (carga) e da "posição" (fluxo de campo magnético), sendo assim

$$\begin{aligned} H(Q,\Phi) &= Q\dot{\Phi} - L(\Phi,\dot{\Phi}) \\ &= \frac{Q^2}{2C} + \frac{\Phi^2}{2L} = \frac{CV^2}{2} + \frac{LI^2}{2}. \end{aligned} \tag{3.106}$$

Note que o resultado acima está de acordo com a equação de movimento, pois a estrutura da Hamiltoniana é semelhante a de um oscilador harmônico, o que corrobora para o comportamento oscilatório da energia do sistema entre a energia elétrica e a energia magnética esperado.

A transição de nosso sistema para a mecânica quântica se dá através da quantização de Dirac, ou seja, vamos calcular os parênteses de Poisson e depois usar a relação entre estes e os comutadores [11, 46, 43]. Então calculando os parênteses de Poisson, temos

$$\{\Phi, Q\} = \frac{\delta\Phi}{\delta\Phi}\frac{\delta Q}{\delta Q} - \frac{\delta Q}{\delta\Phi}\frac{\delta\Phi}{\delta Q} = 1. \tag{3.107}$$

Do resultado acima podemos determinar o comutador de Q e Φ, este é igual a

$$[\Phi, Q] = i\hbar. \tag{3.108}$$

A partir desse ponto, a carga e o fluxo de campo magnético passam a ser vistos como operadores. Utilizando as definições de carga reduzida n, fluxo reduzido ϕ e Φ_0 o *superconducting flux magnetic quantum* [46, 43]

$$n = \frac{Q}{2e};\ \phi = 2\pi\frac{\Phi}{\Phi_0} \text{ e } \phi_0 = \frac{h}{2e}, \tag{3.109}$$

o número n corresponde ao excesso de pares de Cooper na ilha supercondutora e o fluxo reduzido é denotado como a fase invariante de calibre. Sendo assim, podemos escrever a hamiltoniana do sistema da seguinte forma

$$H = 4E_C n^2 + \frac{1}{2}E_L\phi^2, \tag{3.110}$$

onde E_C é a *charging energy* requerida para adicionar cada elétron do par de Cooper a ilha supercondutora, já E_L corresponde a energia indutiva, estas são definidas como [46, 43]

$$E_C = \frac{e^2}{2C} \text{ e } E_L = \frac{(\Phi_0/2\pi)^2}{L}. \tag{3.111}$$

Reescrevendo os operadores de número reduzido e de fluxo reduzido, em termos de operadores de criação e aniquilação, temos [43]

$$n = in_{zpf}(a + a^\dagger) \text{ e } \phi = \phi_{zpf}(a - a^\dagger);\ \phi_{zpf} = \left(\frac{2E_C}{E_J}\right)^{\frac{1}{4}};\ n_{zpf}\left(\frac{E_L}{32E_C}\right)^{1/4} \tag{3.112}$$

onde n_{zpf} e ϕ_{zpf} são os valores desses operadores no ponto de flutuações nulas. A partir dessas definições, podemos escrever a Hamiltoniana do sistema em termos dos operadores

de criação e aniquilação, assim fazendo com que H tome a forma exata da Hamiltoniana de um oscilador harmônico quântico

$$H = \hbar\omega\left(a^\dagger a + \frac{1}{2}\right);\ \omega = \frac{\sqrt{8E_L E_C}}{\hbar} = \frac{1}{\sqrt{LC}}. \tag{3.113}$$

Existe um relação muito especial entre a corrente e o fluxo de campo magnético nas junções de Josephson [11, 46, 43], que é dada por

$$I = I_0 \sin\left(\frac{2\pi\Phi}{\Phi_0}\right), \tag{3.114}$$

onde I_0 é a corrente máxima que pode fluir através da junção, enquanto Φ_0 representa o *flux quantum*. Retornando as ideias apresentadas anteriormente sobre quantização do circuito LC e adotando a corrente acima como o termo de energia potencial, temos a seguinte Lagrangiana

$$L(\Phi, \dot{\Phi}) = \frac{C\dot{\Phi}^2}{2} - I_0 \sin\left(\frac{2\pi\Phi}{\Phi_0}\right), \tag{3.115}$$

que se substituída na equação de Euler-Lagrange nos leva a seguinte equação de movimento

$$C\ddot{\Phi} + \frac{I_0\Phi_0}{2\pi}\cos\left(\frac{2\pi\Phi}{\Phi_0}\right) = 0. \tag{3.116}$$

Novamente, usando uma transformação de Legendre e a Lagrangiana do sistema, obtemos a seguinte Hamiltoniana

$$\begin{aligned} H &= \frac{Q^2}{2C} - \frac{I_0\Phi_0}{2\pi}\cos\left(\frac{2\pi\Phi}{\Phi_0}\right) \\ &= 4E_C n^2 - E_J\cos\phi;\ E_J = \frac{I_0\Phi_0}{2\pi}, \end{aligned} \tag{3.117}$$

onde E_J representa a energia de Josephson, que nesse contexto substitui a energia no indutor do circuito LC. Esta Hamiltoniana, define uma espécie de átomo artificial onde podemos isolar e controlar os dois primeiros níveis de energia que serão usados para representar o qubit. Diferente do oscilador harmônico, este átomo artificial não apresenta a mesma diferença de energia entre os níveis de energia, isto facilita esse controle que buscamos para a realização física dos qubits. A anarmonicidade presente nos níveis de energia deste sistema é dada por [43]

$$\omega_j = \left(\omega - \frac{\delta}{2}\right)j + \frac{\delta}{2}j^2, \tag{3.118}$$

os efeitos introduzidos pela anamorcidade no espectro de energia do Transmon podem ser visualizados na Figura 27.

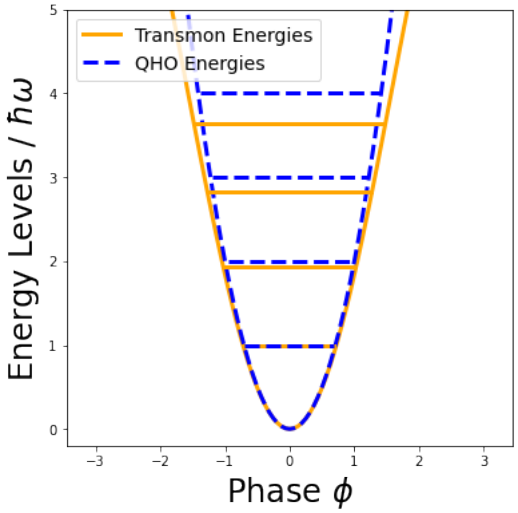


Figura 27 – Comparação entre os níveis de energia de um oscilador harmônico quântico e de um Transmon [43].

3.5.1 Controle do Qubit Através do Uso de Campo Elétrico

Para executarmos os algoritmos nos computadores quânticos, precisamos de uma forma de controlar os qubits, no caso do Transmon são usados pulsos de microondas para reproduzir o comportamento das portas. Para entendermos como isso ocorre devemos começar considerando um campo elétrico com a seguinte forma

$$\vec{E}(t) = \vec{E}_0 e^{-i\omega_d t} + \vec{E}_0^* e^{i\omega_d t}, \tag{3.119}$$

esse campo introduz uma interação de dipolo entre o Transmon e o *microwave field*. Por simplicidade, vamos tratar o transmon como um sistema de dois níveis, vale ressaltar que o Transmon apresenta mais níveis, mas que ajustamos os parâmetros para que ele seja um sistema de dois níveis artificial, dado que existe uma diferença de energia considerável entre os dois primeiros níveis de energia e os demais. A Hamiltoniana desse sistema possui dois termos, um referente ao qubit e o outro um termo de interação fazendo o papel de controle [43]

$$H = H_0 + H_d;\ H_0 = -\frac{1}{2}\hbar\omega_q\sigma_z. \tag{3.120}$$

O vetor do dipolo elétrico produzido pelo campo elétrico aplicado é escrito como

$$\vec{d} = \vec{d}_0\sigma_+ + \vec{d}_0^{\,*}\sigma_-;\ \sigma_\pm = \frac{1}{2}\left(\sigma_x \pm i\sigma_y\right), \tag{3.121}$$

sendo assim, o termo de interação da Hamiltoniana pode ser reescrito como

$$\begin{aligned} H_d &= -\hbar\vec{d}\cdot\vec{E}(t) \\ &= -\hbar\left(\Omega e^{-i\omega_d t} + \tilde{\Omega}e^{i\omega_d t}\right)\sigma_+ - \hbar\left(\tilde{\Omega}^* e^{i\omega_d t} + \Omega^* e^{i\omega_d t}\right)\sigma_-;\ \Omega = \vec{d}_0\cdot\vec{E}_0;\ \tilde{\Omega} = \vec{d}_0\cdot\vec{E}_0^{\,*}, \end{aligned} \tag{3.122}$$

uma vez que temos o termo de interação, precisamos determinar sua forma no quadro de interação, ou seja,

$$H_{d,I} = U(t)H_dU^\dagger(t), \tag{3.123}$$

no caso em questão, o operador que realiza a evolução do operador no quadro de interação, é dado por

$$U(t) = \exp\left(i\frac{\omega_q}{2}\sigma_z t\right). \tag{3.124}$$

Expandindo a exponencial em série de Taylor, e separando em duas séries, uma com termos pares e outra com termos ímpares, temos

$$\exp\left(i\frac{\omega_q}{2}\sigma_z t\right) = \cos\left(\frac{\omega_q t}{2}\right)\mathbb{I} + i\sin\left(\frac{\omega_q t}{2}\right)\sigma_z. \tag{3.125}$$

Dessa maneira, a parte de controle da Hamiltoniana pode ser escrita da seguinte forma no quadro de interação

$$H_{d,I} = -\hbar\left(\Omega e^{-i\omega_d t} + \tilde{\Omega}e^{i\omega_d t}\right)U(t)\sigma_+U^\dagger(t) - \hbar\left(\tilde{\Omega}^* e^{i\omega_d t} + \Omega^* e^{i\omega_d t}\right)U(t)\sigma_-U^\dagger(t), \tag{3.126}$$

substituindo $U(t)$ e $U^\dagger(t)$ pelas expansões em Taylor das exponenciais, aliadas a relação de comutação entre σ_z e σ_+ e as relações trigonométricas envolvendo arco duplo, é possível mostrar que

$$\begin{aligned}
U(t)\sigma_+U^\dagger(t) &= \left[\cos\left(\frac{\omega_q t}{2}\right)\mathbb{I} + i\sin\left(\frac{\omega_q t}{2}\right)\sigma_z\right]\sigma_+\left[\cos\left(\frac{\omega_q t}{2}\right)\mathbb{I} - i\sin\left(\frac{\omega_q t}{2}\right)\sigma_z\right] \\
&= \left[\cos\left(\frac{\omega_q t}{2}\right)\mathbb{I} + i\sin\left(\frac{\omega_q t}{2}\right)\sigma_z\right]\left[\cos\left(\frac{\omega_q t}{2}\right)\sigma_+ - i\sin\left(\frac{\omega_q t}{2}\right)\sigma_+\sigma_z\right] \\
&= \cos^2\left(\frac{\omega_q t}{2}\right)\sigma_+ + \frac{i}{2}\sin(\omega_q t)\,\sigma_z\sigma_+ - \frac{i}{2}\sin(\omega_q t)\,\sigma_+\sigma_z + \sin^2\left(\frac{\omega_q t}{2}\right)\sigma_z\sigma_+\sigma_z \\
&= \cos^2\left(\frac{\omega_q t}{2}\right)\sigma_+ - \sin^2\left(\frac{\omega_q t}{2}\right)\sigma_+ + i\sin(\omega_q t)\,\sigma_+;\ \ \sigma_z\sigma_+\sigma_z = -\sigma_+ \\
&= \cos(\omega_q t)\,\sigma_+ + i\sin(\omega_q t)\,\sigma_+ \\
&= e^{i\omega_q t}\sigma_+,
\end{aligned} \tag{3.127}$$

e de maneira análoga podemos provar que

$$U(t)\sigma_-U^\dagger(t) = e^{-i\omega_q t}\sigma_-. \tag{3.128}$$

Substituindo os resultados obtidos em (3.126) obtém-se

$$\begin{aligned}
H_{d,I} &= U(t)H_dU^\dagger(t) \qquad (3.129)\\
&= -\hbar\left(\Omega e^{-i\omega_d t} + \tilde{\Omega}e^{i\omega_d t}\right)e^{i\omega_q t}\sigma_+ - \hbar(\tilde{\Omega}^* e^{-i\omega_d t} + \Omega^* e^{i\omega_d t})e^{-i\omega_d t}\sigma_- \\
&= -\hbar\left(\Omega e^{-i\Delta_q t} + \tilde{\Omega}e^{i(\omega_d+\omega_q)t}\right)\sigma_+ - \hbar(\tilde{\Omega}^* e^{-i(\omega_d+\omega_q)t} + \Omega^* e^{i\Delta_q t})\sigma_-;\ \ \Delta_q = \omega_q - \omega_d,
\end{aligned}$$

e uma vez que pode-se considerar a condição abaixo

$$\omega_q + \omega_d \gg \Delta_q, \tag{3.130}$$

podemos fazer uso da aproximação RWA (*Rotating Wave Approximation*) [47] como argumento para desprezar os termos que levam $\omega_q + \omega_d$ no expoente. Então, a Hamiltoniana com a aproximação RWA toma a forma

$$H_{d,I}^{(RWA)} = -\hbar\Omega e^{-i\Delta_q t}\sigma_+ - \hbar\Omega^* e^{i\Delta_q t}\sigma_-. \tag{3.131}$$

Retornando ao quadro de Schrödinger, o termo de interação fica expresso por

$$\begin{aligned} H_d^{(RWA)} &= U^\dagger(t) H_{d,I}^{(RWA)} U(t) \\ &= -\hbar\Omega e^{-i\omega_d t}\sigma_+ - \hbar\Omega^* e^{i\omega_d t}\sigma_-, \end{aligned} \tag{3.132}$$

e portanto a Hamiltoniana total (RWA), no quadro de Schrödinger é igual a

$$H^{(RWA)} = -\frac{\hbar\omega_q}{2}\sigma_z - \hbar\Omega e^{-i\omega_d t}\sigma_+ - \hbar\Omega^* e^{i\omega_d t}\sigma_-. \tag{3.133}$$

Utilizando a Hamiltoniana acima na equação de Schrödinger para estudar a evolução do estado do Transmon, temos que

$$i\hbar\frac{\partial\left|\psi(t)\right\rangle}{\partial t} = H^{(RWA)}\left|\psi(t)\right\rangle, \tag{3.134}$$

então visando resolver a equação acima, vamos precisar fazer a seguinte troca

$$\left|\psi(t)\right\rangle = U_d(t)\left|\phi(t)\right\rangle, \tag{3.135}$$

sendo assim se substituirmos e derivarmos parcialmente, vamos produzir a expressão

$$i\hbar\left(\frac{\partial U_d(t)}{\partial t}\left|\phi(t)\right\rangle + U_d(t)\frac{\partial\left|\phi(t)\right\rangle}{\partial t}\right) = H^{(RWA)}U_d(t)\left|\phi(t)\right\rangle. \tag{3.136}$$

Isolando os termos com a derivada parcial de $\left|\phi(t)\right\rangle$ no lado esquerdo da equação e utilizando a propriedade do Hermitiano conjugado do produto no termo com derivada de $U(t)$, temos

$$i\hbar\frac{\partial\left|\phi(t)\right\rangle}{\partial t} = U_d^\dagger(t)H^{(RWA)}U_d(t) - i\hbar\frac{\partial U^\dagger(t)}{\partial t}U(t), \tag{3.137}$$

e com isso chegamos na Hamiltoniana efetiva do sistema em questão. Realizando substituições no resultado acima, manipulando termos e considerando que Ω seja uma constante real, é possível simplificar nossa Hamiltoniana efetiva, então após a realização das etapas descritas é possível deixá-la na forma

$$\begin{aligned} H_{eff} &= -\frac{\hbar\omega_q}{2}\sigma_z - \hbar\Omega\sigma_+ - \hbar\Omega^*\sigma_- + \frac{\hbar\omega_d}{2}\sigma_z;\ \Omega = \Omega^* \\ &= -\frac{\hbar\Delta_q}{2}\sigma_z - \hbar\Omega\sigma_x. \end{aligned} \tag{3.138}$$

O resultado acima é bem interessante, pois quando a frequência do campo de controle entra em ressonância com a frequência do qubit, ou seja, $\Delta_q = 0$, a Hamiltoniana efetiva se reduz a

$$H_{eff} = -\hbar\Omega\sigma_x, \tag{3.139}$$

o que configura a aplicação de uma porta NOT no qubit considerado, esse tipo de operação pode ser realizada remotamente através da plataforma IBM Quantum Experience e o Qiskit Pulse, na referência [43] encontram-se instruções para a realização desse procedimento, que basicamente consistem em duas etapas, a primeira consiste em encontrar a frequência do qubit e depois calibrar e aplicar um pulso π.

3.6 Emaranhamento Como Recurso na Informação Quântica

No capítulo anterior, vimos como identificar um estado emaranhado e como quantificar as correlações quânticas presentes no sistema. A proposta desta seção é apresentar ao leitor que além de ser uma profunda questão dentro da Física, o emaranhamento pode ser usado como recurso na informação quântica, aqui vamos explorar a codificação superdensa e o teletransporte quântico. Estes dois protocolos são usados na tarefa de comunicação, o primeiro consiste na manipulação de n qubits em um estado emaranhado, através de portas mapeadas em um código previamente estabelecido, para transmitir 2^n bits clássicos, enquanto o segundo faz uso de dois bits clássicos para transmitir um estado quântico arbitrário através de um par de qubits emaranhados. Além dos protocolos mencionados, também discutiremos como usar a produção de estados emaranhados em computadores quânticos para mostrar como calcular a energia da estrutura hiperfina do átomo de hidrogênio e como usá-los para realizar uma verificação empírica da desigualdade de Bell.

Antes de discutir ambos os protocolos em mais detalhes, vamos mostrar como criar alguns tipos de estados emaranhados usando circuitos quânticos. Aqui vale ressaltar que vamos nos preocupar apenas com a estrutura do circuito e não vamos entrar nos detalhes físicos por trás das portas como explorado nas seções 3.3 e 3.4, mas vale o lembrete de que por trás do que será apresentado aqui existe toda uma cadeia de processos físicos por trás.

Considere que inicialmente temos o seguinte estado separável de dois qubits

$$|\psi_{AB}\rangle = |0_A 0_B\rangle, \tag{3.140}$$

e que desejamos criar emaranhamento entre os qubits presentes no estado. Para isso, devemos começar aplicando o operador $H_A \otimes \mathbb{I}_B$, dessa forma vamos produzir o estado

$$|\psi'_{AB}\rangle = (H_A \otimes \mathbb{I}_B)|\psi_{AB}\rangle = \frac{1}{\sqrt{2}}(|0_A\rangle + |1_A\rangle) \otimes |0_B\rangle, \tag{3.141}$$

contudo perceba que o qubit pertencente ao espaço $\mathcal{H}_A$ está em uma superposição devido a atuação da porta Hadamard, apesar disso o estado resultante ainda é separável, pois ele nada mais é do que $|\psi'_{AB}\rangle = |+_A\rangle \otimes |0_B\rangle$. Então para que os qubits criem emaranhamento entre eles, é preciso uma interação local entre eles, ou seja, vamos precisar de uma porta que atue sobre dois qubits e que faça o papel de interação local, neste caso devemos usar

a porta CNOT para realizar tal tarefa[3], com isso teremos como resultado final

$$|\psi_+\rangle = U_{CNOT}|\psi'_{AB}\rangle = \frac{1}{\sqrt{2}}(|0_A0_B\rangle + |1_A1_B\rangle). \tag{3.142}$$

Os passos descritos acima para a criação do estado emaranhado $|\psi_+\rangle$ podem ser resumidos pelo circuito mostrado na Figura 28 (a). Uma característica interessante sobre o processo apresentado é que ele possui uma estrutura replicável para diferentes estados de entrada do circuito, ou seja, independente do estado inicial sempre vamos manter a estrutura de primeiro aplicar $H_A \otimes \mathbb{I}_B$ e depois CNOT. Dessa forma, podemos resumir a produção dos estados (2.180), (2.181), (2.182) e (2.183), como a sequência a aplicação em sequência dos operadores $H_A \otimes \mathbb{I}_B$ e CNOT sobre o conjunto de estados $\{|0_A0_B\rangle, |0_A1_B\rangle, |1_A0_B\rangle, |1_A1_B\rangle\}$. A Figura 28 retrata o resumo em questão.

A estrutura apresentada até aqui produz apenas estados de máximo emaranhamento, ou seja, cuja entropia de von Neumann das matrizes de densidade reduzidas são máximas, contudo sabemos que um estado emaranhado não necessariamente precisa ter máxima entropia. Sendo assim, para poder expandir nossa capacidade de produção de estados emaranhados, devemos substituir o operador $H_A \otimes \mathbb{I}_B$ por $R_Y^A(\theta) \otimes \mathbb{I}_B$, neste caso o nível de emaranhamento no sistema dependerá diretamente do ângulo θ. Então se considerarmos como estado inicial $|\psi_{AB}\rangle = |0_A0_B\rangle$ e usarmos os procedimentos conhecidos com o novo operador, teremos como resultado

$$|\psi''_{AB}\rangle = \cos\left(\frac{\theta}{2}\right)|0_A0_B\rangle + \sin\left(\frac{\theta}{2}\right)|1_A1_B\rangle. \tag{3.143}$$

O resultado acima nos diz que nessa nova abordagem, só produziremos um estado com máximo emaranhamento quando θ for igual a $\pi/2$, neste caso a rotação $R_Y(\theta)$ coincide com o operador Hadamard. Essa diferenciação entre níveis de emaranhamento é importante, pois nos permite identificar se existem "vazamentos" de correlações quânticas para outros sistemas. Sendo assim, o estado com máximo emaranhamento é extremamente importante, pois ele garante que as correlações só serão compartilhadas pelas partes que compõe o estado, essa característica é conhecida na literatura como monogamia [48]. A partir dela definimos os conceitos de estado monogâmico e não monogâmico, o primeiro consiste em estado que não apresenta vazamento de correlação e o segundo consiste em estado que compartilha correlação com outros sistemas. Os estados monogâmicos são muito bem vindos na teoria quântica de correção de erros, pois como foi visto no capítulo 1, desejamos criar códigos que maximizem a entropia das variáveis aleatórias associadas a entrada e a saída do canal de comunicação para poder restringir o quanto o ruído destruirá do mapeamento um a um entre as palavras-código e as mensagens recebidas, sendo assim no contexto quântico também vamos querer trabalhar com máxima entropia para evitar

[3] Aqui estamos considerando que o qubit pertencente a $\mathcal{H}_A$ é o qubit de controle e o pertencente a $\mathcal{H}_B$ o alvo.

alta degenerescência nas mensagens recebidas, o que facilita o processo de identificação da palavra-código original na decodificação.

Listing 3.1 – Código em Qiskit que produz o circuito (a) da Figura 28.

```
from qiskit.circuit import QuantumRegister, QuantumCircuit

qubits=QuantumRegister(2, name='qubits')
qc=QuantumCircuit(qubits)

qc.h(qubits[0])
qc.cx(qubits[0],qubits[1])
```

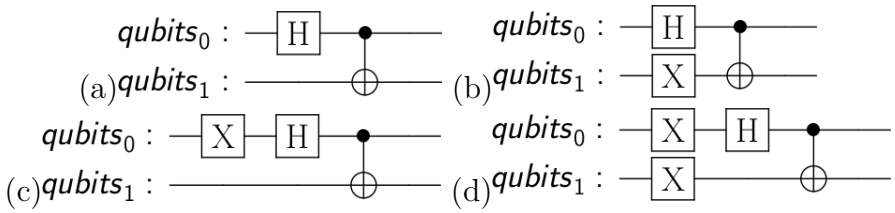


Figura 28 – Circuitos quânticos para criar os estados emaranhados: (a) $|\psi_+\rangle$ (2.180); (b) $|\phi_+\rangle$ (2.181); (c) $|\psi_-\rangle$ (2.182) e (d) $|\phi_-\rangle$ (2.183).

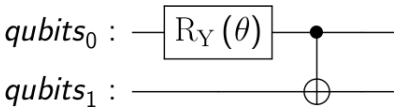


Figura 29 – Circuito parametrizado que permite a criação de estados com vários níveis de emaranhamento a depender do parâmetro θ.

3.6.1 Codificação Superdensa

Em 1992 [1, 49], Charles Bennett e Stephen Wiesner explicaram como poderíamos transmitir dois bits clássicos de informação através da transmissão de um qubit em um canal quântico. Este resultado ficou conhecido como codificação superdensa, pois utiliza da relação N qubits mapeiam 2^N bits, neste caso em específico temos a relação 1 para 2, mas é possível generalizar este protocolo para um número natural N de qubits.

A fim de compreendermos a codificação superdensa, vamos considerar o seguinte cenário: considere dois personagens que desejam se comunicar, Alice e Bob, e que ambos podem assumir os papéis de remetente e destinatário. Os recursos que eles dispõem para realizar o protocolo são um estado emaranhado

$$|\psi_+\rangle = \frac{1}{\sqrt{2}}(|0_A 0_B\rangle + |1_A 1_B\rangle),$$

e um conjunto de portas quânticas que definem o seguinte código de comunicação

$$\mathbb{I} \rightarrow |0_A 0_B\rangle \tag{3.144}$$
$$X \rightarrow |0_A 1_B\rangle \tag{3.145}$$
$$Y \rightarrow |1_A 1_B\rangle \tag{3.146}$$
$$Z \rightarrow |1_A 0_B\rangle. \tag{3.147}$$

Sendo assim, a ideia central é que quando uma das partes envolvidas na comunicação manipula seu qubit através de uma operação local, temos uma alteração do estado emaranhado inicial e que após a realização de uma medida de Bell, obtém-se um estado que representa a mensagem correspondente ao operador aplicado, assim configurando a transmissão da mensagem desejada. Então, podemos concluir que a codificação superdensa no cenário descrito consiste no envio de dois bits clássicos através da manipulação local de um qubit que está emaranhado com outro qubit envolvido no processo.

Agora que sabemos os conceitos por trás da codificação superdensa, vamos construir o circuito quântico que a executa e fazer a ligação com as ideias apresentadas. O primeiro passo é a criação de um estado emaranhado, no exemplo em questão estamos usando (2.180), sendo assim o primeiro passo é realizar os procedimentos descritos no início da seção 3.6, após isso temos a etapa de "escrita" da mensagem através da aplicação de um dos operadores do código por uma das partes em seu respectivo qubit, por exemplo: considere que Alice quer enviar a Bob a mensagem 01, então ela consulta o código e realiza a operação correspondente, dessa maneira ela produzirá o estado

$$|\psi'_+\rangle = X_A \otimes \mathbb{I}_B |\psi_+\rangle = \frac{1}{\sqrt{2}}(|1_A 0_B\rangle + |0_A 1_B\rangle). \tag{3.148}$$

Uma vez que a mensagem chega ao destinatário, este precisará decodificá-la e neste caso o processo corresponde a realizar uma medida de Bell, que nada mais é que a aplicação consecutiva de uma porta CNOT e uma porta Hadamard, então realizando o primeiro passo no estado acima teremos como resultado

$$|\psi''_+\rangle = U_{CNOT} |\psi'_+\rangle = \frac{1}{\sqrt{2}}(|1_A 1_B\rangle + |0_A 1_B\rangle), \tag{3.149}$$

e dando continuidade ao processo temos a etapa de aplicação da porta Hadamard no estado resultado da etapa anterior, com isso obtemos

$$|\psi'''_+\rangle = (H_A \otimes \mathbb{I}_B) |\psi''_+\rangle = |0_A 1_B\rangle, \tag{3.150}$$

que é o estado que carrega a mensagem 01 enviada por Alice. Um aspecto importante a ser ressaltado sobre a comunicação através da codificação superdensa, é que ela é completamente simétrica, ou seja, Alice e Bob podem ter seus papéis de remetente e destinatário alternados sem precisar recorrer a um novo código, por exemplo: considere

que Bob deseja enviar uma resposta a Alice com o mesmo conteúdo da mensagem que recebeu para informá-la que tudo deu certo, sendo assim ele precisará realizar a operação apresentada abaixo

$$|\Psi'_+\rangle = \mathbb{I}_A \otimes X_B |\psi_+\rangle = \frac{1}{\sqrt{2}}(|0_A 1_B\rangle + |1_A 0_B\rangle). \tag{3.151}$$

Note que o estado resultante é o mesmo de quando Alice enviou a mensagem, então podemos esperar que a medida de Bell fará com que o resultado final seja o mesmo, e de fato é isto que ocorre quando realizamos as contas abaixo que representam o procedimento de decodificação

$$|\Psi''_+\rangle = U_{CNOT}|\Psi'_+\rangle = \frac{1}{\sqrt{2}}(|0_A 1_B\rangle + |1_A 1_B\rangle) \tag{3.152}$$

$$|\Psi'''_+\rangle = (H_A \otimes \mathbb{I}_B)|\Psi''_+\rangle = |0_A 1_B\rangle. \tag{3.153}$$

Na Figura 30 encontram-se todos os circuitos relacionados ao código apresentado aqui, todos eles descrevem as contas mostradas de forma gráfica, já na Figura 31 temos os histogramas com os resultados da execução do circuito (d) da Figura 30 em um simulador de um computador quântico e no computador quântico de 5 qubits Santiago da IBM. Para realizar a execução do circuito no computador Santiago foram usados os qubits de número 3 e 4.

Listing 3.2 – Código em Qiskit que produz o circuito (d) da Figura 30.

```
from qiskit.circuit import ClassicalRegister, QuantumRegister, QuantumCircuit

qubits=QuantumRegister(2,name='qubits')
bits=ClassicalRegister(2,name='bits')
qc=QuantumCircuit(qubits)

qc.h(qubits[0])
qc.cx(qubits[0],qubits[1])
qc.barrier()
qc.z(qubits[0])
qc.barrier()
qc.cx(qubits[0],qubits[1])
qc.h(qubits[0])
qc.barrier()
qc.measure(qubits,bits)
```

3.6.2 Teletransporte Quântico

Em 1993 [50], Charles Bennet, Gilles Bassard, Claude Crépeau, Richard Josza, Asher Peres e William K. Wootters propuseram uma técnica que leva um estado quântico arbitrário de um lado para o outro, mesmo na ausência de um canal de comunicação

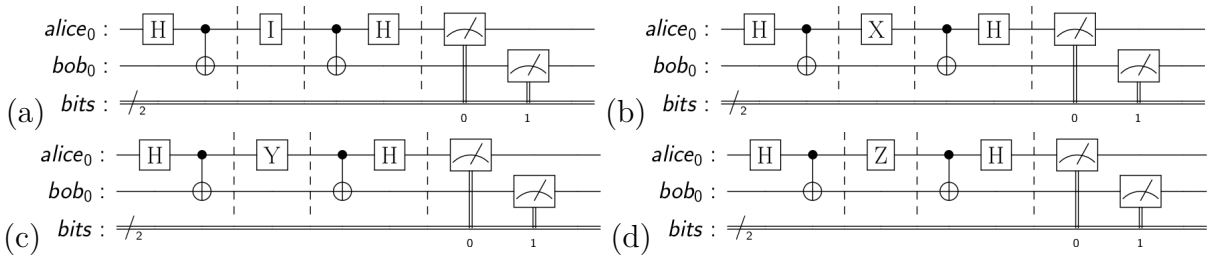


Figura 30 – Circuitos quânticos de codificação superdensa para a transmissão de dois bits clássicos.

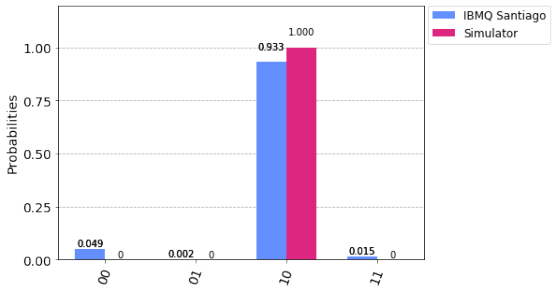


Figura 31 – Resultado de 8192 execuções do circuito quântico que transmite a mensagem 10 no simulador e na QPU (*Quantum Processor Unit*) IBMQ Santiago 5 qubits.

quântico, o único requisito é que sejam transmitidos dois bits clássicos, resultantes de uma medição, que irão nos informar quais operações precisarão ser realizadas no destino. Nos últimos anos, tivemos alguns experimentos importantes buscando implementar o teletransporte quântico no setor de telecomunicações, por exemplo em 2017 os chineses usaram um satélite em órbita para realizar o teletransporte de um estado quântico entre dois laboratórios em território da China [51], já em 2020 foi testada nos EUA, uma tecnologia que utiliza o teletransporte quântico em um protótipo do que vem sendo chamado de internet quântica [52]. O teletransporte quântico também vem sendo usado em códigos de correção de erros [53] e como recurso em projetos de processadores quânticos [54]. Por fim, também existem estudos que combinam a codificação superdensa com o teletransporte quântico para simular uma comunicação segura através de *wormholes* [55].

Dado que desejamos entender como funciona o protocolo de teletransporte quântico, vamos considerar o seguinte cenário: Alice e Bob estão separados por uma distância maior que 1 ano-luz e compartilham um estado emaranhado que produziram através de interações locais, quando estavam juntos em um laboratório em algum momento no passado, em algum momento do presente Alice possui um estado quântico desconhecido que deseja enviá-lo a Bob. Contudo, ela se encontra em uma situação complicada, uma vez que não existe canal de comunicação quântico que faça a ligação entre ela e Bob, e mesmo que existisse tal canal ela não poderia copiar o estado para enviar para Bob, pois o teorema da não clonagem a impede de clonar um estado quântico arbitrário, além disso ela também

não pode realizar medidas no estado para conhecer algum tipo de propriedade interessante que ela gostaria de contar a Bob, pois isto provocaria um colapso na função de onda desse sistema de interesse, o que causaria uma perda da informação quântica codifica neste estado. Diante dessa situação, Alice resolveu se debruçar sobre livros de computação quântica buscando uma solução, até se deparar com o teletransporte quântico, onde ela viu que poderia realizar manipulações nos qubits que estão com ela e realizar uma medição que daria a ela os dois bits clássicos que ela precisa enviar, através de um canal de comunicação clássico, a Bob que uma vez em posse dessa informação pode realizar procedimentos para finalizar o teletransporte do estado quântico desconhecido.

Uma vez conhecido o procedimento, podemos ir para a matemática por trás do fenômeno, sendo assim vamos considerar que o estado que Alice deseja enviar a Bob seja o apresentado abaixo

$$|\psi_{A1}\rangle = \alpha|0_{A1}\rangle + \beta|1_{A1}\rangle, \tag{3.154}$$

então realizando o produto tensorial deste com o estado emaranhado (2.180) que Alice e Bob compartilham, temos que o estado quântico do sistema completo é representado por

$$\begin{aligned} |\Psi\rangle &= |\psi_{A1}\rangle \otimes \frac{1}{\sqrt{2}}(|0_{A2}0_B\rangle + |1_{A2}1_B\rangle) \\ &= (\alpha|0_{A1}\rangle + \beta|1_{A1}\rangle) \otimes \frac{1}{\sqrt{2}}(|0_{A2}0_B\rangle + |1_{A2}1_B\rangle). \end{aligned} \tag{3.155}$$

Alice precisa realizar uma medida de Bell nos qubits que estão com ela, para isso ela primeiro deve realizar a aplicação de uma porta CNOT com $A1$ sendo o qubit de controle e $A2$ o alvo, levando isso em conta a aplicação deste operador sobre o estado $|\Psi\rangle$ produz

$$\begin{aligned} |\Psi'\rangle &= U_{CNOT}|\Psi\rangle \\ &= \frac{1}{\sqrt{2}}[\alpha|0_{A1}\rangle \otimes (|0_{A2}0_B\rangle + |1_{A2}1_B\rangle) + \beta|1_{A1}\rangle \otimes (|1_{A2}0_B\rangle + |0_{A2}1_B\rangle)]. \end{aligned} \tag{3.156}$$

O próximo passo que Alice deve tomar é a aplicação de uma porta Hadamard sobre o estado acima, isto produzirá um estado que nos permitirá agrupar os qubits que pertencem a Alice da seguinte maneira

$$\begin{aligned} |\Psi''\rangle &= H_{A1} \otimes \mathbb{I}_{A2} \otimes \mathbb{I}_B|\Psi'\rangle \\ &= \frac{1}{2}[\alpha(|0_{A1}\rangle + |1_{A1}\rangle) \otimes (|0_{A2}0_B\rangle + |1_{A2}1_B\rangle) + \beta(|0_{A1}\rangle - |1_{A1}\rangle) \otimes (|1_{A2}0_B\rangle + |0_{A2}1_B\rangle)] \\ &= \frac{1}{2}[|0_{A1}0_{A2}\rangle \otimes (\alpha|0_B\rangle + \beta|1_B\rangle) + |0_{A1}1_{A2}\rangle \otimes (\alpha|1_B\rangle + \beta|0_B\rangle) \\ &+ |1_{A1}0_{A2}\rangle \otimes (\alpha|0_B\rangle - \beta|1_B\rangle) + |1_{A1}1_{A2}\rangle \otimes (\alpha|1_B\rangle - \beta|0_B\rangle)], \end{aligned} \tag{3.157}$$

note que o agrupamento feito acima nos remete a quatro opções de medida do estado de Alice, que por sua vez estão ligados a diferentes superposições do qubit de Bob, isto nos

revela o seguinte mapeamento

$$|0_{A1}0_{A2}\rangle \rightarrow |\psi_B\rangle = \alpha|0_B\rangle + \beta|1_B\rangle \tag{3.158}$$
$$|0_{A1}1_{A2}\rangle \rightarrow |\psi_B\rangle = \alpha|1_B\rangle + \beta|0_B\rangle \tag{3.159}$$
$$|1_{A1}0_{A2}\rangle \rightarrow |\psi_B\rangle = \alpha|0_B\rangle - \beta|1_B\rangle \tag{3.160}$$
$$|1_{A1}1_{A2}\rangle \rightarrow |\psi_B\rangle = \alpha|1_B\rangle - \beta|0_B\rangle. \tag{3.161}$$

Comparando os estados de Bob no mapeamento acima, com o estado original (3.154), podemos identificar as operações que ele deverá realizar em seu qubit para recuperar (3.154) de acordo com a medida que Alice obter, para o caso de medida 00 Bob deve aplicar uma porta identidade ou simplesmente não fazer nada, pois seu qubit já está no estado correto, já para 01 ele deve aplicar uma porta X, para 10 deve aplicar Z e quando ele receber 11 deve aplicar uma X seguida de uma Z, com isso garantimos que em todos os quatro casos que o qubit de Bob está no estado desconhecido que Alice desejava teletransportar. Na Figura 32 temos um exemplo de circuito que realiza o protocolo de teletransporte quântico, já na Figura 33 podemos encontrar o resultado da execução do circuito exibido na Figura 32 em um simulador e no computador quântico Santiago da IBM.

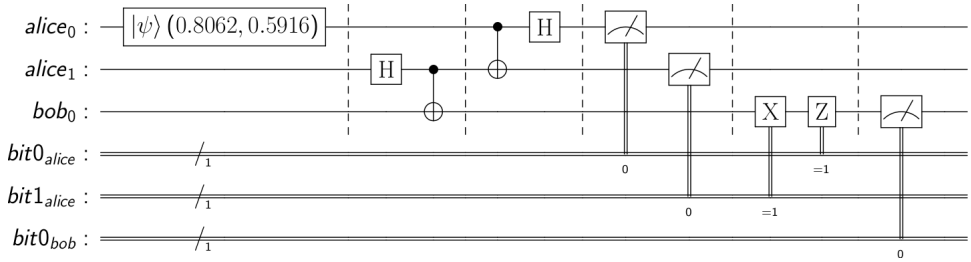


Figura 32 – Circuito quântico que executa um teletransporte quântico de um estado quântico $|\psi\rangle$.

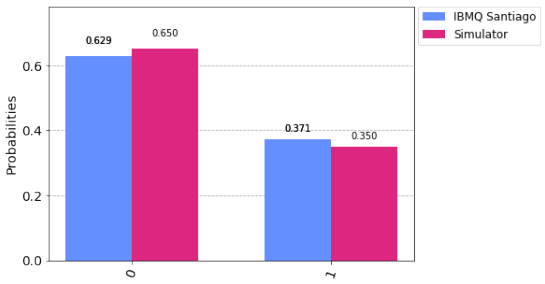


Figura 33 – Resultado de 8192 execuções do circuito quântico que realiza o teletransporte do estado $|\psi\rangle = \sqrt{0.65}|0\rangle + \sqrt{0.35}|1\rangle$ no simulador e na QPU (*Quantum Processor Unit*) IBMQ Santiago 5 qubits.

Aparentemente o teletransporte quântico parecer violar dois pontos importantes da física, o primeiro é o teorema da não clonagem e o segundo o axioma da relatividade

restrita que estabelece a velocidade da luz como velocidade limite no universo, no primeiro caso é simples de ver que não há violação, pois o estado final do qubit $A1$ de Alice é diferente do estado final do qubit de Bob, ou seja, os procedimentos realizados durante o protocolo não criaram e enviaram uma cópia de (3.154) a Bob. No segundo caso, precisamos fazer uso do formalismo de matrizes de densidade e da entropia de von Neumann para mostrar que não há transmissão de informação a uma velocidade superluminal, para demonstrar isso devemos considerar a matriz de densidade apresentada abaixo

$$\rho = |\Psi''\rangle\langle\Psi''|. \tag{3.162}$$

Tomando o traço parcial de ρ em relação aos graus de liberdade dos qubits de Alice, ou seja, em relação ao espaço de Hilbert $\mathcal{H}_A$, temos como resultado a matriz de densidade reduzida

$$\begin{aligned} \rho_B &= \mathrm{tr}_A(\rho) \\ &= \frac{1}{2}(|0_B\rangle\langle 0_B| + |1_B\rangle\langle 1_B|), \end{aligned} \tag{3.163}$$

a partir dela podemos determinar a entropia de von Neumann relacionada ao qubit de Bob e podemos constatar que este apresenta entropia máxima $S(\rho_B) = 1$, o que faz com que as medições de Bob sejam completamente aleatórias se este não receber as instruções de Alice via o canal de comunicação clássico, este fato derruba a aparente violação da relatividade restrita no protocolo de teletransporte quântico. Uma última informação que podemos extrair do teletransporte quântico é que ele enfatiza a troca entre diferentes recursos na mecânica quântica, neste caso é mostrado que um estado emaranhado e dois bits clássicos de comunicação formam um recurso igual a pelo menos um qubit de comunicação [1].

Listing 3.3 – Código em Qiskit que produz o circuito de teletransporte quântico da Figura 32.

```
from qiskit.circuit import ClassicalRegister, QuantumRegister, QuantumCircuit

alice = QuantumRegister(2,name='alice')
bob = QuantumRegister(1,name='bob')
bit_a0 = ClassicalRegister(1,name='bit0_{alice}')
bit_a1 = ClassicalRegister(1,name='bit1_{alice}')
bit_b0 = ClassicalRegister(1,name='bit0_{bob}')

qc=QuantumCircuit(alice,bob,bit_a0,bit_a1,bit_b0)

qc.initialize(np.array([np.sqrt(0.65),np.sqrt(0.35)]),alice[0])
qc.barrier()
qc.h(alice[1])
qc.cx(alice[1],bob[0])
qc.barrier()
qc.cx(alice[0],alice[1])
```

```
qc.h(alice[0])
qc.barrier()
qc.measure(alice[0],bita0)
qc.measure(alice[1],bita1)
qc.barrier()
qc.x(bob).c_if(bit_a1,1)
qc.z(bob).c_if(bit_a0,1)
qc.barrier()
qc.measure(bob,bit_b0)
```

3.6.3 Desigualdade CHSH

A desigualdade CHSH foi proposta por Clauser, Horne, Shimony e Holt e atualmente é uma das formulações mais populares de apresentação da desigualdade de Bell [43]. Nosso objetivo é realizar uma breve discussão sobre essa desigualdade e como hoje é possível testá-la usando os computadores quânticos que temos disponíveis atualmente [43, 44]. Para introduzirmos a desigualdade, vamos considerar o seguinte cenário: imagine que Alice e Bob compartilham um estado emaranhado bipartido igual a (2.180) e que cada um deles realiza duas medidas em duas bases diferentes em seus respectivos sistemas, considerando que as bases de Alice são A e a, e as de Bob B e b, sendo assim podemos definir uma quantidade $CHSH$, cujo valor médio queremos determinar e este pode ser calculado através da expressão

$$\langle CHSH\rangle = \langle AB\rangle - \langle Ab\rangle + \langle aB\rangle + \langle ab\rangle. \tag{3.164}$$

Considerando que a quantidade física que Alice e Bob estão medindo só possui duas possibilidades de resultados iguais a -1 ou 1, então temos que as quantidades $(B-b)$ e $(B+b)$ só podem ser iguais a 0 ou ± 2, o que implica que a quantidade $A(B-b)+a(B+b)$ só pode assumir os valores -2 ou +2, este fato estabelece um limite para o valor médio da quantidade $CHSH$ definida anteriormente, este limite é igual a

$$|\langle AB\rangle - \langle Ab\rangle + \langle aB\rangle + \langle ab\rangle| \leq 2. \tag{3.165}$$

No aparato experimental de Bob ele possui um dispositivo que altera o entre as bases de medida dele e de Alice, dessa maneira o valor médio das medidas A e B pode ser descrito como

$$\langle A\otimes B\rangle = \langle\psi_+|A\otimes B|\psi_+\rangle = -\cos\theta_{AB}. \tag{3.166}$$

Com isso, temos que o esquema de medição determinado por eles pode ser descrito como os circuitos da Figura 34, dessa forma podemos reproduzir o experimento de Alice e Bob nos computadores quânticos da seguinte forma: vamos usar 15 valores de θ no intervalo $[0, 2\pi)$ para as portas R_Y, com isso ao final teremos 60 circuitos que vão preparar o estado

emaranhado compartilhado por Alice e Bob, depois farão uma rotação no sistema de Bob e por fim farão medidas na base computacional e conjugada. Quando o valor de θ é igual a $\pi/4$ temos os seguintes operadores que definem as quantidades A, a, B e b

$$A = \frac{1}{\sqrt{2}}(\sigma_z - \sigma_x) \text{ e } a = \frac{1}{\sqrt{2}}(\sigma_z + \sigma_x) \tag{3.167}$$

$$B = \sigma_z \text{ e } b = \sigma_x, \tag{3.168}$$

este conjunto de operadores faz com que o valor médio da quantidade $CHSH$ seja igual a

$$|\langle CHSH \rangle| = 2\sqrt{2} > 2, \tag{3.169}$$

com isso temos um resultado que deixa explícita a violação da desigualdade de Bell. Na Figura 35 podemos ver os resultados obtidos nos experimentos conduzidos nos computadores quânticos Belém, Santiago e Lagos da IBM. No gráfico apresentado é possível ver que na região próxima de $\theta = \pi/4$ todos os dispositivos ultrapassam o limite imposto pelas suposições de localismo e realismo. Um fato que deve ser ressaltado aqui é que a união do acesso remoto aos computadores quânticos e saber criar um circuito quântico em Qiskit nos permite realizar uma verificação experimental, como a de Alain Aspect e seus colaboradores [42], do resultado teórico obtido Bell.

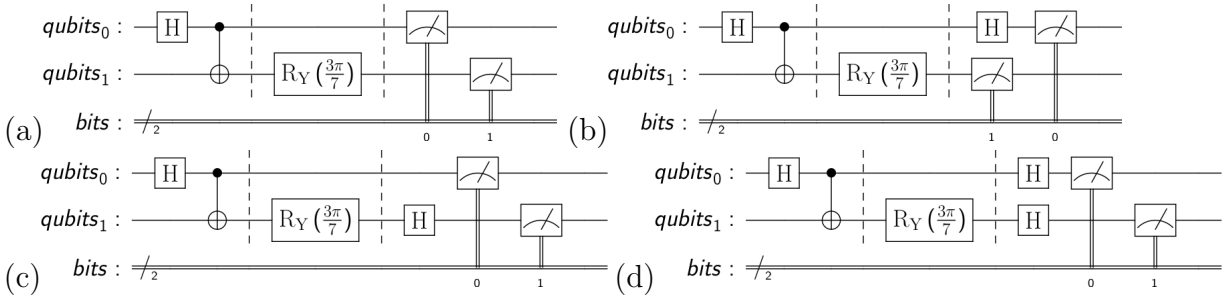


Figura 34 – Alguns dos circuitos quânticos utilizados para demonstrar a violação da desigualdade de Bell nos computadores quânticos Santiago, Belém e Lagos da IBM. No início de cada circuito é preparado o estado emaranhado (2.180), onde o qubit 0 representa o qubit de Alice e o qubit 1 o de Bob. Após essa etapa temos a aplicação de uma porta R_Y no qubit de Bob para realizar a rotação entre as bases de medida escolhidas por Alice e Bob. Nos circuitos presentes em (b), (c) e (d) temos a aplicação de portas Hadamard antes da medição, isso é feito para mudar a base de medição para os autoestados de σ_x, pois neste cenário Alice e Bob podem escolher medir se vão realizar medidas em termos dos autoestados de σ_z e σ_x.

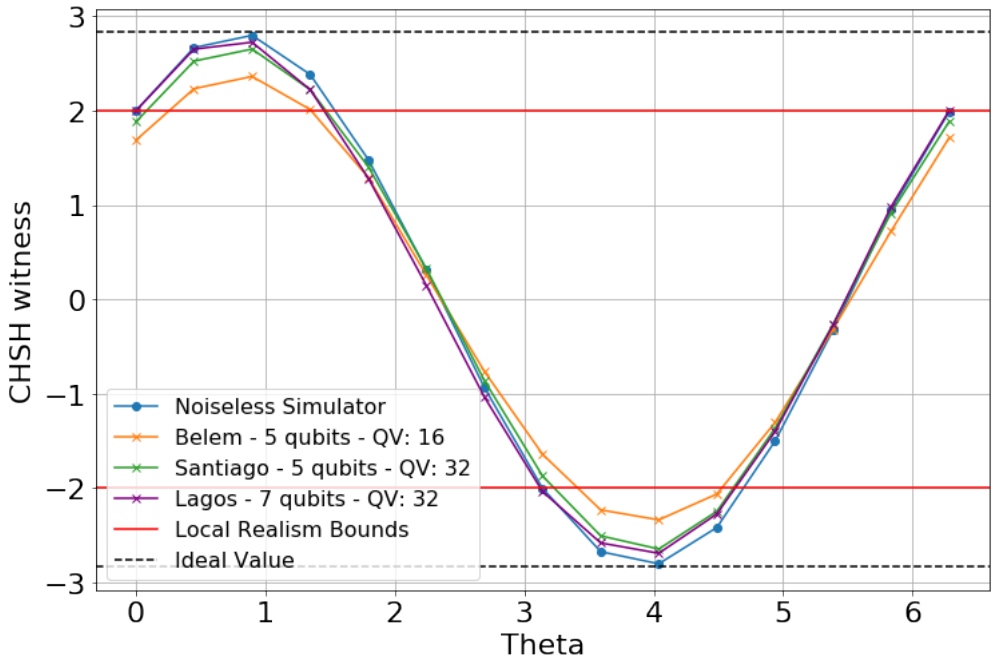


Figura 35 – Resultados das desigualdades CHSH para os computadores quânticos Santiago, Belém e Lagos da IBM.

Listing 3.4 – Código em Qiskit que produz os circuitos para realização do experimento de verificação da violação da desigualdade de Bell.

```
from qiskit.circuit import ClassicalRegister, QuantumRegister, QuantumCircuit
import numpy as np

angles = np.linspace(0, 2*np.pi,15)
chsh_circuits = []

qubits = QuantumRegister(2, name='qubits')
bits = ClassicalRegister(2, name='bits')

for theta in angles:
        for i in range(4):
                if i == 0:
                        qc = QuantumCircuit(qubits, bits)
                        qc.h(qubits[0])
                        qc.cx(qubits[0], qubits[1])
                        qc.ry(theta, qubits[1])
                        qc.measure()
                        chsh_circuits.append(qc)
                elif i == 1:
                        qc = QuantumCircuit(qubits, bits)
                        qc.h(qubits[0])
                        qc.cx(qubits[0], qubits[1])
                        qc.ry(theta, qubits[1])
                        qc.h(qubits[0])
```

```
            qc.measure()
            chsh_circuits.append(qc)
    elif i == 2:
            qc = QuantumCircuit(qubits, bits)
            qc.h(qubits[0])
            qc.cx(qubits[0], qubits[1])
            qc.ry(theta, qubits[1])
            qc.h(qubits[1])
            qc.measure()
            chsh_circuits.append(qc)
    else:
            qc = QuantumCircuit(qubits, bits)
            qc.h(qubits[0])
            qc.cx(qubits[0], qubits[1])
            qc.ry(theta, qubits[1])
            qc.h(qubits)
            qc.measure()
            chsh_circuits.append(qc)
```

3.6.4 Cálculo da Energia da Estrutura Hiperfina do Átomo de Hidrogênio

Outro problema que pode ser resolvido através da preparação de estados emaranhados, é o do cálculo das energias da estrutura hiperfina do átomo de hidrogênio e a determinação do comprimento de onda característico dessas energias. Este é um dos problemas de interesse da rádio astronomia e nos ajuda a obter informações sobre o hidrogênio contido em galáxias. Nossa discussão sobre a estrutura hiperfina do átomo de hidrogênio é a solução de um exercício da referência [43] e tem como base teórica o capítulo 12 da referência [56].

O átomo de hidrogênio tem em seu núcleo apenas um próton e nas vizinhanças deste, há um elétron em um dos níveis discretos de energia. O primeiro estado excitado está, aproximadamente, 10 eV acima do estado fundamental [56]. Devido a interações entre o spin do elétron e o spin do próton, o nível de energia do estado fundamental é dividido em 4 níveis de energia, 3 bem próximos e um mais isolado. Estes níveis estão relacionados às quatro diferentes configurações de orientação dos spins do elétron e do próton que são representadas pelos estados [56]

Estado 1 - elétron e próton com spin para cima: $|0_e 0_p\rangle$

Estado 2 - spin do elétron para cima e spin do próton para baixo: $|0_e 1_p\rangle$

Estado 3 - spin do elétron para baixo e spin do próton para cima: $|1_e 0_p\rangle$

Estado 4 - elétron e próton com spin para baixo: $|1_e 1_p\rangle$.

A Hamiltoniana que descreve nosso sistema de interesse pode ser escrita da

seguinte maneira [56]

$$H = E_0 + A\vec{\sigma^e} \cdot \vec{\sigma^p}, \tag{3.170}$$

onde E_0 é um termo constante que não possui relação com a estrutura hiperfina e depende do nível de energia que escolhemos medir. Já o segundo termo representa o operador

$$\vec{\sigma^e} \cdot \vec{\sigma^p} = \sigma_x^e \sigma_x^p + \sigma_y^e \sigma_y^p + \sigma_z^e \sigma_z^p, \tag{3.171}$$

e contêm toda a informação que queremos saber sobre a separação dos níveis de energia. Uma vez que desejamos calcular as energias associadas a estrutura hiperfina, devemos encontrar os estados que representam o sistema de interesse e usá-los para calcular o valor médio de (3.170) [43]

$$E = \langle H \rangle = A(\langle X_e X_p \rangle + \langle Y_e Y_p \rangle + \langle Z_e Z_p \rangle). \tag{3.172}$$

Os estados (2.180), (2.182), (2.181) e (2.183) representam as duas configurações de energia presentes no sistema, o tripleto (os três primeiros estados) e o singleto (o último estado). Estes são estados emaranhados que representam as interações entre os spins do elétron e do próton. Através deles podemos calcular os valores médios de $X_e X_p$, $Y_e Y_p$ e $Z_e Z_p$

$$\begin{aligned}
\langle X_e X_p \rangle &= \langle\psi|0_e 0_p\rangle\langle 1_e 1_p|\psi\rangle + \langle\psi|0_e 1_p\rangle\langle 1_e 0_p|\psi\rangle + \langle\psi|1_e 0_p\rangle\langle 0_e 1_p|\psi\rangle + \langle\psi|1_e 1_p\rangle\langle 0_e 0_p|\psi\rangle \\
\langle Y_e Y_p \rangle &= -\langle\psi|0_e 0_p\rangle\langle 1_e 1_p|\psi\rangle + \langle\psi|0_e 1_p\rangle\langle 1_e 0_p|\psi\rangle + \langle\psi|1_e 0_p\rangle\langle 0_e 1_p|\psi\rangle - \langle\psi|1_e 1_p\rangle\langle 0_e 0_p|\psi\rangle \\
\langle Z_e Z_p \rangle &= |\langle 0_e 0_p|\psi\rangle|^2 - |\langle 0_e 1_p|\psi\rangle|^2 - |\langle 1_e 0_p|\psi\rangle|^2 + |\langle 1_e 1_p|\psi\rangle|^2.
\end{aligned}$$

Com essas quantidades calculadas, é possível determinar as energias $E_{Singleto}$ (2.183) e $E_{Tripleto}$ através substituindo os resultados obtidos em (3.172), assim chegamos em

$$E_{Singleto} = -3A \tag{3.173}$$

$$E_{Tripleto} = A, \tag{3.174}$$

onde $A = 1.47 \times 10^{-6}\ eV$. Com essas informações em mãos, é possível determinar a diferença de energia entre os níveis de energia do singleto e do tripleto através de

$$\Delta E = |E_{Singleto} - E_{Tripleto}|. \tag{3.175}$$

O resultado acima pode ser usada para calcular a frequência do fóton que precisa ser emitido ou absorvido para realizar a transição entre os níveis através da expressão

$$f = \frac{\Delta E}{\hbar}, \tag{3.176}$$

uma vez determinada a frequência, podemos calcular o comprimento de onda associado a esse fóton através de

$$\lambda_{hf} = \frac{c}{f}, \tag{3.177}$$

cujo resultado deve ser $\lambda_{hf} = 21\ cm$. O comprimento de onda λ_{hf} é conhecido como a linha de 21 cm do átomo de hidrogênio, pois este é o comprimento de onda da radiação emitida ou absorvida pelo gás de hidrogênio presente nas galáxias. Com o auxílio de rádios telescópios que são capazes de captar o comprimento de onda 21 cm, é possível observar as velocidades e a localização das concentrações do gás de hidrogênio. Medindo a intensidade, temos uma estimativa da quantidade de hidrogênio presente e medindo o desvio na frequência, causado pelo efeito Doppler, podemos descobrir como o gás se movimenta na galáxia [56].

Para realizarmos essas contas usando a computação quântica, podemos fazer uso do *Qiskit* para escrever os códigos que irão gerar os circuitos apropriados para o problema. De início precisamos do código presente na Figura 36, este basicamente importa e define as ferramentas que vamos precisar para calcularmos as energias e o comprimento de onda característico da estrutura hiperfina do hidrogênio.

```
In [1]: import numpy as np
        from qiskit import Aer, execute, IBMQ
        from qiskit.circuit import ClassicalRegister, QuantumRegister, QuantumCircuit
        from qiskit.tools.visualization import plot_histogram, plot_bloch_multivector, plot_state_qsphere
        from qiskit.tools.monitor import job_monitor
        from qiskit.ignis.mitigation.measurement import (complete_meas_cal, CompleteMeasFitter)

In [2]: IBMQ.load_account()

Out[2]: <AccountProvider for IBMQ(hub='ibm-q', group='open', project='main')>

In [3]: shots = 2**14
        A = 1.47e-6 #unit of A is eV
        hbar = 4.1357e-15 #Planck's constant in eV
        c = 3e10 #cgs units
        E_sim = []

        qubits = QuantumRegister(2, name='qubits')
        bits = ClassicalRegister(2, name='bits')

In [4]: backend = Aer.get_backend('qasm_simulator')
```

Figura 36 – Trecho de código que importa os pacotes e as funções necessárias para o cálculo das energias da estrutura hiperfina do átomo de hidrogênio. Além disso, também temos a realização do *login* na plataforma da IBM e as definições das variáveis importantes para o problema.

Em seguida, devemos definir as medições nas bases de estados de X, Y e Z, pois vamos precisar medir os estados gerados pelos circuitos nessas 3 bases para calcularmos os valores médios dos operadores presentes na Hamiltoniana do problema. Para isso, devemos usar o código presente no *listing* 3.5, este irá gerar os circuitos presentes na Figura 37.

Listing 3.5 – Código que gera os circuitos de medida nas bases de estados de X, Y e Z.

```
measure_ZZ = QuantumCircuit(qubits, bits)
measure_ZZ.measure(qubits, bits)

measure_XX = QuantumCircuit(qubits, bits)
measure_XX.h(qubits)
measure_XX.measure(qubits, bits)
```

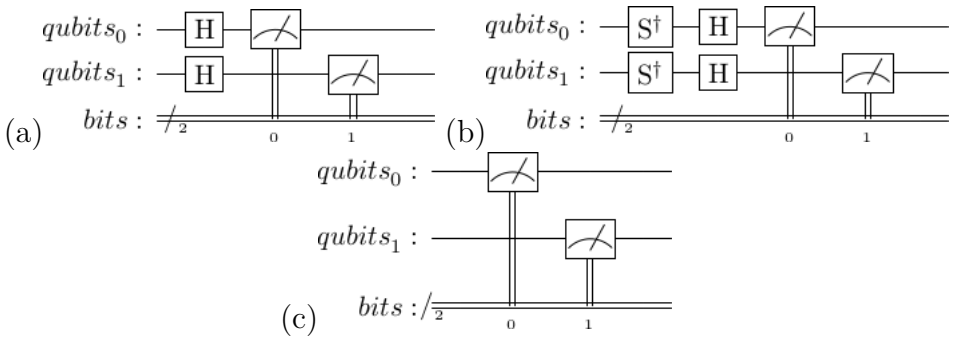


Figura 37 – Circuitos de medida nas bases: (a) X; (b) Y e (c) Z.

```
measure_YY = QuantumCircuit(qubits, bits)
measure_YY.sdg(qubits)
measure_YY.h(qubits)
measure_YY.measure(qubits, bits)
```

Os estados do tripleto ((2.180), (2.182) e (2.181)) e o estado do singleto (2.183) podem ser gerados através do trecho de código presente em *listing* 3.6.

Listing 3.6 – Código que gera os circuitos dos estados do tripleto e do estado do singleto.

```
Triplet1 = QuantumCircuit(qubits, bits)
Triplet1.h(qubits[0])
Triplet1.cx(qubits[0], qubits[1])

Triplet2 = QuantumCircuit(qubits, bits)
Triplet2.x(qubits[0])
Triplet2.h(qubits[0])
Triplet2.cx(qubits[0], qubits[1])

Triplet3 = QuantumCircuit(qubits, bits)
Triplet3.h(qubits[0])
Triplet3.x(qubits[1])
Triplet3.cx(qubits[0], qubits[1])

Singlet = QuantumCircuit(qubits, bits)
Singlet.x(qubits)
Singlet.h(qubits[0])
Singlet.cx(qubits[0], qubits[1])
```

No código presente na Figura 38, os circuitos definidos pelos códigos *listing* 3.5 e *listing* 3.6 serão combinados, de forma que garantimos que todos os estados do tripleto e o estado do singleto serão medidos em todas as bases especificadas. A partir dos resultados obtidos das 8192 execuções desses circuitos no simulador, são calculadas as energias da estrutura hiperfina do átomo de hidrogênio. Os valores obtidos podem ser vistos na Figura

38 ou na Tabela 10, note que estes são os valores exatos para o problema, pois a princípio estamos rodando em um simulador, que nada mais é do que um computador clássico imitando o comportamento de um computador quântico ideal.

Estado	Energia (eV)
$\|\psi_+\rangle$	1.470×10^{-6}
$\|\psi_-\rangle$	1.470×10^{-6}
$\|\phi_+\rangle$	1.470×10^{-6}
$\|\phi_-\rangle$	-4.410×10^{-6}

Tabela 10 – Valores exatos das energias da estrutura hiperfina do átomo de hidrogênio.

Também é possível realizar o cálculo das energias $E_{Singleto}$ e $E_{Tripleto}$ através de estados preparados em computadores quânticos reais, para fazer isso precisamos usar as linhas de código apresentadas na Figura 39. Dos resultados obtidos através das 8192 execuções dos circuitos de interesse, usando os qubits 3 e 4 do computador quântico Santiago, podemos calcular as energias através da função *Energy* definida no trecho de código presente na Figura 39. As energias calculadas podem ser vistas na mesma figura citada ou na Tabela 11. Note que os resultados apresentados não são os valores de referência, isso era de se esperar dado que os computadores quânticos atuais não contam com correções de erros e ainda possuem um nível de ruído que não pode ser desprezado, contudo existem ferramentas para mitigar os erros oriundos de erros de medição.

Estado	Energia (eV)
$\|\psi_+\rangle$	1.419×10^{-6}
$\|\psi_-\rangle$	1.376×10^{-6}
$\|\phi_+\rangle$	1.382×10^{-6}
$\|\phi_-\rangle$	-4.186×10^{-6}

Tabela 11 – Energias calculadas com os estados preparados no computador quântico Santiago.

Para aplicarmos uma mitigação de erros, precisamos executar alguns circuitos de calibração de medidas através de uma ferramenta implementada no *Qiskit*. Usando as linhas de código presentes na Figura 41, obtemos um filtro que deve ser aplicado aos resultados brutos obtidos anteriormente, com isso chegamos aos novos valores para as energias $E_{Singleto}$ e $E_{Tripleto}$ apresentados na Tabela 12.

Uma vez que dispomos de todos os resultados, podemos determinar o erro percentual em relação aos valores de referência, com isso teremos uma noção de o quão bem um computador quântico pode realizar o cálculo de interesse. Para o caso sem mitigação de erros, nossos resultados ficaram abaixo da faixa de $6,5\%$ de erro, como pode ser visto na

Estado	Energia (eV)
$\|\psi_+\rangle$	1.470×10^{-6}
$\|\psi_-\rangle$	1.470×10^{-6}
$\|\phi_+\rangle$	1.470×10^{-6}
$\|\phi_-\rangle$	-4.408×10^{-6}

Tabela 12 – Energias calculadas com os estados preparados no computador quântico Santiago após a aplicação da mitigação de erros.

Tabela 13, já com a mitigação os resultados são quase que exatos, pois todos são muitos próximos de 0%. Estes estão presentes na Tabela 14.

Erro percentual (sem mitigação de erros)
3,442%
6,396%
5,981%
5,086%

Tabela 13 – Erros percentuais das energias calculadas através dos resultados obtidos nas 8192 execuções, no computador quântico Santiago, das combinações dos circuitos gerados pelos trechos de código *listing* 3.5 e 3.6 .

Erro percentual (com mitigação de erros)
0%
0%
0%
0,041%

Tabela 14 – Erros percentuais após a aplicação da mitigação de erros nos resultados obtidos nas 8192 obtidos nas 8192 execuções, no computador quântico Santiago, das combinações dos circuitos gerados pelos trechos de código *listing* 3.5 e 3.6 .

In [14]:
```
def Calculate_Energy(Triplet1,Triplet2,Triplet3,Singlet,measure_XX,measure_YY,measure_ZZ,backend,shots,A,E_sim):
    for state_init in [Triplet1,Triplet2,Triplet3,Singlet]:
        Energy_meas = []
        for measure_circuit in [measure_XX,measure_YY,measure_ZZ]:
            qc = state_init.compose(measure_circuit, qubits, bits)
            counts = execute(qc,backend=backend,shots=shots).result().get_counts()
            probs = {}
            for output in ['00','01','10','11']:
                if output in counts:
                    probs[output] = counts[output]/shots
                else:
                    probs[output] = 0
            Energy_meas.append(probs['00']-probs['01']-probs['10']+probs['11'])

        E_sim.append(A*np.sum(np.array(Energy_meas)))
```

In [15]:
```
Calculate_Energy(Triplet1,Triplet2,Triplet3,Singlet,measure_XX,measure_YY,measure_ZZ,backend,shots,A,E_sim)
```

In [16]:
```
print('Energy expection value of the state Triplet1 : {:.3e} eV'.format(E_sim[0]))
print('Energy expection value of the state Triplet2 : {:.3e} eV'.format(E_sim[1]))
print('Energy expection value of the state Triplet3 : {:.3e} eV'.format(E_sim[2]))
print('Energy expection value of the state Singlet : {:.3e} eV'.format(E_sim[3]))
```

```
Energy expection value of the state Triplet1 : 1.470e-06 eV
Energy expection value of the state Triplet2 : 1.470e-06 eV
Energy expection value of the state Triplet3 : 1.470e-06 eV
Energy expection value of the state Singlet : -4.410e-06 eV
```

In [17]:
```
E_del = abs(E_sim[0]-E_sim[3])

f = E_del/hbar

wavelength = c/f

print('The wavelength of the radiation from the transition\
 in the hyperfine structure is : {:.1f} cm'.format(wavelength))
```

```
The wavelength of the radiation from the transition in the hyperfine structure is : 21.1 cm
```

Figura 38 – Trecho de código com a função usada para calcular a energia exata da estrutura hiperfina do átomo de hidrogênio usando o simulador *qasm_simulator* do *Qiskit* a partir dos circuitos gerados pelos códigos *listing* 3.5 e 3.6.

In [18]:
```
provider = IBMQ.get_provider('ibm-q')
qcomp = provider.get_backend('ibmq_santiago')
```

In [19]:
```
qc_all = [state_init+measure_circuit for state_init in [Triplet1,Triplet2,Triplet3,Singlet]
          for measure_circuit in [measure_XX, measure_YY, measure_ZZ] ]

shots = 8192
job = execute(qc_all,initial_layout=[3,4], backend=qcomp, optimization_level=3, shots=shots)
print(job.job_id())
job_monitor(job)
```

```
607279cdbb1d8817c508d0e6
Job Status: job has successfully run
```

In [20]:
```
#results = qcomp.retrieve_job('6045c84f1708848283289090').result()
results = job.result()
```

Figura 39 – Trecho de código que envia os circuitos para serem executados no computador quântico Santiago.

In [22]:
```python
def Energy(results, shots):
    """Compute the energy levels of the hydrogen ground state.

    Parameters:
        results (obj): results, results from executing the circuits for measuring a hamiltonian.
        shots (int): shots, number of shots used for the circuit execution.

    Returns:
        Energy (list): energy values of the four different hydrogen ground states
    """
    E = []
    A = 1.47e-6

    for ind_state in range(4):
        Energy_meas = []
        for ind_comp in range(3):
            counts = results.get_counts(ind_state*3+ind_comp)

            # calculate the probabilities for each computational basis
            probs = {}
            for output in ['00','01', '10', '11']:
                if output in counts:
                    probs[output] = counts[output]/shots
                else:
                    probs[output] = 0

            Energy_meas.append( probs['00'] - probs['01'] - probs['10'] + probs['11'] )

        E.append(A * np.sum(np.array(Energy_meas)))

    return E
```

In [22]:
```python
E = Energy(results, shots)

print('Energy expection value of the state Triplet1 : {:.3e} eV'.format(E[0]))
print('Energy expection value of the state Triplet2 : {:.3e} eV'.format(E[1]))
print('Energy expection value of the state Triplet3 : {:.3e} eV'.format(E[2]))
print('Energy expection value of the state Singlet : {:.3e} eV'.format(E[3]))
```

```
Energy expection value of the state Triplet1 : 1.419e-06 eV
Energy expection value of the state Triplet2 : 1.376e-06 eV
Energy expection value of the state Triplet3 : 1.382e-06 eV
Energy expection value of the state Singlet : -4.186e-06 eV
```

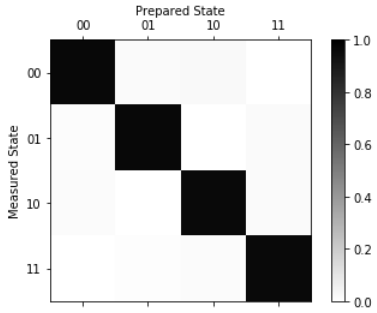
Figura 40 – Trecho de código que pega os resultados de preparação dos estados do tripleto e do estado do singleto no computador quântico Santiago, para calcular as energias da estrutura hiperfina do átomo de hidrogênio.

```
In [23]: cal_circuits, state_labels = complete_meas_cal(qr = 2, circlabel = 'measerrormitigationcal')

         cal_job = execute(cal_circuits, backend = qcomp, shots = 8192, optimization_level = 3)
         print(cal_job.job_id())
         from qiskit.tools.monitor import job_monitor
         job_monitor(cal_job)
         cal_results = cal_job.result()

         60728075826b0d7e5bcdf2a9
         Job Status: job has successfully run

In [24]: meas_fitter = CompleteMeasFitter(cal_results, state_labels)
         meas_filter = meas_fitter.filter
         meas_fitter.plot_calibration()
```




```
In [25]: results_new = meas_filter.apply(results)

In [26]: E_new = Energy(results_new, shots)

         print('Energy expection value of the state Tri1 : {:.3e} eV'.format(E_new[0]))
         print('Energy expection value of the state Tri2 : {:.3e} eV'.format(E_new[1]))
         print('Energy expection value of the state Tri3 : {:.3e} eV'.format(E_new[2]))
         print('Energy expection value of the state Sing : {:.3e} eV'.format(E_new[3]))

         Energy expection value of the state Tri1 : 1.470e-06 eV
         Energy expection value of the state Tri2 : 1.470e-06 eV
         Energy expection value of the state Tri3 : 1.470e-06 eV
         Energy expection value of the state Sing : -4.408e-06 eV
```

Figura 41 – Trecho de código que prepara um circuito de calibração para a técnica de mitigação de erros. No *print* do código podemos ver um gráfico mostrando o resultado da execução dos circuitos, onde temos uma comparação entre os estados que esperávamos medir e os estados medidos. Após o gráfico, temos um trecho de código que aplica a mitigação sobre os resultados obtidos na execução direta no computador quântico Santiago e que exibe o resultado final.

In [27]:
```
# results for the energy estimation from the simulation,
# execution on a quantum system without error mitigation and
# with error mitigation in numpy array format
Energy_exact, Energy_exp_orig, Energy_exp_new = np.array(E_sim), np.array(E), np.array(E_new)
```

In [28]:
```
# Calculate the relative errors of the energy values without error mitigation
# and assign to the numpy array variable `Err_rel_orig` of size 4
Err_rel_orig = ((Energy_exact-Energy_exp_orig)/Energy_exact)*100
```

In [29]:
```
# Calculate the relative errors of the energy values with error mitigation
# and assign to the numpy array variable `Err_rel_new` of size 4
Err_rel_new = ((Energy_exact-Energy_exp_new)/Energy_exact)*100
```

In [30]:
```
np.set_printoptions(precision=3)

print('The relative errors of the energy values for four bell basis\
 without measurement error mitigation : {}'.format(Err_rel_orig))
```

```
The relative errors of the energy values for four bell basis without measurement error mitigation : [3.442 6.396 5.981 5.086]
```

In [31]:
```
np.set_printoptions(precision=3)

print('The relative errors of the energy values for four bell basis\
 with measurement error mitigation : {}'.format(Err_rel_new))
```

```
The relative errors of the energy values for four bell basis with measurement error mitigation : [-2.396e-11 -2.991e-11 -2.936e
-11  4.100e-02]
```

Figura 42 – Trecho de código que faz o cálculo do erro percentual relativo para o caso em que há mitigação de erros e para o caso em que não há mitigação de erros.

4 Teoria Quântica de Correção de Erros

"We cannot clone, perforce; instead, we split
Coherence to protect it from that wrong
That would destroy our valued quantum bit
And make our computation take too long.

Correct a flip and phase - that will suffice.
If in our code another error's bred,
We simply measure it, then God plays dice,
Collapsing it to X or Y or Zed.

We start with noisy seven, nine, or five
And end with perfect one. To better spot
Those flaws we must avoid, we first must strive
To find which ones commute and which do not.

With group and eigenstate, we've learned to fix
Your quantum errors with our quantumt tricks."
- Daniel Gottesman

4.1 Introdução aos Erros na Computação Quântica

O cenário de erros na computação quântica é mais desafiador que o clássico, pois precisamos lidar com os novos tipos de erros, por exemplo erros ligados à fase relativa, e com a descoerência, que faz os qubits perderem suas características quânticas. Além disso, algumas técnicas para lidar com erros já conhecidas e desenvolvidas para a computação clássica não podem ser automaticamente reaproveitadas no contexto quântico, como no caso da técnica copiar bits para realizar redundância da informação visando protegê-la de ruído, pois como já vimos anteriormente as leis da mecânica quântica nos impedem de ter um operador que atue como uma copiadora universal de estados e devido a este fato precisamos recorrer a outros meios para realizar redundância da informação.

Antes de explorarmos mais a fundo os códigos quânticos de correção de erros, vamos realizar uma análise do impacto dos erros coerentes sobre os algoritmos quânticos. Os erros coerentes são a classe de erros que modificam o estado quântico do qubit sem causar perda de coerência no sistema, alguns exemplos de erros que pertencem a essa classe são: *bit-flip*, *phase-flip* e *bit-phase-flip*. A métrica que vamos usar para mensurar o

impacto de tais erros será o valor médio do operador Z. Faremos esta escolha porque a medição padrão na computação quântica é feita na base de estados de Z, já o cenário que vamos avaliar será a aplicação da porta X, então considerando que temos um qubit que é governado pela Hamiltoniana

$$H = \frac{\hbar\omega}{2}X. \tag{4.1}$$

A partir dela podemos determinar o operador de evolução temporal que dita a evolução do sistema

$$U(t) = e^{-\frac{i\omega t}{2}X}, \tag{4.2}$$

e usando a definição $\theta := \omega t$, temos que o operador acima se torna um operador de rotação em torno do eixo x e sua atuação sobre o estado $|0\rangle$ produz

$$|\psi(\theta)\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle - i\sin\left(\frac{\theta}{2}\right)|1\rangle. \tag{4.3}$$

Usando o estado acima para calcular o valor médio dos operadores X, Y e Z chegaremos aos seguintes resultados

$$\langle X\rangle_{\psi(\theta)} = \langle Y\rangle_{\psi(\theta)} = 0 \tag{4.4}$$

$$\langle Z\rangle_{\psi(\theta)} = \cos\theta. \tag{4.5}$$

Note que o valor médio de Z apresenta dependência do ângulo θ e por ser igual a função cosseno, este fica limitado entre -1 e 1, e só assume o valor 0 quando $\theta = \pi/2$, mas como neste primeiro momento estamos assumindo que a porta X é perfeita, não iremos registrar o valor médio de Z assumindo valores diferentes de -1 e 1.

Uma vez que os algoritmos quânticos, em geral, apresentam mais de uma porta, precisamos buscar compreender como os erros podem se acumular durante a sequência de operações, para isso supor uma sequência de d portas X

$$U_{\text{total}} = X^d = \underbrace{XX\ldots X}_{d \text{ vezes}}, \tag{4.6}$$

onde d representa a profundidade do circuito, lembrando que a porta X pode ser escrita como $R_X(\pi)$, então se realizarmos essa substituição no operador mostrado acima, teremos como resultado

$$U_{\text{total}} = [R_X(\pi)]^d = R_X(d\pi), \tag{4.7}$$

e aplicando o operador resultante da substituição sobre o estado $|0\rangle$, obtemos

$$\begin{aligned} |\psi_f\rangle &= U_{\text{total}}|0\rangle \\ &= \cos\left(\frac{d\pi}{2}\right)|0\rangle - i\sin\left(\frac{d\pi}{2}\right)|1\rangle. \end{aligned} \tag{4.8}$$

Do resultado acima, podemos concluir que

$$|\psi_f\rangle = \begin{cases} |0\rangle, & \text{se } d \text{ for par} \\ |1\rangle, & \text{se } d \text{ for ímpar} \end{cases}, \tag{4.9}$$

e se usarmos o estado $|\psi_f\rangle$ no cálculo do valor médio de Z, chegaremos ao resultado

$$\langle Z \rangle_{\psi_f} = \cos(d\pi) = (-1)^d, \tag{4.10}$$

onde é possível ver que o valor de $\langle Z \rangle_{\psi_f}$ depende da profundidade d do circuito e fica oscilando entre -1 e 1, como pode ser visto na Figura 43 (a).

Agora que já foi estabelecido o perfil do comportamento ideal da porta X, vamos adicionar um erro ε e ver como este impacta o funcionamento da porta, sendo assim vamos denotar a porta X imperfeita como $\tilde{X}$ e sendo escrita como

$$\tilde{X} := R_X(\pi + \varepsilon). \tag{4.11}$$

O cenário que estamos supondo é equivalente a problemas de calibração dos pulsos de microondas usados para controlar os transmons. A partir da relação

$$R_X(\theta + \phi) = R_X(\theta) R_X(\phi), \tag{4.12}$$

podemos reescrever $\tilde{X}$ da seguinte forma

$$\begin{aligned} \tilde{X} &= e^{-\frac{i(\pi+\varepsilon)}{2} X} \\ &= e^{-\frac{i\varepsilon}{2} X} e^{-\frac{i\pi}{2} X} \\ &= R_X(\varepsilon) R_X(\pi) \\ &= X_\varepsilon X. \end{aligned} \tag{4.13}$$

O resultado obtido, indica que podemos interpretar a porta $\tilde{X}$ como uma porta X perfeita seguida de uma rotação de ε graus em torno do eixo x, dessa maneira a aplicação de d portas $\tilde{X}$ pode ser resumida como o operador

$$\begin{aligned} \tilde{U}_{\text{total}} &= \tilde{X}^d \\ &= [R_X(\varepsilon) R_X(\pi)]^d \\ &= R_X(d\epsilon) R_X(d\pi) \\ &= R_X(d\epsilon) U_{\text{total}}. \end{aligned} \tag{4.14}$$

Aplicando $\tilde{U}_{\text{total}}$ sobre o estado inicial $|0\rangle$ teremos como resultado

$$|\tilde{\psi}_f\rangle = R_X(d\epsilon) \begin{cases} |0\rangle, & \text{se } d \text{ for par} \\ |1\rangle, & \text{se } d \text{ for ímpar} \end{cases}, \tag{4.15}$$

e utilizando o estado acima para determinar o valor médio de Z, chegaremos ao resultado

$$\langle Z \rangle_{\tilde{\psi}_f} = \cos(d\pi + d\varepsilon), \tag{4.16}$$

cujo comportamento pode ser visto na Figura 43 (b). Analisando o desvio entre $\langle Z\rangle_{\tilde{\psi}_f}$ e $\langle Z\rangle_{\psi_f}$ [57], é possível ver que esse tipo de erro coerente apresenta impacto quadrático no resultado final da computação, como mostrado abaixo

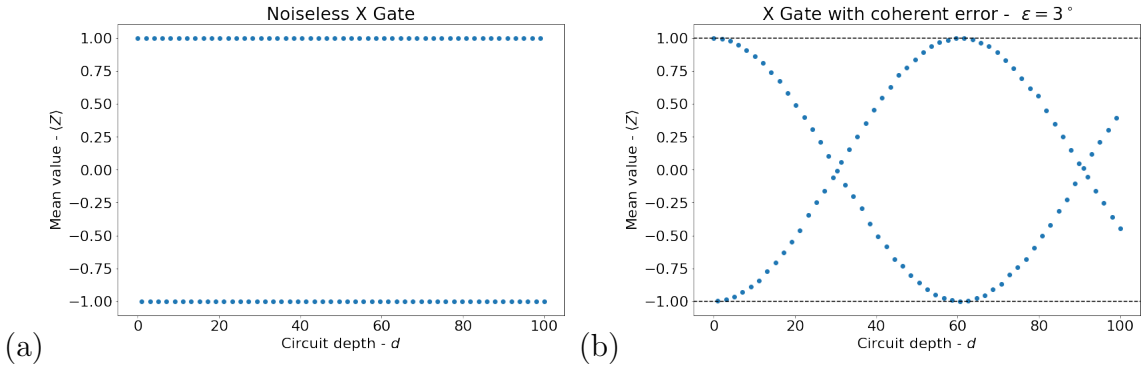
$$\begin{aligned}\langle Z\rangle_{\tilde{\psi}_f} - \langle Z\rangle_{\psi_f} &= \cos(d\pi + d\varepsilon) - \cos(d\pi) \\ &= \cos(d\pi)\cos(d\varepsilon) - \underbrace{\sin(d\pi)\sin(d\varepsilon)}_{=0} - \cos(d\pi) \\ &= (-1)^d(\cos(d\varepsilon) - 1) \simeq -\frac{1}{2}(-1)^d d^2\varepsilon^2 + \mathcal{O}(\varepsilon^3);\ \varepsilon \ll 1. \quad (4.17)\end{aligned}$$

É importante ressaltar que na era de dispositivos NISQ, a profundidade do circuito e a quantidade de portas CNOT presentes no circuito possuem muito impacto sobre a qualidade do resultado final da computação, portanto enquanto ainda não temos códigos quânticos de correção de erros implementados que garantam uma supressão exponencial dos erros, devemos tentar diminuir ao máximo os elementos mencionados. Outra métrica que podemos usar para analisar o impacto dos erros é a fidelidade média das portas [43]

$$F_{avg}(U, U') = \frac{|\mathrm{tr}(U^\dagger U')|^2/(2^m + 1)}{2^m + 1}, \quad (4.18)$$

onde U representa a unitária ideal (em nosso caso X), U' a porta executada (em nosso caso $\tilde{X}$) e m o número de qubits que sofrem a ação do operador. Na Figura 44 podemos ver o impacto da profundidade do circuito sobre a fidelidade da porta considerada em nosso modelo. Nas Figuras 45, 46 e 47, temos os resultados de experimentos feitos com os computadores quânticos Santiago, Lima e Lagos da IBM para verificar a qualidade da porta X aplicada sobre cada um dos qubits presentes nestas QPUs. As duas primeiras contam com 5 qubits cada e a última com 7 qubits. Os gráficos foram elaborados a partir dos valores médios de Z calculados a partir dos estados medidos nas 8192 execuções do circuito para cada uma das 100 profundidades do circuito. Dos gráficos podemos constatar a presença de erros incoerentes e erros de *readout*, o que faz com que o resultado de alguns qubits seja diferente do que era esperado de acordo com a Figura 43, contudo vemos também que alguns qubits apresentam comportamento muito próximo do esperado, o que sinaliza que estes possuem altos tempos de relaxação e de descoerência, além de uma baixa taxa de erros na porta X.

Na teoria clássica de correção de erros, o único tipo de erro que nós precisávamos nos preocupar era o erro de *bit-flip*, mas no contexto da computação quântica sabemos que um qubit pode assumir um contínuo de valores entre seus vetores da base, como pode ser visto em (3.1), este fato parece implicar que desenvolver códigos quânticos de correção de erros parece ser extremamente complicado, pois poderíamos ter que lidar com um número infinito de erros [58]. Como já foi dito anteriormente, os erros coerentes podem ser entendidos como alterações no estado quântico do qubit, de forma que se usarmos a representação da esfera de Bloch, veremos que eles deslocam o vetor de seu ponto original



(a) (b)

Figura 43 – Valor médio do operador Z para: (a) Estados da base computacional preparados por portas X perfeitas. (b) Estados da base computacional preparados por portas X com imperfeições causadas por erros coerentes que acrescentam $\varepsilon = 3°$ ao ângulo original da porta.

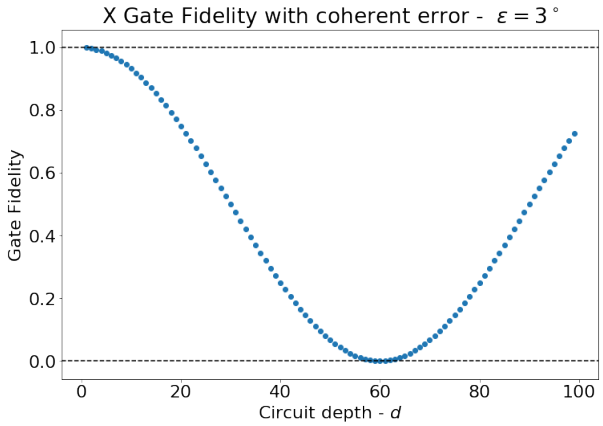


Figura 44 – Comportamento da fidelidade da porta X imperfeita de acordo com a profundidade do circuito.

e independente da fonte desse tipo de erro. Assim podemos modelá-lo matematicamente como o operador $U(\delta\theta, \delta\phi)$, cuja atuação sobre (3.1) produz o estado

$$U(\delta\theta, \delta\phi)|\psi\rangle = \cos\left(\frac{\theta + \delta\theta}{2}\right)|0\rangle + e^{i(\phi+\delta\phi)}\sin\left(\frac{\theta + \delta\theta}{2}\right)|1\rangle, \tag{4.19}$$

note que $\theta + \delta\theta$ e $\phi + \delta\phi$ representam as novas coordenadas do vetor na esfera de Bloch. Felizmente, os erros contínuos podem ser tratados através da utilização de um processo conhecido como digitalização de erros, este procedimento vem do fato que as matrizes que representam as portas que atuam sobre um qubit podem ser escritas usando a forma geral das matrizes do grupo de simetria SU(2) (3.6) [34], que por sua vez pode ser expandida em termos das matrizes geradoras do grupo que são as conhecidas matrizes de Pauli, portanto o operador $U(\delta\theta, \delta\phi)$ pode ser reescrito como

$$U(\delta\theta, \delta\phi)|\psi\rangle = \alpha\mathbb{I}|\psi\rangle + \beta X|\psi\rangle + \delta Y|\psi\rangle + \gamma Z|\psi\rangle. \tag{4.20}$$

Um detalhe importante que deve ser ressaltado, é que um código quântico que possui a

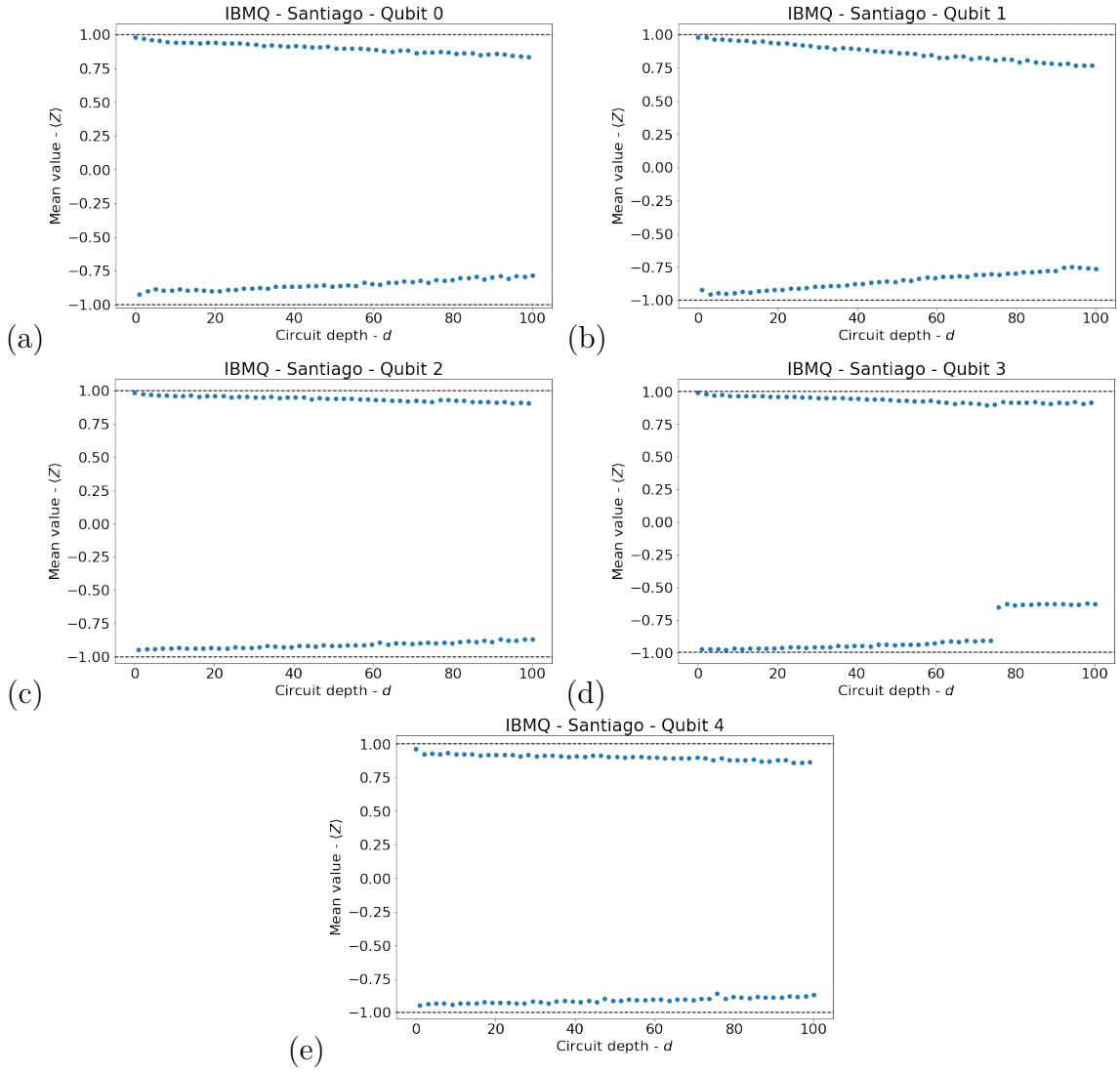


Figura 45 – Resultados do valor médio do operador Z calculado através da preparação dos estados na base computacional no computador quântico Santiago da IBM para os qubits: (a) Qubit 0; (b) Qubit 1; (c) Qubit 2; (d) Qubit 3 e (e) Qubit 4.

habilidade de corrigir erros do tipo X e Z pode corrigir qualquer erro coerente devido a digitalização de erros e as relações entre as matrizes de Pauli expostas no capítulo 2.

Uma vez superado o problema do contínuo de erros, devemos nos atentar a outros três problemas que surgem no contexto da teoria quântica de correção de erros, o primeiro é não podemos mais fazer cópias para realizar redundância da informação devido ao teorema da não clonagem, mais à frente veremos que a solução para essa questão é o emaranhamento de estados quânticos. O segundo problema é que os qubits estão sujeitos a *bit-flips*, *phase-flips* e combinação de ambos, portanto os códigos que devemos projetar devem ser capazes de lidar com todos esses tipos de erros. Por fim, temos o terceiro problema que é o colapso da função de onda, ou seja, não podemos medir diretamente o estado do qubit para obter informações sobre ele sem perder informação que estava

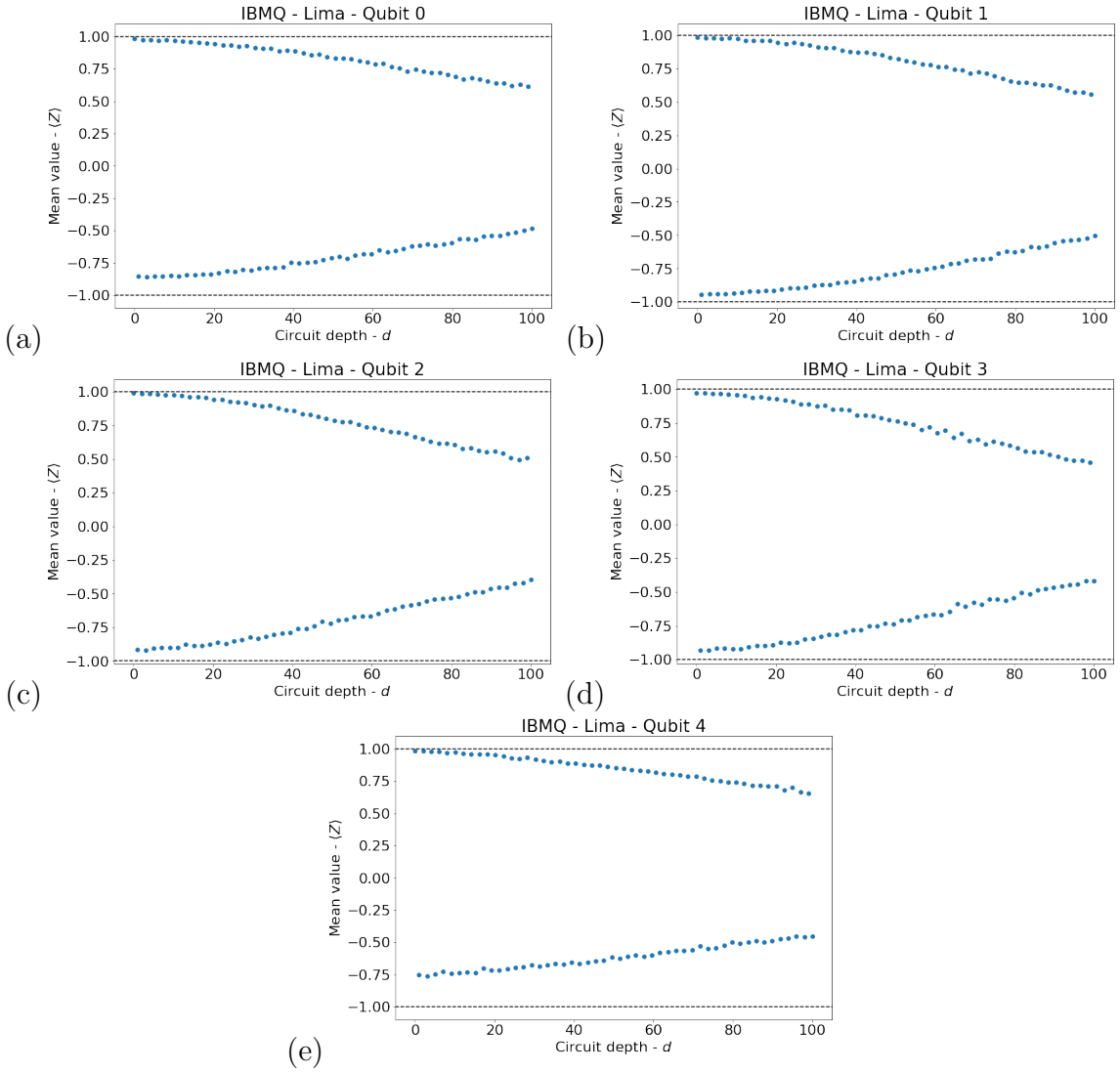


Figura 46 – Resultados do valor médio do operador Z calculado através da preparação dos estados na base computacional no computador quântico Lima da IBM para os qubits: (a) Qubit 0; (b) Qubit 1; (c) Qubit 2; (d) Qubit 3 e (e) Qubit 4.

contida no estado antes do colapso, para lidar com isso vamos usar qubits auxiliares e o formalismo de estabilizadores que foi introduzido por Daniel Gottesman [59].

Antes de prosseguirmos para as definições do formalismo de estabilizadores, vamos fazer uma breve discussão sobre os efeitos dos erros *bit-flip* e *phase-flip*, para podermos visualizar a atuação dos erros devemos considerar o estado $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. O primeiro erro a ser discutido será o *bit-flip*, lembrando que este também existe na computação clássica e que ele troca 0 por 1 e vice-versa, sendo assim queremos ver o que esse tipo de erro faz quando ocorre em um estado de superposição como $|\psi\rangle$, então considerando a ocorrência de tal erro no estado de interesse, temos que

$$X|\psi\rangle = \alpha|1\rangle + \beta|0\rangle. \tag{4.21}$$

Note que a ação do erro na superposição provoca uma troca das amplitudes de probabilidade

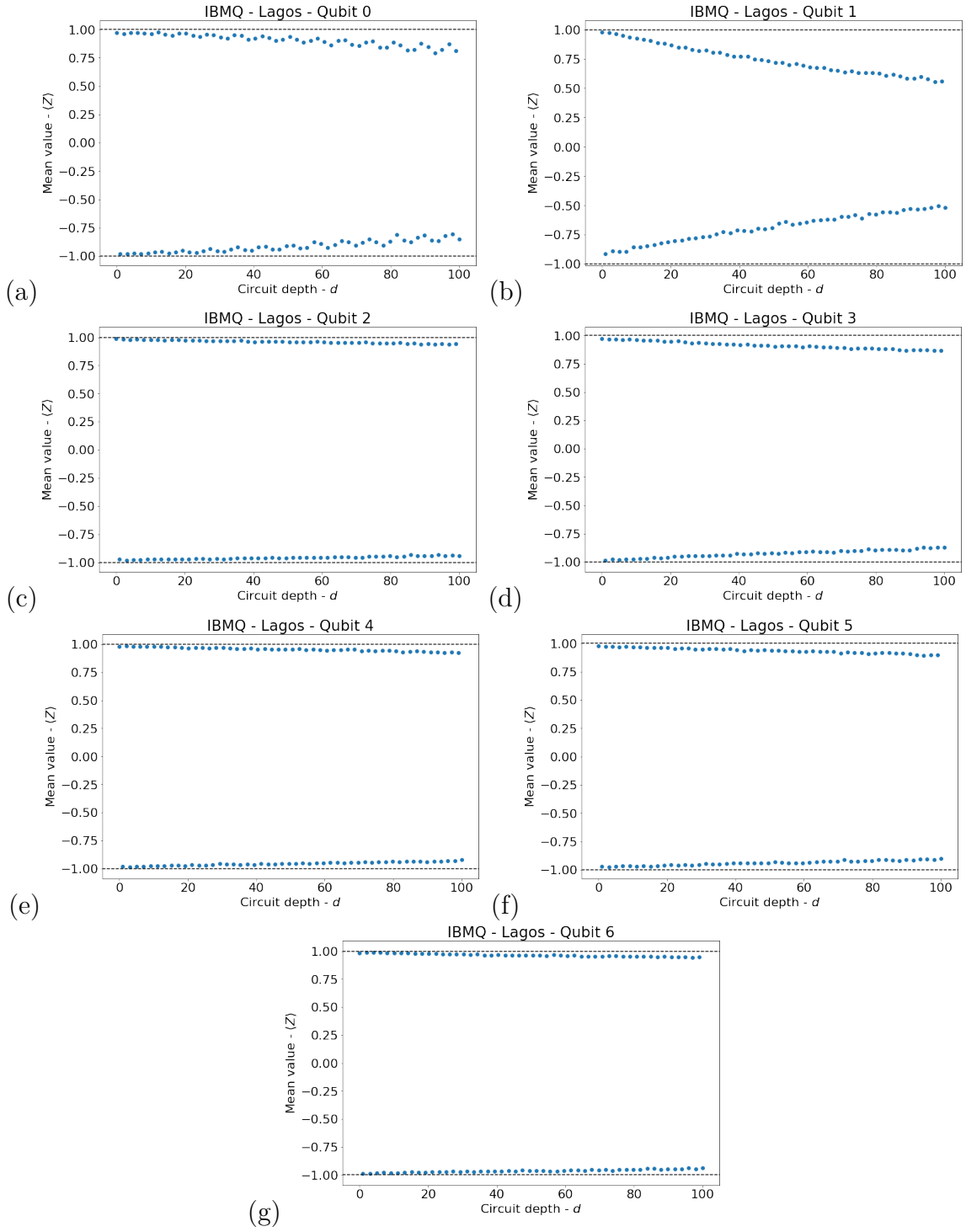


Figura 47 – Resultados do valor médio do operador Z calculado através da preparação dos estados na base computacional no computador quântico Lagos da IBM para os qubits: (a) Qubit 0; (b) Qubit 1; (c) Qubit 2; (d) Qubit 3; (e) Qubit 4; (f) Qubit 5 e (g) Qubit 6.

dos estados da base computacional, as consequências dessa troca são a mudança nas probabilidades de medição de cada um dos estados e uma rotação de π radianos do vetor de estado na esfera de Bloch. O erro *phase-flip* só ocorre na computação quântica, já que o bit clássico não apresenta fase, este tipo de erro só produz efeito quando atinge um estado

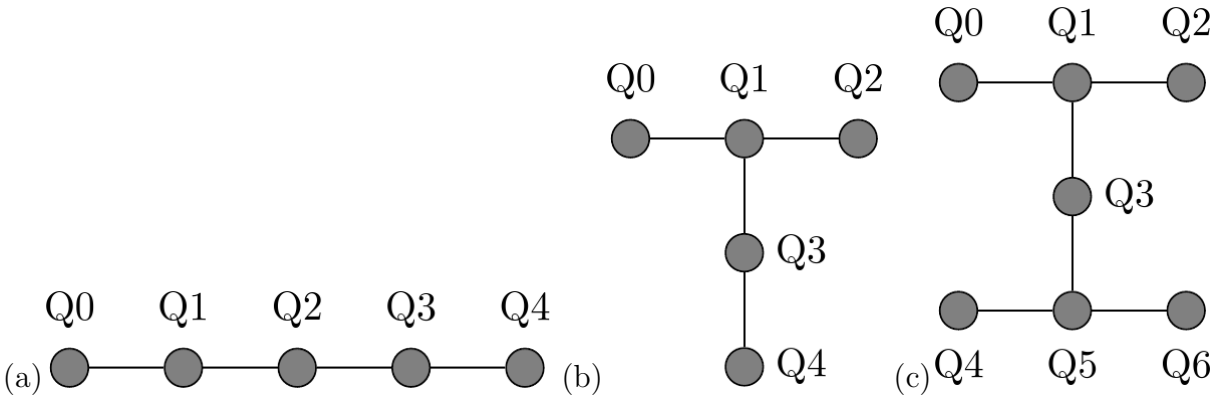


Figura 48 – Mapa de acoplamento dos qubits: (a) IBMQ Santiago; (b) IBMQ Belém e Lima e (c) IBMQ Lagos.

em superposição, pois cria uma fase relativa no estado, como exemplificado abaixo

$$Z|\psi\rangle = \alpha|0\rangle - \beta|1\rangle. \tag{4.22}$$

Por fim, é possível que ambos os erros ocorram sobre o mesmo estado, neste caso teremos a troca das amplitudes de probabilidade e o aparecimento de uma fase relativa, para vermos que isto é verdade vamos considerar que o estado $|\psi\rangle$ foi acometido primeiro pelo erro X e depois por Z, após a ação dos erros ficamos com o seguinte estado

$$ZX|\psi\rangle = \beta|0\rangle - \alpha|1\rangle. \tag{4.23}$$

É válido destacar que a combinação dos erros X e Z é equivalente a um erro do tipo Y a menos de uma fase.

4.2 Canais Quânticos

4.2.1 Bit-flip e Phase-flip

Uma das possíveis fontes de erros na computação quântica são as interações entre os qubits e o ambiente, sendo assim precisamos usar o formalismo de operações quânticas que foi apresentado no capítulo 2 para modelar erros dessa natureza.

O erro de *bit-flip* transforma um estado $|0\rangle$ em um estado $|1\rangle$ e vice-versa. Podemos modelar este erro através de uma operação quântica, para visualizarmos como isso é possível, vamos considerar que inicialmente o sistema de interesse seja representado pelo seguinte estado quântico

$$|\psi\rangle = |0_Q\rangle \otimes (\sqrt{p}|0_E\rangle + \sqrt{1-p}|1_E\rangle);\ 0 \leq p \leq 1, \tag{4.24}$$

onde o Q se refere ao qubit e E ao ambiente. Em dado momento, o ambiente interage com o qubit e provoca uma alteração em seu estado, aqui vamos considerar que esta interação

é semelhante a U_{CNOT}, com o qubit de controle sendo o estado do ambiente e o alvo o qubit do computador quântico, dessa maneira temos que o estado após a interação passa a ser escrito como

$$U_{CNOT}|\psi\rangle = \sqrt{p}|0_Q\rangle \otimes |0_E\rangle + \sqrt{1-p}|1_Q\rangle \otimes |1_E\rangle. \tag{4.25}$$

Considerando que ρ seja a matriz de densidade do qubit Q e que queremos estudar a dinâmica deste sistema após a interação com o ambiente, então devemos tomar o traço dos graus de liberdade da matriz de densidade gerada por $U_{CNOT}|\psi\rangle$, com isso definimos uma operação quântica que produz o seguinte resultado

$$\begin{aligned} \mathcal{E}(\rho) &= \mathrm{tr}_E(U_{CNOT}|\psi\rangle\langle\psi|U_{CNOT}^\dagger) \\ &= p|0_Q\rangle\langle 0_Q| + (1-p)|1_Q\rangle\langle 1_Q|. \end{aligned} \tag{4.26}$$

Do resultado acima podemos obter os operadores de Kraus que descrevem a dinâmica do qubit após a interação com o ambiente, estes são iguais a

$$K_0 = \sqrt{p}\mathbb{I} \text{ e } K_1 = \sqrt{1-p}\sigma_x, \tag{4.27}$$

dos operadores podemos ver que nosso sistema possui $1-p$ de probabilidade de sofrer um erro do tipo X. Na Figura 49 temos um circuito que simula um canal de *bit-flip* que atua sobre um código [[3,1,3]] que consegue corrigir um único *bit-flip*, o código para construir esse modelo encontra-se no apêndice C. De maneira análoga é possível definir o canal de *phase-flip*, lembrando que este tipo de erro causa uma mudança na fase relativa do estado quântico. Seguindo a receita do canal de *bit-flip*, vamos considerar um estado inicial parecido com o do caso anterior, a diferença é que o estado de nosso qubit agora encontra-se na base de σ_x, dessa maneira temos

$$|\phi\rangle = |+_Q\rangle \otimes (\sqrt{p}|0_E\rangle + \sqrt{1-p}|1_E\rangle), \tag{4.28}$$

dado que nosso interesse agora é modelar um *phase-flip*, o operador mais adequado para tal tarefa é a porta CZ, pois ela é capaz de alterar a fase relativa de um estado, com isso temos que o estado após a interação entre sistema e ambiente passa a ser descrito por

$$U_{CZ}|\phi\rangle = \sqrt{p}|+_Q\rangle \otimes |0_E\rangle + \sqrt{1-p}|-_Q\rangle \otimes |1_E\rangle. \tag{4.29}$$

Assumindo que σ representa a matriz de densidade do qubit antes da interação, temos que a operação quântica definida pela interação em questão modifica o estado inicial da seguinte maneira

$$\begin{aligned} \mathcal{E}(\sigma) &= \mathrm{tr}_E(U_{CZ}|\phi\rangle\langle\phi|U_{CZ}^\dagger) \\ &= p|+_Q\rangle\langle +_Q| + (1-p)|-_Q\rangle\langle -_Q|, \end{aligned} \tag{4.30}$$

e do resultado acima podemos concluir que os operadores de Kraus que modelam o canal de *phase-flip* são iguais a

$$K_0 = \sqrt{p}\mathbb{I} \text{ e } K_1 = \sqrt{1-p}\sigma_z. \tag{4.31}$$

Por fim, é possível combinar os dois erros, neste caso temos um canal que é conhecido como *bit-phase-flip*, cuja interação entre sistema e ambiente pode ser modelada como uma porta Y controlada, usando as mesmas ideias apresentadas anteriormente é possível obter os operadores de Kraus apresentados abaixo

$$K_0 = \sqrt{p}\mathbb{I} \text{ e } K_1 = \sqrt{1-p}\sigma_y. \tag{4.32}$$

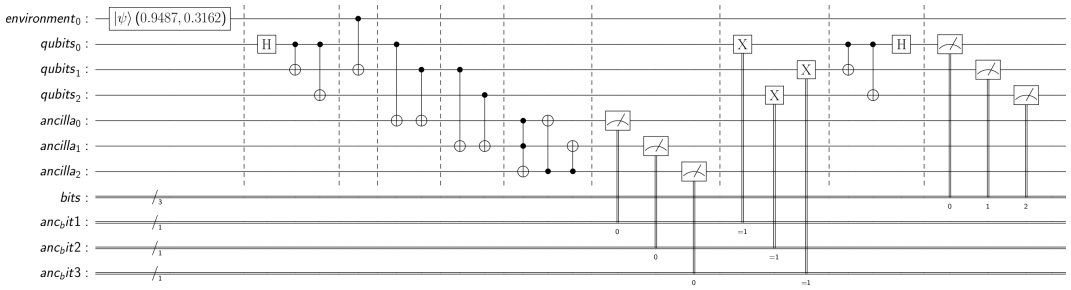


Figura 49 – Circuito que representa um canal de *bit-flip* onde a probabilidade de ocorrer este erro é $p = 0.1$.

4.2.2 Depolarizing

O segundo tipo canal que vamos discutir é o de *depolarizing*, cujo efeito é transformar o estado de um qubit, que inicialmente é puro, em um estado completamente misto, ou seja, $\rho = (1/2)\mathbb{I}$. Neste cenário o sistema sai de uma entropia nula e termina com máxima entropia, fazendo com que nosso grau de ignorância acerca dos graus de liberdade desse sistema seja o maior possível, fazendo com que este seja completamente aleatório. O canal de *depolarizing* pode ser modelado da seguinte forma [1]

$$\mathcal{E}(\rho) = p\frac{\mathbb{I}}{2} + (1-p)\rho. \tag{4.33}$$

Observe que da expressão acima podemos ver que a medida que a probabilidade de ocorrência p deste erro aumenta, temos uma contração uniforme da esfera de Bloch do estado inicial, cuja interpretação é um aumento de entropia, pois dado que inicialmente o sistema era puro, ou seja, seu estado estava na superfície da esfera de Bloch. Para obtermos os operadores de Kraus que modelam o canal em questão, precisamos realizar algumas transformações, primeiro devemos considerar que a matriz de densidade ρ do qubit pode ser escrita como

$$\rho = \frac{1}{2}(\mathbb{I} + \vec{r}\cdot\vec{\sigma}), \tag{4.34}$$

com isso temos que a relação abaixo é válida [1]

$$\frac{\mathbb{I}}{2} = \frac{\rho + \sigma_x\rho\sigma_x + \sigma_y\rho\sigma_y + \sigma_z\rho\sigma_z}{4}. \tag{4.35}$$

Substituindo o resultado acima em (4.33) obtemos a equação

$$\mathcal{E}(\rho) = \left(1 - \frac{3p}{4}\right)\rho + \frac{p}{4}(\sigma_x\rho\sigma_x + \sigma_y\rho\sigma_y + \sigma_z\rho\sigma_z), \tag{4.36}$$

que nos permite obter os operadores de Kraus

$$K_0 = \sqrt{1 - \frac{3p}{4}}\mathbb{I} \tag{4.37}$$

$$K_i = \sqrt{p}\frac{\sigma_i}{2};\ i = 1, 2 \text{ e } 3;\ \sigma_1 = \sigma_x; \sigma_2 = \sigma_y \text{ e } \sigma_3 = \sigma_z. \tag{4.38}$$

Entretanto a representação dada acima não é a mais conveniente para se trabalhar, em geral optamos por fazer a troca $p \to (4/3)p$ de forma a obter

$$\mathcal{E}(\rho) = (1-p)\rho + \frac{p}{3}\left(\sigma_x\rho\sigma_x + \sigma_y\rho\sigma_y + \sigma_z\rho\sigma_z\right). \tag{4.39}$$

Uma característica interessante do canal de *depolarizing* é que ele pode ser facilmente generalizado para sistemas quânticos com dimensão maior que $d = 2$. Diante disso, temos que em um sistema quântico d-dimensional, o canal de depolarizing atua de forma a fazer com que o estado inicial do sistema seja substituído por um estado completamente misto, ou seja, $\rho = (1/d)\mathbb{I}$ com probabilidade p, cujo modelo em termos de operação quântica pode ser descrito como

$$\mathcal{E}(\rho) = \frac{p\mathbb{I}}{d} + (1-p)\rho. \tag{4.40}$$

4.3 Formalismo de Estabilizadores e Códigos Quânticos

Para definirmos os estabilizadores, precisaremos trabalhar com os elementos do grupo de Pauli $\mathcal{G}_n$, a definição deste grupo para o caso em que temos n qubits é dada por [36]

$$\mathcal{G}_n := \{\pm 1, \pm i\} \times \{\mathbb{I}, X, Y, Z\}^{\otimes n}. \tag{4.41}$$

Um elemento do grupo de Pauli é chamado de produto de Pauli. Considerando um cenário em que dispomos de 2 qubits, então o grupo de Pauli $\mathcal{G}_2$ é dado por

$$\mathcal{G}_2 := \{\pm 1, \pm i\} \times \{\mathbb{II}, \mathbb{I}X, \mathbb{I}Y, \mathbb{I}Z, X\mathbb{I}, XY, XZ, Y\mathbb{I}, YX, YZ, Z\mathbb{I}, ZX, ZY, ZZ\}. \tag{4.42}$$

Note que em vez de usarmos a notação usual de produto tensorial, ou seja, $X \otimes Y$, estamos usando a notação simplificada XY, esta será a notação a ser utilizada daqui para frente para simplificar a escrita dos códigos quânticos de correção de erros e a operação com os

estabilizadores. Outro ponto importante a ser destacado sobre a notação, é que na notação simplificada fica implícita o produto tensorial com a identidade relacionada ao vetores dos espaços de Hilbert que não serão modificados pelo operador, por exemplo se temos 3 qubits e o operador simplificado é dado por Z_1Z_2, entendemos que o operador original seria $Z_1 \otimes Z_2 \otimes \mathbb{I}_3$.

Os estabilizadores P_i de um código $[[n, k, d]]$[1], formam um subgrupo Abeliano $\mathcal{S}$ do grupo de Pauli e satisfazem as condições [58]

$$\mathcal{S} = \{P_i \in \mathcal{G}_n \mid P_i|\psi_L\rangle = (+1)|\psi_L\rangle \ \forall \ |\psi_L\rangle \wedge [P_i, P_j] = 0 \ \forall \ i, j\}, \tag{4.43}$$

onde $|\psi_L\rangle$ representa um estado lógico do qubit que pertence ao espaço do código $\mathcal{C}_{[[n,k,d]]}$. A comutação dos estabilizadores é exigida para que estes possam ser medidos simultaneamente sem que exista algum tipo de incerteza, isso nos auxilia a construir um mecanismo de identificação de erros. É importante ressaltar que quando $\mathcal{S}$ é um conjunto mínimo, ou seja, nenhum de seus elementos pode ser escrito como produto de outros elementos do grupo, por exemplo: o conjunto $\mathcal{S} = \{Z_1Z_2, Z_2Z_3, Z_1Z_3\}$ não é mínimo, pois Z_1Z_3 pode ser escrito como produto de Z_1Z_2 e Z_2Z_3, mas se retirarmos Z_1Z_3 o conjunto passa a ser mínimo e denotado por $\mathcal{S} = \langle Z_1Z_2, Z_2Z_3\rangle$[2], mais adiante veremos que o conjunto utilizado neste exemplo está relacionado com o código de três qubits para *bit-flip*. Se um conjunto for mínimo, então temos que o número $m = n - k$ de elementos presentes no conjunto determina a quantidade de qubits auxiliares que serão usados e as síndromes presentes no código, outro ponto importante é que a atuação dos estabilizadores P_i é análoga a da matriz de paridade definida no capítulo 1, ou seja, nos auxilia a encontrar eventuais erros que tenham corrompido o estado original. Note também que o produto de estabilizadores P_iP_j também é um estabilizador, pois

$$P_iP_j|\psi_L\rangle = P_i(+1)|\psi_L\rangle = (+1)|\psi_L\rangle. \tag{4.44}$$

Através dos estabilizadores é possível criar uma forma geral de circuitos de codificação para a preparação de estados lógicos, para qualquer código estabilizador $[[n, k, d]]$ é possível construir o estado lógico $|0_L\rangle$ da seguinte forma [58]

$$|0_L\rangle = \frac{1}{N} \prod_{P_i \in \langle S\rangle} (\mathbb{I}^{\otimes n} + P_i)|0^{\otimes n}\rangle, \tag{4.45}$$

onde $\langle S\rangle$ é o conjunto mínimo de estabilizadores e $1/N$ é a constante de normalização. Também é possível codificar o estado através das operações do grupo de Clifford, neste caso precisamos usar as relações (3.38), (3.39), (3.40) e (3.41), mais à diante no texto será dado um exemplo de como realizar a codificação através de portas do grupo de

[1] Para distinguir códigos quânticos dos códigos clássicos usamos a notação com dois colchetes.

[2] Para diferenciar um conjunto mínimo dos demais conjuntos, usamos a notação $\langle . \rangle$ em vez da notação usual $\{.\}$

Clifford. Como foi dito anteriormente, o conjunto mínimo de estabilizadores nos permite definir as síndromes a serem aplicadas no circuito para detectar erros. O número mínimo de estabilizadores também dita a quantidade de qubits auxiliares que vamos usar para armazenar o resultado das síndromes, é necessário fazer isso para saber se um erro ocorreu nos qubits codificados sem realizar uma medida direta neles, essa técnica é extremamente importante, pois se a medida fosse realizada nos qubits codificados parte da informação contida neles seria perdida, sendo assim temos que o processo de síndrome nos códigos quânticos pode ser representado como

$$E|\psi_L\rangle \otimes |0_A^{(i)}\rangle \text{ Síndrome} \rightarrow \frac{1}{2}(\mathbb{I}^{\otimes n} + P_i)E|\psi_L\rangle \otimes |0_A^{(i)}\rangle + \frac{1}{2}(\mathbb{I}^{\otimes n} - P_i)E|\psi_L\rangle \otimes |1_A^{(i)}\rangle. \quad (4.46)$$

Note que a expressão acima revela que se o estabilizador P_i comutar com o erro E teremos o valor 0 como resultado da medida do qubit ancilla, mas se o comutador entre o estabilizador e o erro for diferente de zero,, então o valor a ser medido será igual a 1, o que indicará a presença de um erro, portanto podemos concluir que um bom código deve ter estabilizadores que não comutam com os erros para que possamos detectá-los [58].

Outro conjunto importante para a definição de um código quântico é o conjunto $\mathcal{L}$ dos operadores lógicos, este conjunto possui $2k$ operadores lógicos de Pauli $\bar{L}_i$ que permitem modificar os estados lógicos sem que seja necessário realizar uma decodificação e uma recodificação, além disso esse conjunto define o espaço vetorial dos vetores pertencentes ao código $\mathcal{C}_{[[n,k,d]]}$. O conjunto $\mathcal{L}$ dos operadores lógicos é formado pelos $2k$ operadores $\bar{L}_i$ que satisfazem as propriedades

$$\begin{aligned} \mathcal{L} \;=\; & \{\bar{L}_i \in \mathcal{G}_n \mid \bar{L}_i|\psi_L\rangle = |\phi_L\rangle \in \mathcal{C}_{[[n,k,d]]} \;\forall\; |\psi_L\rangle \in \mathcal{C}_{[[n,k,d]]} \;\wedge\; \{\bar{L}_i, \bar{L}_j\}_+ = 0 \;\forall (i,j) \\ & \wedge \quad [\bar{L}_i, P_j] = 0 \;\forall\; P_j \in \mathcal{S}\}. \end{aligned} \quad (4.47)$$

Uma propriedade importante dos operadores lógicos $\bar{L}_i$ é que o produto de qualquer um deles com um estabilizador também é um operador lógico, pois

$$\bar{L}_i P_j |\psi_L\rangle = \bar{L}_i |\psi_L\rangle. \quad (4.48)$$

Outro aspecto importante sobre os operadores lógicos, é que a menor quantidade de operadores de Pauli nos L_i define a distância do código estabilizador $\mathcal{C}_{[[n,k,d]]}$, por exemplo: se $\bar{X} = X_1X_2X_3$ e $\bar{Z} = Z_1Z_2Z_3$ são os operadores lógicos do código $[[3,1,d]]$ que apresenta o conjunto de estabilizadores $\mathcal{S} = \langle Z_1Z_2, Z_2Z_3\rangle$, então podemos concluir que a distância d deste código deve ser $d = 3$, pois a menor quantidade de operadores de Pauli em $\bar{X}$ e $\bar{Z}$ é igual a 3, portanto o código em questão seria um código [[3,1,3]].

Uma vez que temos o conjunto mínimo de estabilizadores estabelecido e se a distância do código for $d \geq 3$, então podemos usar as síndromes e os qubits auxiliares, como mostrado em (4.46), para definir o processo de decodificação, onde iremos buscar a melhor operação de recuperação $\mathcal{R}$, de acordo com as síndromes, que fará o estado

codificado $|\psi_L\rangle$ retornar ao espaço vetorial do código. Para que o processo de decodificação seja considerado bem sucedido, é preciso que a equação abaixo seja satisfeita

$$\mathcal{R}E|\psi_L\rangle = (+1)|\psi_L\rangle, \tag{4.49}$$

note que a equação acima é satisfeita quando o produto $\mathcal{R}E$ é igual a

$$\mathcal{R}E = \mathbb{I}\ (\mathcal{R} = E^{\dagger}) \text{ ou } \mathcal{R}E = P_i \in \mathcal{S}. \tag{4.50}$$

O primeiro caso é a solução trivial, já no segundo caso temos o produto $\mathcal{R}E$ igual a estabilizador do código, o fato de $\mathcal{R}$ não apresentar solução única implica na possibilidade de um código ser degenerado, ou seja, possuir vários erros mapeados em uma mesma síndrome. Em geral, buscamos criar códigos não degenerados para que seja possível ter um mapeamento um a um entre erro e síndrome, de forma a possibilitar que os erros sejam corrigidos, mas mesmo em casos em que o código é degenerado, pode ser possível corrigir erros, um exemplo é o código de Shor, este apresenta degenerescência nas síndromes relacionadas a *phase-flips*, mas como vamos ver mais à frente no texto, este fato não nos impossibilita de corrigir tais erros. Dizemos que o processo de decodificação falha quando o produto $\mathcal{R}E$ resulta em um operador lógico, como mostrado na equação abaixo

$$\mathcal{R}E = \bar{L}_i \in \mathcal{L}, \tag{4.51}$$

isto causaria uma falha de decodificação, pois levaria a outro estado do código devido ao operador lógico $\bar{L}_i$.

Para deixar os conceitos apresentados aqui mais claros, vamos usá-los em dois casos simples que serão reaproveitados na construção do código de Shor, o primeiro caso que vamos discutir será o código [[3,1,3]] para *bit-flip*, cujo conjunto mínimo de estabilizadores é dado por

$$\mathcal{S}_{bit-flip} = \langle Z_1Z_2, Z_2Z_3\rangle.$$

No caso clássico os bits lógicos eram feitos a partir de cópias do estado original, mas como o teorema da não clonagem nos coloca limitações na realização de redundância da informação através de cópias, devemos usar estados emaranhados para distribuir a informação entre as partes presentes no estado,, com isso temos uma forma de redundância no contexto quântico. Os qubits lógicos $|0_L\rangle$ e $|1_L\rangle$ do código em questão são iguais aos estados emaranhados

$$|0_L\rangle = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) \text{ e } |1_L\rangle = \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle), \tag{4.52}$$

observe que os estados acima indicam que estamos trabalhando com um espaço de Hilbert do tipo $\mathcal{H} = \mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2$ que apresenta dimensão $d = 8$, sendo assim vamos particionar

este espaço em quatro subespaços que serão usados para definir o espaço do código $\mathcal{C}_{\textit{bit-flip}}$ e os espaços dos erros $\mathcal{F}_i$, como pode ser visto abaixo

$$\mathcal{C}_{\textit{bit-flip}} = \{|000\rangle, |111\rangle\} \tag{4.53}$$
$$\mathcal{F}_1 = \{|100\rangle, |011\rangle\} \tag{4.54}$$
$$\mathcal{F}_2 = \{|010\rangle, |101\rangle\} \tag{4.55}$$
$$\mathcal{F}_3 = \{|001\rangle, |110\rangle\}, \tag{4.56}$$

perceba que os índices $i = 1, 2$ e 3 representam a posição (da direita para à esquerda) em que um *bit-flip* ocorreu nos estados do espaço $\mathcal{C}_{\textit{bit-flip}}$. Para visualizarmos como todo o processo de correção de erros ocorre, vamos usar $\mathcal{C}_{\textit{bit-flip}}$ como exemplo e escrever cada etapa envolvida no processo explicitamente e identificá-la com o circuito presente na Figura 50, sendo assim de início temos a codificação, ou seja, a transformação dos qubits em um qubit lógico para proteger a informação de erros, neste exemplo vamos considerar o qubit lógico $|0_L\rangle$ que pode ser produzido da seguinte forma

$$|0_L\rangle \otimes |0_{A1}0_{A2}\rangle = \Lambda_{(1,3)}(X)\Lambda_{(1,2)}(X)H_1|000\rangle \otimes |0_{A1}0_{A2}\rangle, \tag{4.57}$$

onde $\Lambda_{(c,t)}(X)$ representa a porta CNOT com c representando o qubit de controle e t o qubit alvo. Observe que a ordem de atuação dos operadores no circuito 50 é da esquerda para à direita, mas na representação de estados é o inverso, outro ponto a ser notado é que os qubits estão acoplados aos qubits auxiliares $|00_A\rangle$. Ao final da etapa de codificação, considerando que erros ocorrem durante o processo, devemos ter o seguinte estado quântico

$$|0_L\rangle \otimes |0_{A1}0_{A2}\rangle = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) \otimes |0_{A1}0_{A2}\rangle, \tag{4.58}$$

considerar que após o processo de codificação um *bit-flip* ocorra na posição 2 do estado $|0_L\rangle$, conforme Figura 50, perceba que após a ação do erro, o estado resultante não mais faz parte de $\mathcal{C}_{\textit{bit-flip}}$, mas sim do espaço de erro $\mathcal{F}_2$, sendo assim temos que o estado original transforma-se em

$$X_2|0_L\rangle \otimes |0_{A1}0_{A2}\rangle = \frac{1}{\sqrt{2}}(|010\rangle + |101\rangle) \otimes |0_{A1}0_{A2}\rangle. \tag{4.59}$$

A princípio não temos como saber que o erro ocorreu, então para que possamos tomar conhecimento da ocorrência de X_2, precisamos realizar a síndrome relacionada aos estabilizadores de $\mathcal{C}_{\textit{bit-flip}}$, esta pode ser escrita como o seguinte operador

$$U_{\text{Síndrome}} = \Lambda_{(3,A2)}(X)\Lambda_{(2,A2)}(X)\Lambda_{(2,A1)}(X)\Lambda_{(1,A1)}(X), \tag{4.60}$$

onde $A0$ e $A1$ representam os qubits auxiliares que vão ser alvos das portas CNOT. Note também que o operador apresentado acima pode ser visualizado na Figura 50. Aplicando o operador relacionado à síndrome sobre o estado corrompido pelo erro vamos transferir a

informação de que o erro X_2 ocorreu para os qubits auxiliares, dessa forma o estado que representa o sistema como um todo se transforma em

$$U_{\text{Síndrome}} X_2 |0_L\rangle \otimes |0_{A1} 0_{A2}\rangle = \frac{1}{\sqrt{2}}(|010\rangle + |101\rangle) \otimes |1_{A1} 1_{A2}\rangle. \tag{4.61}$$

Realizando uma medida nos qubits auxiliares, vamos encontrá-los no estado $|11_A\rangle$ e consultando a Tabela 15 é possível dizer, através de métodos estatísticos mencionados no capítulo 1, que o erro mais provável de ter ocorrido é X_2, além disso podemos selecionar também como operação de recuperação $\mathcal{R}$ o próprio erro X_2, então aplicando essa operação no estado acima temos como resultado final

$$\mathcal{R} U_{\text{Síndrome}} X_2 |0_L\rangle \otimes |0_{A1} 0_{A2}\rangle = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) \otimes |1_{A1} 1_{A2}\rangle. \tag{4.62}$$

Observe que o estado acima é $|0_L\rangle$, mas com os qubits auxiliares em $|11_A\rangle$ carregando a informação do erro, com isso temos o retorno do estado a $\mathcal{C}_{bit-flip}$, o que nos permite concluir que decodificamos o estado com sucesso. É válido ressaltar que o processo descrito aqui é idealizado, uma vez que assume que erros só ocorrem após a codificação e antes da síndrome, também é assumido que os qubits auxiliares não apresentam erros. Contudo, sabemos que no cenário real essas suposições não são verdadeiras, o que torna-o mais desafiador, sendo assim para que os códigos que serão apresentados aqui sejam úteis é preciso desenvolver uma computação quântica tolerante a erros, com isso queremos dizer que cada etapa de um algoritmo ou protocolo quântico apresenta mecanismos de detecção e correção de erros, dessa maneira é possível garantir que tudo irá funcionar bem na presença de ruído. Uma observação importante sobre a Figura 50 é que as portas Toffoli e CNOT nos qubits auxiliares servem para contornar problemas relacionados a implementação do circuito que performa o código, pois sem elas não temos recursos para endereçar corretamente uma correção para o erro X_2.

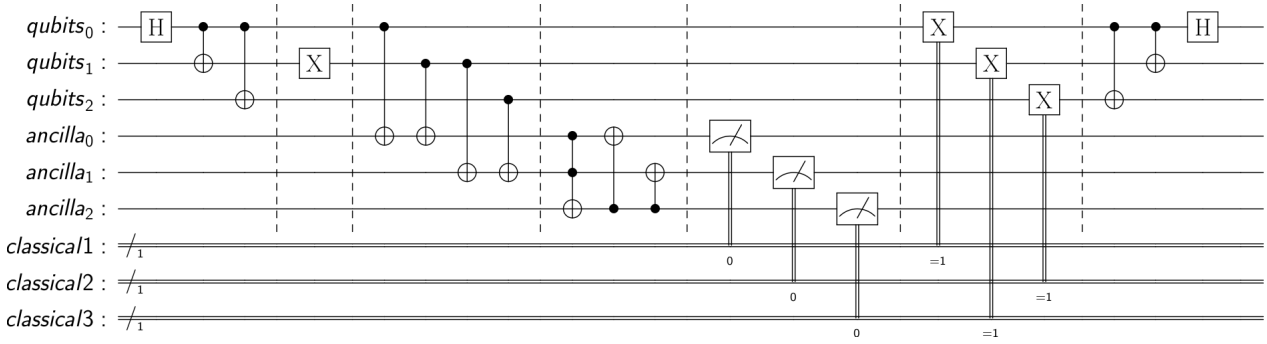


Figura 50 – Circuito quântico que representa a implementação do código [[3,1,3]] para corrigir um único *bit-flip*.

É possível criar um código para lidar com erro do *phase-flip* a partir das ideias apresentadas na implementação do código [[3,1,3]] para *bit-flips*, basicamente o que precisa

Erro	Síndrome
$\mathbb{I}$	00(0)
X_1	10(0)
X_2	11(1)
X_3	01(0)
X_1X_2	01(0)
X_1X_3	11(1)
X_2X_3	10(0)
$X_1X_2X_3$	00(0)

Tabela 15 – Mapeamento entre síndrome e as posições em que os *bit-flips* ocorreram. Observação: A *bitstring* da síndrome contém um bit a mais entre parênteses para representar o qubit auxiliar extra introduzido no circuito para corrigir um problema na decodificação devido a uma limitação das portas quânticas controladas por bits clássicos, pois nos casos em que ambos os auxiliares são iguais a 1, uma correção incorreta era aplicada ao estado o que resultava em falha na decodificação, mas com esse auxiliar extra conseguimos garantir que a correção correta é aplicada.

ser feito é usar a relação (3.20) para transformar os estabilizadores de *bit-flip* em

$$\mathcal{S}_{phase-flip} = \langle X_1X_2, X_2X_3 \rangle.$$

Além disso, podemos usar a porta Hadamard para transformar os estados lógicos $|0_L\rangle$ e $|1_L\rangle$ nos estados lógicos para *phase-flip*

$$|+_L\rangle = H_1 \otimes H_2 \otimes H_3|0_L\rangle = \frac{1}{\sqrt{2}}(|+++\rangle + |---\rangle) \tag{4.63}$$

$$|-_L\rangle = H_1 \otimes H_2 \otimes H_3|1_L\rangle = \frac{1}{\sqrt{2}}(|+++\rangle - |---\rangle\,. \tag{4.64}$$

Os novos estados podem ser divididos nos seguintes subespaços

$$\mathcal{C}_{phase\text{-}flip} = \{|+++\rangle\,, |---\rangle\} \tag{4.65}$$

$$\mathcal{F}_1 = \{|-++\rangle\,, |+--\rangle\} \tag{4.66}$$

$$\mathcal{F}_2 = \{|+-+\rangle\,, |-+-\rangle\} \tag{4.67}$$

$$\mathcal{F}_3 = \{|++-\rangle\,, |--+\rangle\}, \tag{4.68}$$

onde $\mathcal{C}_{phase\text{-}flip}$ representa o código [[3,1,3]] de *phase-flip* e os $\mathcal{F}_i$ representam os espaços dos erros com o índice $i = 1, 2\ e\ 3$ representando a posição onde um *phase-flip* ocorreu. Repetindo os procedimentos descritos na construção do código de *bit-flip* [[3,1,3]], geramos o circuito 51 que implementa o código $\mathcal{C}_{phase\text{-}flip}$, cujo procedimento de decodificação pode ser encontrado na Tabela 16. Observe que no operador de síndrome deste código foram usadas portas CNOT com portas Hadamard no qubit de controle, isto é necessário pois as portas CNOT foram construídas no contexto clássico e portanto só trabalham com

0 ou 1, sendo assim é preciso aplicar as portas Hadamard para garantir que o qubit no controle esteja na base computacional, assim podemos garantir que a síndrome seja efetuada corretamente.

Erro	Síndrome
$\mathbb{I}$	00(0)
Z_1	10(0)
Z_2	11(1)
Z_3	01(0)
Z_1Z_2	01(0)
Z_1Z_3	11(1)
Z_2Z_3	10(0)
$Z_1Z_2Z_3$	00(0)

Tabela 16 – Mapeamento entre a síndrome e as posições em que os *phase-flips* ocorreram.

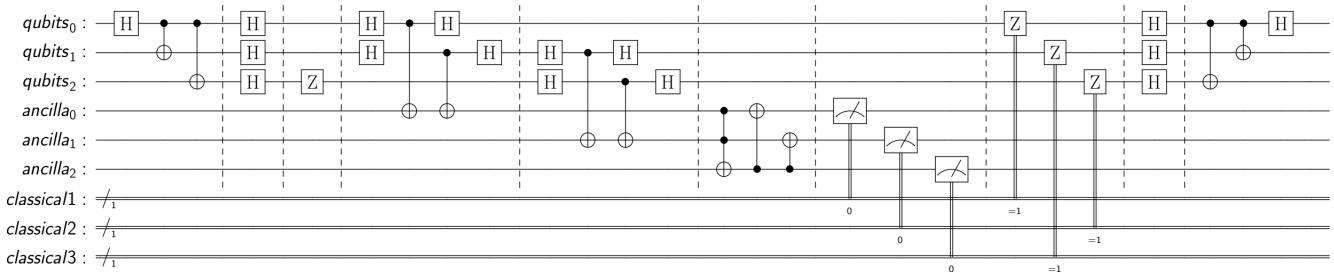


Figura 51 – Circuito quântico que representa a implementação do código [[3,1,3]] para corrigir um único *phase-flip*.

4.3.1 Uso da Fidelidade na Análise de Erros

No capítulo 1, vimos algumas ferramentas estatísticas que nos auxiliam a identificar quando erros ocorreram, infelizmente não podemos aplicá-las diretamente no contexto quântico devido às diferenças discutidas anteriormente, por exemplo, os erros quânticos podem ser contínuos e podem causar níveis diferentes de danos na informação. No capítulo 2, discutimos a fidelidade que é uma forma de medir a "distância" entre estados quânticos, aqui iremos usar essa habilidade para poder analisar os possíveis danos que podem ser causados à informação quântica por erros coerentes e incoerentes. Como exemplo vamos considerar um erro *bit-flip*, sabemos que este erro quando atua sobre um estado $|0\rangle$ leva-o em $|1\rangle$, que é um estado ortogonal ao estado original, o que implica $F(|0\rangle, |1\rangle) = 0$, já em superposições do tipo $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ ocorre a troca de amplitude de probabilidade dos estados, o que é suficiente para modificar a fidelidade do estado, mas note que no caso especial $\alpha = \beta = 1/\sqrt{2}$ este tipo de erro não causa nenhuma alteração no sistema.

O objetivo dos códigos de correção de erros é garantir que a fidelidade do estado quântico que codifica a informação quântica que temos interesse em armazenar ou transmitir seja preservada. Visando compreender como a fidelidade pode ser usada na análise de erros, vamos considerar uma situação em que nosso qubit seja representado por um estado puro $|\psi\rangle$, tenha sofrido a ação de um canal de *bit-flip*, após isso sua matriz de densidade passa a ser escrita como

$$\rho = (1-p)\,|\psi\rangle\langle\psi| + pX\,|\psi\rangle\langle\psi|\,X, \tag{4.69}$$

com isso podemos usar (2.155) para verificar o quanto o canal modificou nosso estado inicial $|\psi\rangle$, sendo assim temos

$$\begin{aligned} F &= \sqrt{\langle\psi|\,\rho\,|\psi\rangle} \\ &= \sqrt{(1-p)\langle\psi|\psi\rangle\langle\psi|\psi\rangle + p\langle\psi|X|\psi\rangle\langle\psi|X|\psi\rangle}, \end{aligned} \tag{4.70}$$

se $|\psi\rangle$ é um estado normalizado, então $\langle\psi|\psi\rangle = 1$ e nos casos em que $|\psi\rangle = |0\rangle$ ou $|\psi\rangle = |1\rangle$, temos que

$$\langle\psi|\sigma_x|\psi\rangle = 0, \tag{4.71}$$

com isso a fidelidade se reduz a

$$F = \sqrt{1-p}. \tag{4.72}$$

Agora vamos considerar que um código de correção de erros contra *bit-flips* foi utilizado em $|\psi\rangle$ e o transformou no estado lógico $|\psi'\rangle = \alpha|0_L\rangle + \beta|1_L\rangle$, dessa forma a matriz de densidade que representa o sistema após a ocorrência do erro e de uma decodificação com correção de erros é igual a [1]

$$\rho' = \left[(1-p)^3 + 3p(1-p)^2\right]|\psi\rangle\langle\psi| + \ldots, \tag{4.73}$$

onde são omitidas as contribuições de termos de dois ou três *bit-flips*, uma vez que estes termos são operadores positivos, então a fidelidade a ser calculada a partir da matriz de densidade acima com essa omissão de termos nos dará um limite inferior para a fidelidade real. Fazendo uso de (2.155) para determinar a fidelidade entre ρ' e $|\psi'\rangle$ obtemos

$$F' = \sqrt{(1-p)^3 + 3p(1-p)^2}, \tag{4.74}$$

comparando os resultados F e F' podemos chegar a conclusão que $F \geq F'$, este resultado nos diz que o estado quântico não é corrompido pelo erro enquanto $p < 1/2$, como pode ser visto na Figura 52.

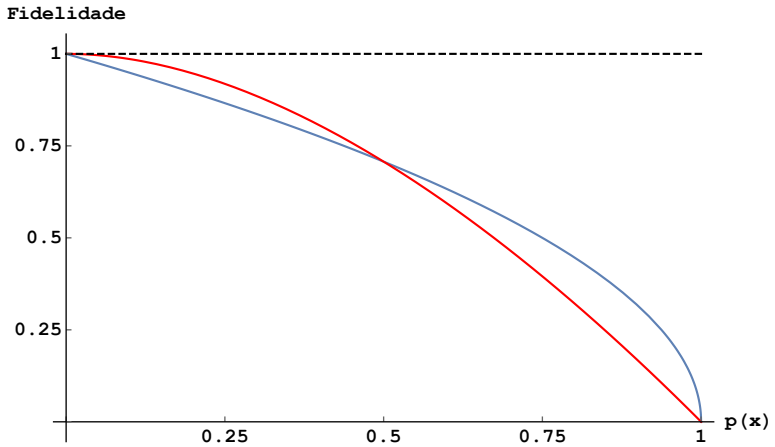


Figura 52 – Comparação dos comportamentos de F e F' em função da probabilidade de ocorrência de *bit-flips*.

4.4 Código de Shor [[9,1,3]]

O primeiro código quântico de correção de erros capaz de lidar com os desafios impostos no contexto quântico foi criado por Peter Shor [10], este código foi o responsável por destravar o desenvolvimento da computação quântica, pois antes de sua criação existia muita incerteza acerca da possibilidade de corrigir erros nesse paradigma de computação. O código de Shor apresenta distância de código $d = 3$ e consegue lidar com erros do tipo *bit-flip*, *phase-flip* e ambos, apesar deste código ser um exemplo de código degenerado, é possível realizar uma recuperação com sucesso para um erro em qualquer qubit.

O código de Shor é construído através de uma técnica conhecida como concatenação de código, esta consiste em incorporar o resultado da codificação de um código dentro da inicialização do outro, no código em questão devemos fazer uma incorporação do código [[3,1,3]] para *bit-flip* e dentro do código [[3,1,3]] para *phase-flip*, como exposto abaixo

$$|+_L\rangle = |+++\rangle \text{ Concatenação } \rightarrow |0_L\rangle_{Shor} = |+\rangle \otimes |+\rangle \otimes |+\rangle, \tag{4.75}$$

observe que no processo de concatenação cada $|+\rangle$ é reformulado da seguinte forma

$$|+\rangle = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle). \tag{4.76}$$

Essa redefinição é igual ao $|0_L\rangle$ do código de *bit-flip*. Repetindo o processo para o estado $|-_L\rangle$, obtemos

$$|-_L\rangle = |---\rangle \text{ Concatenação } \rightarrow |1_L\rangle_{Shor} = |-\rangle \otimes |-\rangle \otimes |-\rangle, \tag{4.77}$$

com o estado $|-\rangle$ sendo reescrito, após a concatenação, como

$$|-\rangle = \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle). \tag{4.78}$$

Os estados $|0_L\rangle_{Shor}$ e $|1_L\rangle_{Shor}$ podem ser reescritos através das transformações (4.76) e (4.78), nesta formam eles definem o espaço vetorial do código de Shor que é escrito como o conjunto

$$\mathcal{C}_{[[9,1,3]]} = \left\{ \begin{array}{l} |0_L\rangle_{Shor} = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \\ |1_L\rangle_{Shor} = \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \end{array} \right\}. \tag{4.79}$$

Note que o processo de concatenação para obter os vetores da base de $\mathcal{C}_{[[9,1,3]]}$ pode ser resumido como o circuito da Figura 53.

Listing 4.1 – Código que cria o circuito de codificação do código de Shor.

```
from qiskit import QuantumRegister, QuantumCircuit

qubits=QuantumRegister(9,name='qubits')
```

```
qc=QuantumCircuit(qubits)

qc.h([qubits[0],qubits[3],qubits[6]])
qc.cx(qubits[0],[qubits[1],qubits[2]])
qc.cx(qubits[3],[qubits[4],qubits[5]])
qc.cx(qubits[6],[qubits[7],qubits[8]])
```

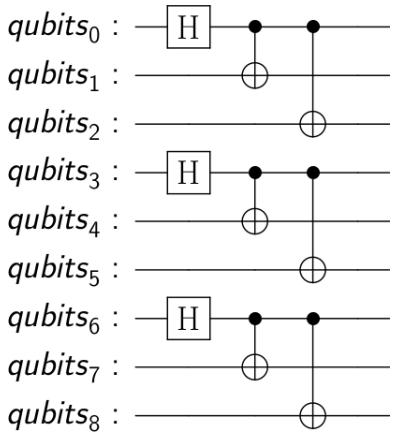


Figura 53 – Circuito de codificação do estado lógico $|0_L\rangle_{Shor}$.

O conjunto mínimo de operadores que estabilizam os estados $|0_L\rangle_{Shor}$ e $|1_L\rangle_{Shor}$ e que definem as síndromes do código de Shor é igual a

$$\mathcal{S}_{[[9,1,3]]} = \langle Z_1Z_2, Z_2Z_3, Z_4Z_5, Z_5Z_6, Z_7Z_8, Z_8Z_9, X_1X_2X_3X_4X_5X_6, X_4X_5X_6X_7X_8X_9\rangle.$$

A partir deste conjunto podemos construir os circuitos presentes na Figura 54, estes são responsáveis por realizar o diagnóstico de erro no código. Note que existe uma assimetria na quantidade de síndromes, temos 6 para erros do tipo X e apenas duas para erros do tipo Z, isto decorre do processo de concatenação, pois para checar a ocorrência de um *bit-flip* precisamos ir na segunda camada dos estados do código de Shor, que é definida por um código de *bit-flip*, já para erros tipo *phase-flip* temos uma resolução reduzida que nos confere apenas dois estabilizadores. Na Tabela 17 podemos encontrar o mapeamento entre as síndromes e cada erro em um único qubit. Observando os dados da tabela é possível ver que cada erro do tipo X é mapeado em uma única síndrome, entretanto para erros do tipo Z há uma certa degenerescência, mas felizmente isto não é um problema dado que a falta de resolução das síndromes de erros dessa natureza não acarreta em diminuição da distância de código [58]. Para compreendermos o porquê disso, vamos considerar que durante o processo de decodificação obtivemos a síndrome '00000011', o que implica que podem ter ocorrido os erros Z_4, Z_5 ou Z_6, neste ponto parece que não é possível reparar o estado, mas mesmo com informação faltando é possível realizar uma recuperação bem sucedida. Para entendermos como isso ocorre, vamos assumir que o erro que gerou a síndrome obtida foi $E = Z_4$, então operação de recuperação trivial seria $\mathcal{R} = Z_4$, contudo

se em vez de Z_4 o erro fosse igual a $E = Z_5$ ainda seria possível restaurar o estado lógico original através da recuperação $\mathcal{R} = Z_4$, pois neste caso o produto $\mathcal{R}E = Z_4Z_5$ pertence ao conjunto de estabilizadores. É válido ressaltar que as síndromes referentes aos erros do tipo Y são combinações das síndromes dos respectivos erros do tipo X e Z, isso se deve ao fato de que é possível escrever Y como um produto de X e Z.

Erro	Síndrome	Erro	Síndrome	Erro	Síndrome
X_1	10000000	Z_1	00000010	Y_1	10000010
X_2	11000000	Z_2	00000010	Y_2	11000010
X_3	01000000	Z_3	00000010	Y_3	01000010
X_4	00100000	Z_4	00000011	Y_4	00100011
X_5	00110000	Z_5	00000011	Y_5	00110011
X_6	00010000	Z_6	00000011	Y_6	00010011
X_7	00001000	Z_7	00000001	Y_7	00001001
X_8	00001100	Z_8	00000001	Y_8	00001101
X_9	00000100	Z_9	00000001	Y_9	00000101

Tabela 17 – Mapeamento entre as síndromes e todos os erros que podem ser corrigidos pelo código de Shor [[9,1,3]].

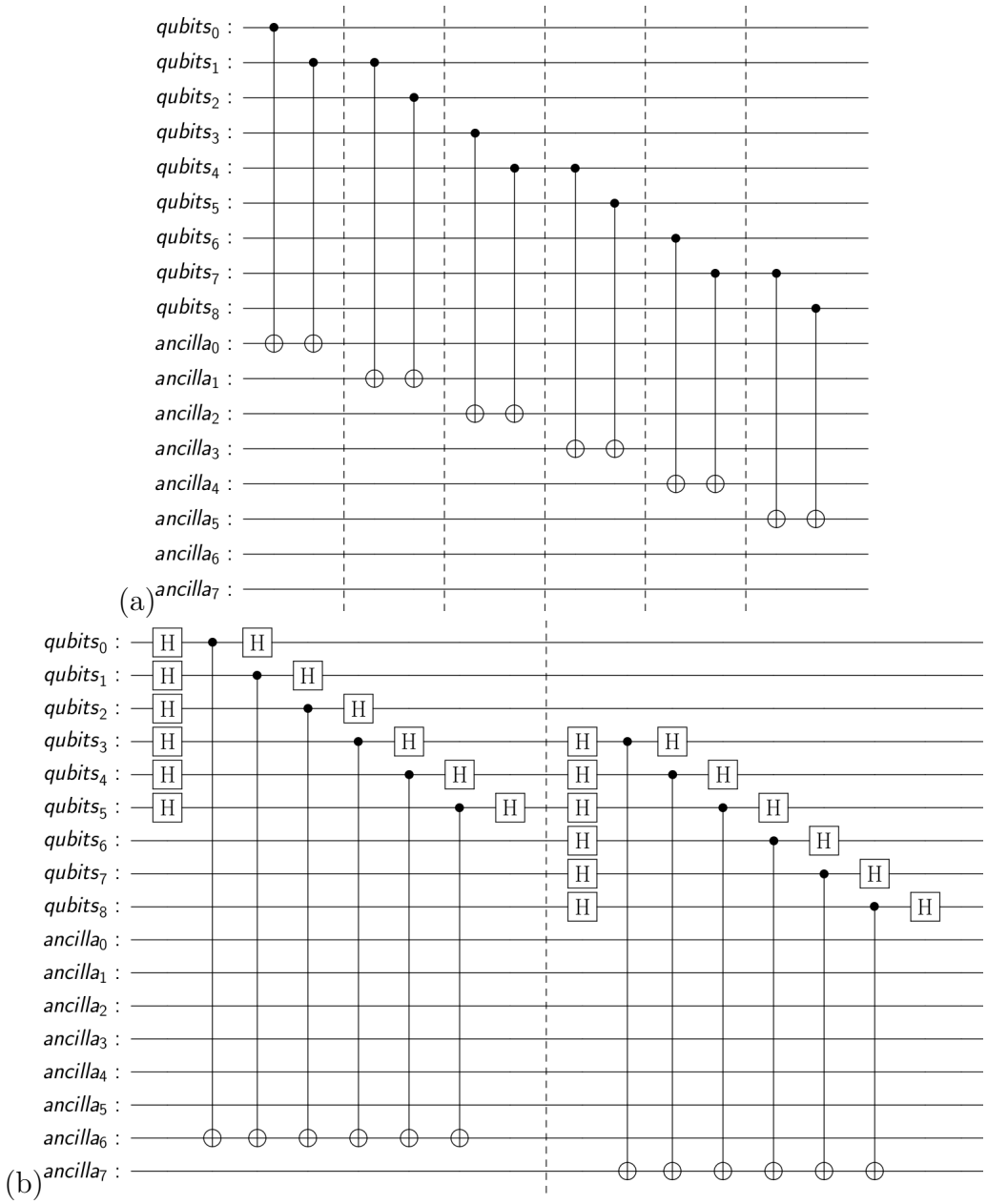


Figura 54 – Circuitos que executam as síndromes definidas pelo conjunto de estabilizadores $\mathcal{S}_{[[9,1,3]]}$.

4.5 Surface Codes

A criação de um código quântico de correção de erros pode ser algo desafiador, pois consiste em encontrar o conjunto de estabilizadores que nos permite identificar os erros sem causar danos a informação codificada nos qubits. Em geral, essa não é uma tarefa fácil e demanda certa quantidade de trabalho para encontrar os operadores que satisfazem as propriedades de estabilizador [58]. Nesta seção iremos discutir os *surface codes*, estes fazem parte de uma classe de códigos que pertence a família dos códigos topológicos [36, 58]. A construção desse tipo de código é feita através de uma representação gráfica que facilita o trabalho de identificação do conjunto de estabilizadores e de operadores lógicos.

Atualmente grande parte dos esforços em computação quântica visam obter a supressão exponencial de erros através de códigos quânticos de correção de erros, um dos principais candidatos a nos fazer alcançar esta meta são os *surface codes*, pois eles influenciam o mapa de acoplamento dos qubits nos *hardwares* atuais, fazendo com que sejam consideradas apenas interações entre os vizinhos mais próximos, por exemplo a IBM tem usado uma arquitetura que torna natural a implementação desse tipo código [60].

Como foi dito no início desta seção, é possível utilizar uma representação gráfica para construir os *surface codes*. O bloco fundamental dessa classe de código é dado pelas representações presentes na Figura 55, onde os qubits marcados com a letra D representam os que vão ser usados para codificar a informação, já os rotulados com A são os qubits auxiliares relacionados às síndromes. Note que esses qubits são ligados por linhas vermelhas e azuis, estas representam portas controladas que usam os qubits auxiliares como controle e os qubits de codificação como alvo, as linhas vermelhas representam portas CNOT e as azuis representam portas CZ, dessa forma podemos ler da Figura 55 que os estabilizadores são $X_{D1}X_{D2}$ e $Z_{D1}Z_{D2}$. O circuito da Figura 56 representa a implementação do bloco fundamental dos *surface codes*. Observe que o bloco definido aqui não representa um código, pois o número de qubits para codificação $n = 2$ é igual ao número de qubits auxiliares $m = 2$, este fato nos leva a ter $k = n - m = 0$ que reflete em não ter qubits codificados, portanto esse bloco não serve como um código em si, mas combinações de vários desses blocos definem códigos úteis.

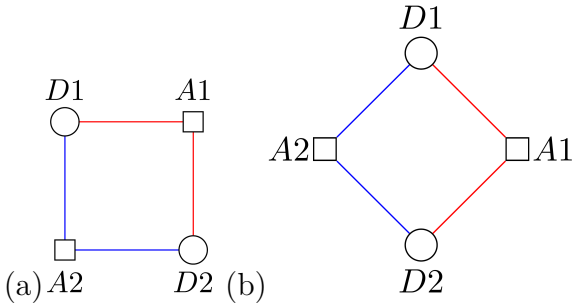


Figura 55 – Representações mais comuns do bloco fundamental dos *surface codes*.

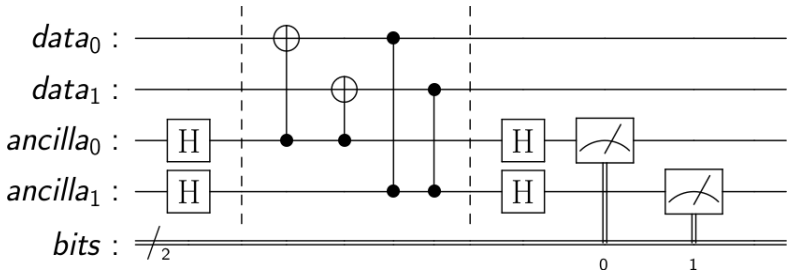


Figura 56 – Circuito que implementa o bloco fundamental dos *surface codes*.

A fórmula geral que determina os *surface codes* que codificam 1 qubit lógico é

dada por [58]

$$[[n = \lambda^2 + (\lambda - 1)^2, k = 1, d = \lambda]], \tag{4.80}$$

mas isto não quer dizer que essa classe de código se limita apenas a $k = 1$, é possível construir códigos com $k > 1$ através de certas combinações de blocos fundamentais ou de cortes na malha. Um exemplo pode ser visto na Figura 59. Adotando $\lambda = 2$ em (4.80) define-se um código com distância $d = 2$ que serve apenas para detectar erros e introduzir as ideias de combinação de blocos fundamentais. Este código [[5,1,2]] é formado pela combinação de 4 blocos fundamentais e sua representação gráfica pode ser vista na Figura 57 (a), desta é possível identificar o seguinte conjunto mínimo de estabilizadores

$$\mathcal{S}_{[[5,1,2]]} = \langle X_{D1}X_{D2}X_{D3}, Z_{D1}Z_{D3}Z_{D4}, Z_{D2}Z_{D3}Z_{D5}, X_{D3}X_{D4}X_{D5}\rangle,$$

para determinar os operadores lógicos precisamos do código dual, este é definido através da utilização das relações (3.20) e (3.21) para transformar o conjunto de estabilizadores acima, o que implica na troca de cores na representação gráfica. Isso pode ser visto na Figura 57 (b), sendo assim os operadores lógicos serão iguais aos qubits nas extremidades, neste caso como $k = 1$ temos apenas um par de operadores lógicos que podem ser um dos conjuntos apresentados abaixo

$$\mathcal{L}_{[[5,1,2]]} = \{\bar{X} = X_{D1}X_{D4}, \bar{Z} = Z_{D1}Z_{D2}\}$$
$$\mathcal{L}_{[[5,1,2]]} = \{\bar{X} = X_{D2}X_{D5}, \bar{Z} = Z_{D4}Z_{D5}\},$$

note que qualquer uma das opções acima satisfaz as condições de operadores lógicos (4.47). Através do conjunto de estabilizadores $\mathcal{S}_{[[5,1,2]]}$ e de (4.45) é possível determinar o estado $|0_L\rangle_{[[5,1,2]]}$

$$|0_L\rangle_{[[5,1,2]]} = \frac{1}{2}(|00000\rangle + |00111\rangle + |11011\rangle + |11100\rangle). \tag{4.81}$$

Existe um caminho alternativo para criar o estado acima, este substitui os estabilizadores do tipo X por portas CNOT de acordo com (3.38), além disso é preciso observar os qubits que se repetem nos estabilizadores X e olhar para os qubits que aparecem junto a ele em estabilizadores Z. Uma vez identificados, eles devem ser separados para serem inicializados com porta Hadamard, por exemplo: considere $\mathcal{S}_{[[5,1,2]]}$, observando os estabilizadores do tipo X vemos que $D3$ se repete nestes, então ao olharmos para os estabilizadores do tipo Z vemos que ele aparece com os pares $(D1, D4)$ e $(D2, D5)$, escolhendo o primeiro par e usando (3.38) temos

$$\Lambda_{(1,3)}(X)\Lambda_{(1,2)}(X)H_1 \tag{4.82}$$
$$\Lambda_{(4,5)}(X)\Lambda_{(4,3)}(X)H_4, \tag{4.83}$$

que são os operadores que aplicados sobre $|00000\rangle$ produzem (4.81). Note que se escolhermos o par $(D2, D5)$ teremos operadores diferentes, mas o resultado final será o mesmo.

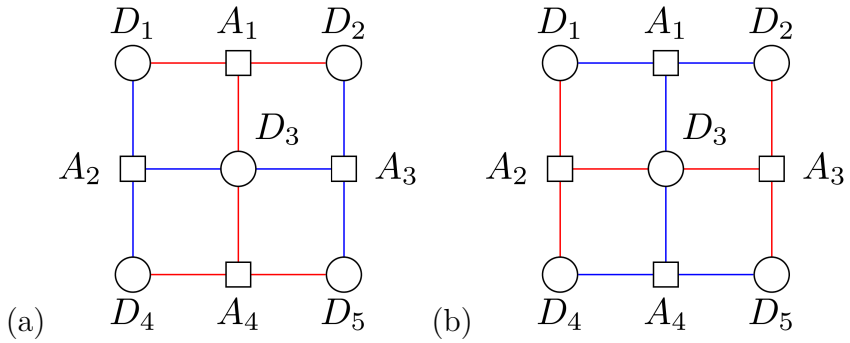


Figura 57 – Representação gráfica do *surface code* [[5,1,2]]. (a) Código original. (b) Código dual.

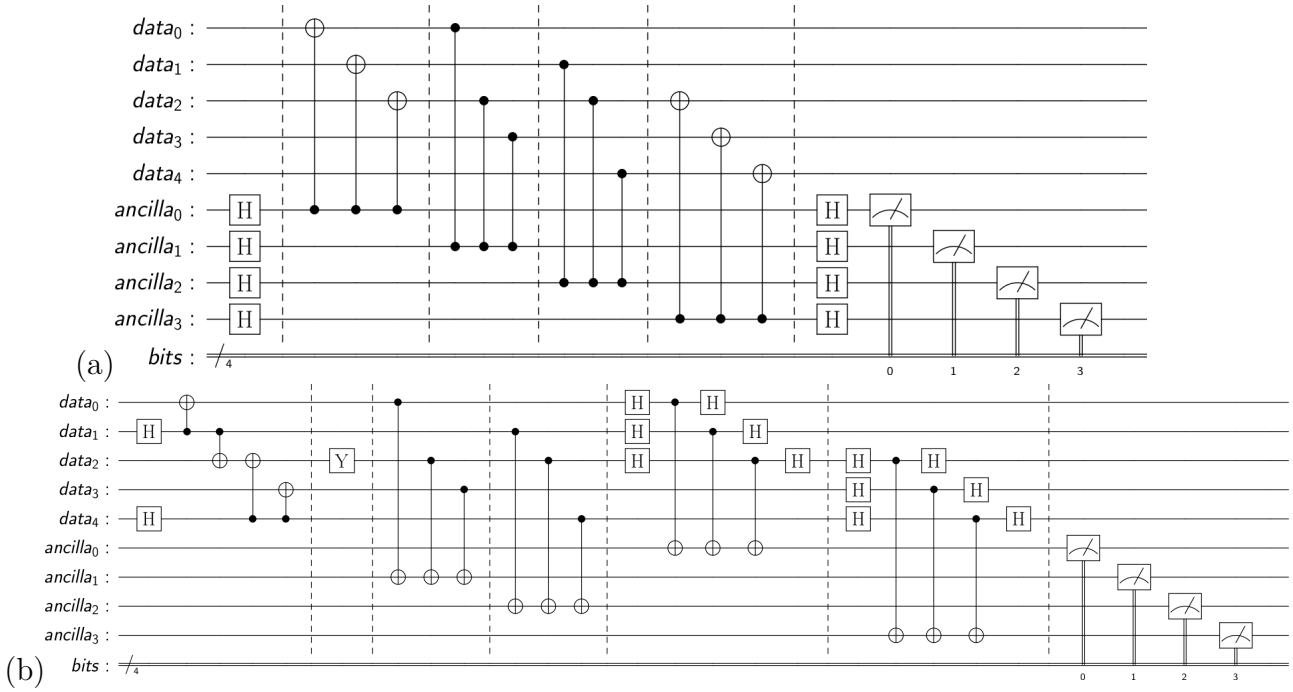


Figura 58 – Circuitos que implementam o *surface code* [[5,1,2]]: (a) Codificação pelo método de estabilizadores. (b) Codificação com portas do grupo de Clifford. O circuito na figura está detectando um erro Y no qubit $D2$.

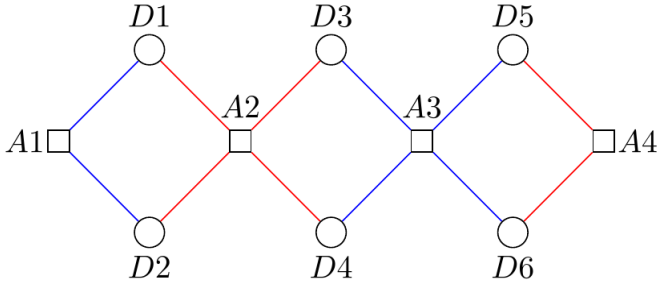


Figura 59 – Representação gráfica do *surface code* [[6,2,2]].

De acordo com a fórmula (4.80) o primeiro *surface code* com distância $d = 3$ é o [[13,1,3]], ou seja, o primeiro código desse tipo que nos permite corrigir um erro. Para construir o [[13,1,3]] são necessário 25 qubits, 13 para a codificação do estado e 12 auxiliares

para serem usados nas síndromes, como pode ser visto na Figura 18. Dela também podemos tirar o conjunto mínimo de estabilizadores

$$\begin{aligned}\mathcal{S}_{[[13,1,3]]} &= \langle X_{D1}X_{D2}X_{D4}, X_{D2}X_{D3}X_{D5}, Z_{D1}Z_{D4}Z_{D6}, Z_{D2}Z_{D4}Z_{D5}Z_{D7}, Z_{D3}Z_{D5}Z_{D8} \\ &, X_{D4}X_{D6}X_{D7}X_{D9}, X_{D5}X_{D7}X_{D8}X_{D10}, Z_{D6}Z_{D9}Z_{D11}, Z_{D7}Z_{D9}Z_{D10}Z_{D12} \\ &, Z_{D8}Z_{D10}Z_{D13}, X_{D9}X_{D11}X_{D12}, X_{D10}X_{D12}X_{D13}\rangle, \end{aligned} \tag{4.84}$$

com auxílio dos estabilizadores apresentados acima e de (4.45) podemos obter o estado $|0_L\rangle_{[[13,1,3]]}$ ou através dos seguintes operadores do grupo de Clifford

$$\Lambda_{(1,4)}(X)\Lambda_{(1,2)}(X)H_1 \tag{4.85}$$
$$\Lambda_{(3,5)}(X)\Lambda_{(3,2)}(X)H_3 \tag{4.86}$$
$$\Lambda_{(6,9)}(X)\Lambda_{(6,7)}(X)\Lambda_{(6,4)}(X)H_6 \tag{4.87}$$
$$\Lambda_{(8,10)}(X)\Lambda_{(8,7)}(X)\Lambda_{(8,5)}(X)H_8 \tag{4.88}$$
$$\Lambda_{(11,12)}(X)\Lambda_{(11,9)}(X)H_{11} \tag{4.89}$$
$$\Lambda_{(13,10)}(X)\Lambda_{(13,12)}(X)H_{13}. \tag{4.90}$$

Através do código dual é possível chegar aos operadores lógicos

$$\mathcal{L}_{[[13,1,3]]} = \{\bar{X} = X_{D1}X_{D6}X_{D11}, \bar{Z} = Z_{D1}Z_{D2}Z_{D3}\}, \tag{4.91}$$

estes fazem a passagem de $|0_L\rangle_{[[13,1,3]]}$ para $|1_L\rangle_{[[13,1,3]]}$ e vice-versa sem a necessidade de decodificar e recodificar. Na Tabela 18, temos o mapeamento entre todos os erros em apenas um qubit e suas respectivas síndromes. Observe que este mapeamento é um a um, o que nos diz que este código é não degenerado, o que implica que todos os erros tabelados podem ser corrigidos.

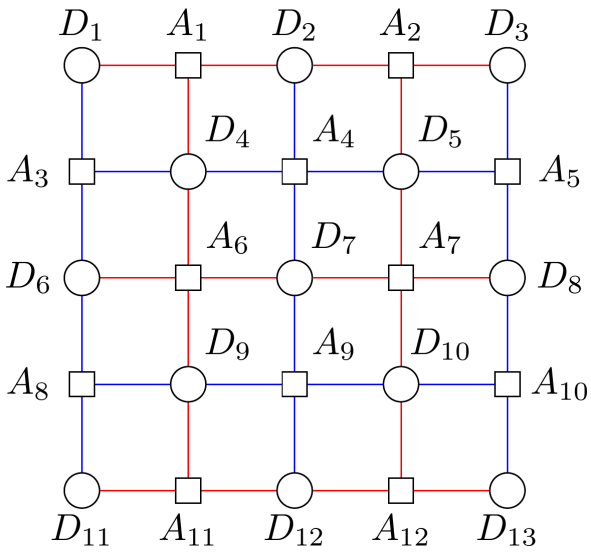


Figura 60 – Representação gráfica do *surface code* [[13,1,3]].

Erro	Síndrome	Erro	Síndrome	Erro	Síndrome
X_1	100000000000	Z_1	001000000000	Y_1	101000000000
X_2	110000000000	Z_2	000100000000	Y_2	110010000000
X_3	010000000000	Z_3	000010000000	Y_3	010010000000
X_4	100001000000	Z_4	001100000000	Y_4	101101000000
X_5	010000100000	Z_5	000110000000	Y_5	010110100000
X_6	000001000000	Z_6	001000010000	Y_6	001001010000
X_7	000001100000	Z_7	000100001000	Y_7	000101101000
X_8	000000100000	Z_8	000010000100	Y_8	000010100100
X_9	000001000010	Z_9	000000011000	Y_9	000001011010
X_{10}	000000100001	Z_{10}	000000001100	Y_{10}	000000111001
X_{11}	000000000010	Z_{11}	000000010000	Y_{11}	000000010010
X_{12}	000000000011	Z_{12}	000000001000	Y_{12}	000000001011
X_{13}	000000000001	Z_{13}	000000000100	Y_{13}	000000000101

Tabela 18 – Mapeamento entre as síndromes e todos os erros que podem ser corrigidos pelo *surface code* [[13,1,3]].

Um último comentário relevante sobre os *surface codes* que deve ser feito, é sobre a questão da eficiência de decodificação, pois um código do tipo $[[n, k, d]]$ nos retorna síndromes com m bits, sendo $m = n - k$, o que nos dá um número de síndromes igual a 2^m. Dependendo do número m pode ser impraticável trabalhar com tabelas de síndromes como temos feito até agora, pois teríamos que consultar um número muito grande de itens, por exemplo: para o *surface code* [[41,1,5]] temos $m = 40$, o que resultaria em uma tabela com tamanho $2^{40} \approx 10^{12}$ [58]. Nos computadores quânticos de larga escala, as tabelas de síndrome serão substituídas por técnicas de inferência que conseguem determinar qual é o erro mais provável de ter ocorrido dada uma síndrome S. Este tipo de método permite executar operações de recuperação em tempo real. Infelizmente ainda não temos um decodificador universal que possa ser aplicado de forma eficiente em todos os códigos quânticos, mas é sabido que para *surface codes* existe uma técnica conhecida como *minimum weight perfect matching* (MWPM) que pode ser usada para decodificação [58].

4.6 Teste de Implementação do Surface Code [[4,1,2]]

A correção quântica de erros é um ingrediente essencial para atingirmos objetivo de construir computadores quânticos tolerantes a erros, estes dispositivos nos permitirão desfrutar completamente do potencial desse novo paradigma de computação. Sendo assim, precisamos verificar que os códigos projetados conseguem ser implementados experimentalmente e que atingem níveis satisfatórios de detecção e correção dos erros. Aqui vamos explorar um teste do *surface code* [[4,1,2]] nos dispositivos Santiago, Belém e Lagos da IBM, o estudo a ser discutido aqui foi inspirado pela referência [17].

Da Figura 61 é possível extrair o seguinte conjunto mínimo de estabilizadores do código [[4,1,2]]

$$\mathcal{S}_{[[4,1,2]]} = \langle X_{D1}X_{D2}X_{D3}X_{D4}, Z_{D1}Z_{D3}, Z_{D2}Z_{D4}\rangle, \tag{4.92}$$

e através de $\mathcal{S}_{[[4,1,2]]}$ podemos obter os conjuntos dos pares de operadores lógicos relacionados ao código

$$\mathcal{L}^{(1)}_{[[4,1,2]]} = \{\bar{Z} = Z_{D1}Z_{D2}, \bar{X} = X_{D1}X_{D3}\} \tag{4.93}$$
$$\mathcal{L}^{(2)}_{[[4,1,2]]} = \{\bar{Z} = Z_{D3}Z_{D4}, \bar{X} = X_{D2}X_{D4}\}. \tag{4.94}$$

Dos conjuntos de estabilizadores e de operadores lógicos, construímos os estados lógicos

$$|0_L\rangle_{[[4,1,2]]} = \frac{1}{\sqrt{2}}(|0000\rangle + |1111\rangle) \tag{4.95}$$
$$|1_L\rangle_{[[4,1,2]]} = \frac{1}{\sqrt{2}}(|0101\rangle + |1010\rangle), \tag{4.96}$$

estes serão um de nossos objetos de estudo, pois uma parte do teste que vamos discutir aqui é verificar o quão bem os dispositivos atuais conseguem preparar esses estados. As métricas que vamos utilizar para fazer essa análise serão os valores médios dos estabilizadores e dos operadores lógicos, além dessas quantidades também usaremos a fidelidade entre as matrizes de densidade dos estados de referência e as matrizes densidade dos estados preparados nos computadores quânticos obtidas através de tomografias de estado quântico.

O mapa de acoplamento da arquitetura da QPU utilizada em [17] é equivalente as ligações exibidas na Figura 61, o que permitiu otimizar a quantidade de portas de controle entre os qubits, pois os qubits ligados por elas são vizinhos próximos. Este fato elimina a necessidade de introdução de portas de controle adicionais devido a portas SWAP. No estudo realizado no artigo citado, eles utilizaram pulsos de microondas que performaram as portas R_Y e CZ para executar os circuitos presentes na Figura 62, em nosso caso trabalhamos com as operações $\mathbb{I}$, X, $\sqrt{X}$, U_{CNOT} e R_Z, pois estas são o conjunto de portas disponíveis nos computadores quânticos, além disso é preciso ressaltar que infelizmente os dispositivos usados em nossos testes não possuem um mapa de acoplamento como

o da Figura 61 (seus mapas de acoplamento podem ser vistos na Figura 48), isso nos levou a introduzir portas SWAP para lidar com essa falta de conectividade, o que gerou um impacto negativo nos resultados obtidos. Além disso, devemos ressaltar que os testes realizados nas QPUs Santiago e Belém se limitaram a preparar (4.95) e (4.96) através de operações do grupo de Clifford, como mostrado na Figura 63, os estados obtidos foram usados para calcular os valores médios dos estabilizadores e dos operadores lógicos, optamos por essa abordagem para essas máquinas porque elas apresentam 5 qubits, mas nos testes realizados na QPU Lagos fizemos o teste completo, ou seja, valores médios e tomografia de estado com as preparações de estado descritas pelos circuitos das Figuras 62 e 63.

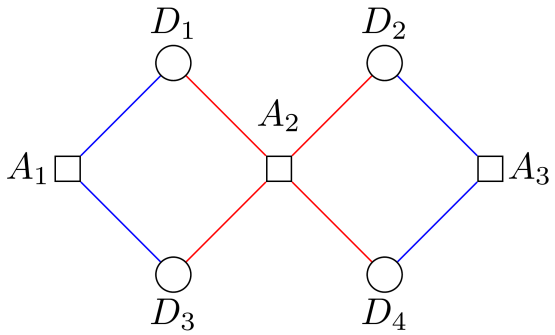


Figura 61 – Representação gráfica do mapa de acoplamento dos qubits usados nos experimentos conduzidos na referência [17] e do *surface code* [[4,1,2]] implementado no dispositivo.

No cenário ideal os valores médios dos estabilizadores presentes em $\mathcal{S}_{[[4,1,2]]}$ devem ser todos iguais a 1, já os valores médios dos operadores lógicos calculados usando (4.95) e (4.96) devem ser iguais a 1 e -1, contudo devido a interferências externas e erros de execução na computação os estados quânticos preparados pelos circuitos mencionados acabam sendo estados mistos e devido a este fato os valores médios calculados diferem dos valores ideais. Nas Tabelas 19, 20, 21 e 22 encontram-se os valores médios dos estabilizadores e do operador lógico Z_1Z_2 calculados a partir dos estados preparados nos computadores quânticos Santiago, Belém e Lagos. Analisando os dados mostrados nas tabelas citadas, podemos verificar que os valores médios não coincidem com o esperado, em alguns casos temos resultados bem próximos do ideal e outros bem distante, isso se deve a profundidade do circuito executado e a quantidade de portas SWAP que foram adicionadas ao circuito para contornar o problema de conectividade. Nas Figuras 65, 66, 67 e 68 podemos observar os estados medidos nos experimentos realizados e compará-los com o previsto pela simulação, note que em alguns casos o comportamento observado é próximo ao esperado.

Por fim realizamos algumas tomografias de estado quântico nos circuitos presentes nas Figuras 62 e 63 usando a técnica apresentada em [61] e os conceitos discutidos no capítulo 2. Os resultados obtidos para a QPU Lagos encontram-se na Tabela 23. Analisando os dados exibidos, podemos ver que usando todos os 7 qubits disponíveis temos

uma fidelidade muito baixa e que encontra-se bem abaixo do resultado $F = 0.703$ obtido em [17]. Atribuímos essa discrepância a diferença de conectividade entre a QPU usada no artigo e a que usamos em nossos testes, pois como já foi dito anteriormente, isto faz com que tenhamos um número maior de portas CNOT devido as portas SWAP introduzidas para transmitir as informações entre os qubits que não estão ligados diretamente. Quando desconsideramos os qubits auxiliares, ou seja, preparamos o estado de acordo com os circuitos (c) e (d) da Figura 63 obtemos uma fidelidade mais próxima de 1, $\underline{F = 0.887}$ para $|0_L\rangle$ e $\underline{F = 0.890}$ para $|1_L\rangle$, o que demonstra que neste caso a QPU Lagos consegue obter um nível maior de precisão no processo de preparação dos estados de interesse, devido a menor profundidade do circuito e a menor quantidade de portas CNOT, entretanto seria interessante testar os circuitos da Figura 64 em uma QPU com mapa de acoplamento igual ao apresentado em 61, para podermos comparar melhor qual abordagem produz melhores resultados na codificação e na detecção de erros. As Figuras 69, 70, 71, 72, 73, 74, 75 e 76 são representações gráficas das matrizes de densidade obtidas através das tomografias de estado quântico realizadas no computador quântico Lagos.

Operador	Valor Médio ($\|0_L\rangle_{[[4,1,2]]}$)	Valor Médio ($\|1_L\rangle_{[[4,1,2]]}$)
$Z_{D1}Z_{D2}$	0.886	-0.910
$Z_{D1}Z_{D3}$	0.867	0.880
$Z_{D2}Z_{D4}$	0.905	0.928
$X_{D1}X_{D2}X_{D3}X_{D4}$	0.743	0.737

Tabela 19 – Valores médios de alguns estabilizadores e operadores lógicos do código [[4,1,2]] calculados com os estados produzidos pela execução dos circuitos (c) e (d) da Figura 63 no computador quântico Santiago.

Operador	Valor Médio ($\|0_L\rangle_{[[4,1,2]]}$)	Valor Médio ($\|1_L\rangle_{[[4,1,2]]}$)
$Z_{D1}Z_{D2}$	0.791	-0.793
$Z_{D1}Z_{D3}$	0.758	0.791
$Z_{D2}Z_{D4}$	0.851	0.859
$X_{D1}X_{D2}X_{D3}X_{D4}$	0.674	0.663

Tabela 20 – Valores médios de alguns estabilizadores e operadores lógicos do código [[4,1,2]] calculados com os estados produzidos pela execução dos circuitos (c) e (d) da Figura 63 no computador quântico Belém.

Operador	Valor Médio ($\|0_L\rangle_{[[4,1,2]]}$)	Valor Médio ($\|1_L\rangle_{[[4,1,2]]}$)
$Z_{D1}Z_{D2}$	0.353	-0.169
$Z_{D1}Z_{D3}$	0.317	0.141
$Z_{D2}Z_{D4}$	0.298	0.263

Tabela 21 – Valores médios de alguns estabilizadores e operadores lógicos do código [[4,1,2]] calculados com os estados produzidos pela execução dos circuitos (c) e (d) da Figura 62 no computador quântico Lagos.

Operador	Valor Médio ($\|0_L\rangle_{[[4,1,2]]}$)	Valor Médio ($\|1_L\rangle_{[[4,1,2]]}$)
$Z_{D1}Z_{D2}$	0.946	-0.950
$Z_{D1}Z_{D3}$	0.940	0.940
$Z_{D2}Z_{D4}$	0.953	0.956
$X_{D1}X_{D2}X_{D3}X_{D4}$	0.845	0.857

Tabela 22 – Valores médios de alguns estabilizadores e operadores lógicos do código [[4,1,2]] calculados com os estados produzidos pela execução dos circuitos (c) e (d) da Figura 63 no computador quântico Lagos.

QPU	Fidelidade ($\|0_L\rangle_{[[4,1,2]]}$)	Fidelidade ($\|1_L\rangle_{[[4,1,2]]}$)
Lagos (s/ qubits auxiliares)	0.887	0.890
Lagos (c/ qubits auxiliares - ETH Zurich)	0.197	0.183
Lagos (c/ qubits auxiliares - Clifford)	0.174	0.154

Tabela 23 – Fidelidade das matrizes de densidade dos estados quânticos preparados no computador quântico Lagos da IBM através dos circuitos (c) e (d) das Figuras 62 e 63 em relação aos estados de referência (4.95) e (4.96).

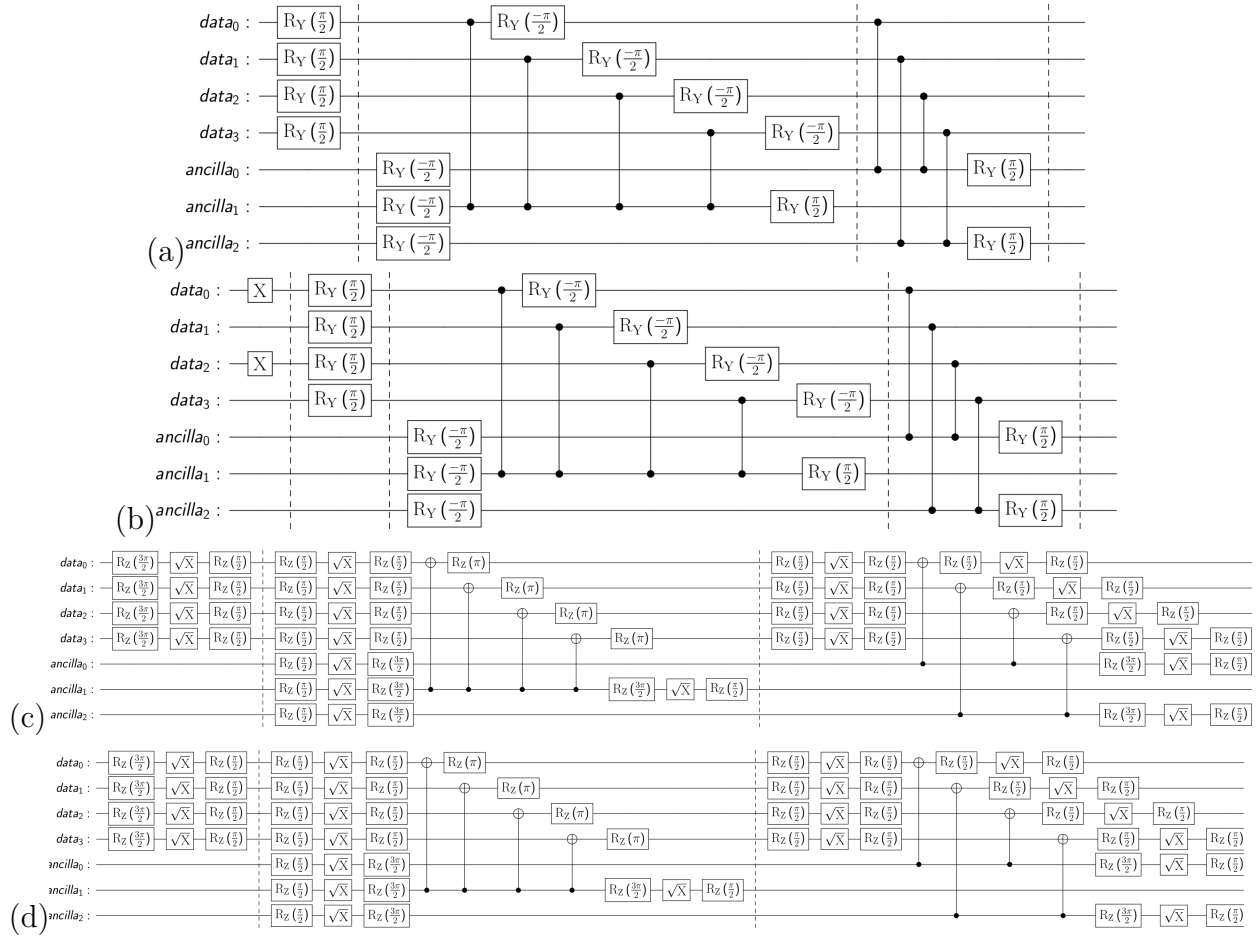


Figura 62 – Os circuitos (a) e (b) foram utilizados para realizar a codificação e síndrome do código [[4,1,2]] em [17]. Os circuitos (c) e (d) são as versões transpiladas que foram executados no computador Lagos da IBM para obter os dados discutidos no texto.

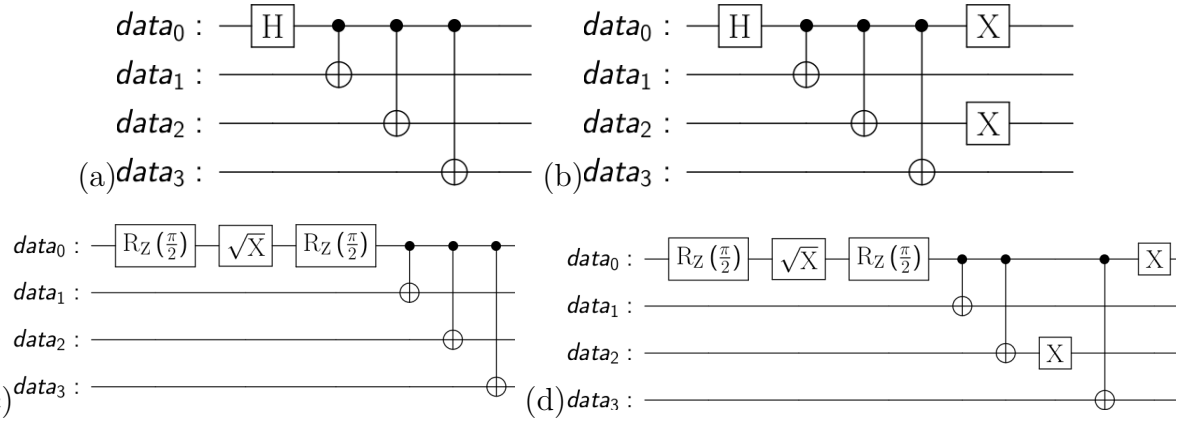


Figura 63 – Os circuitos (a) e (b) são uma versão alternativa dos circuitos apresentados em 62, já os circuitos (c) e (d) são as versões transpiladas que foram usadas nos experimentos realizados nos computadores Belém, Santiago e Lagos da IBM, cujos resultados são apresentados no texto.

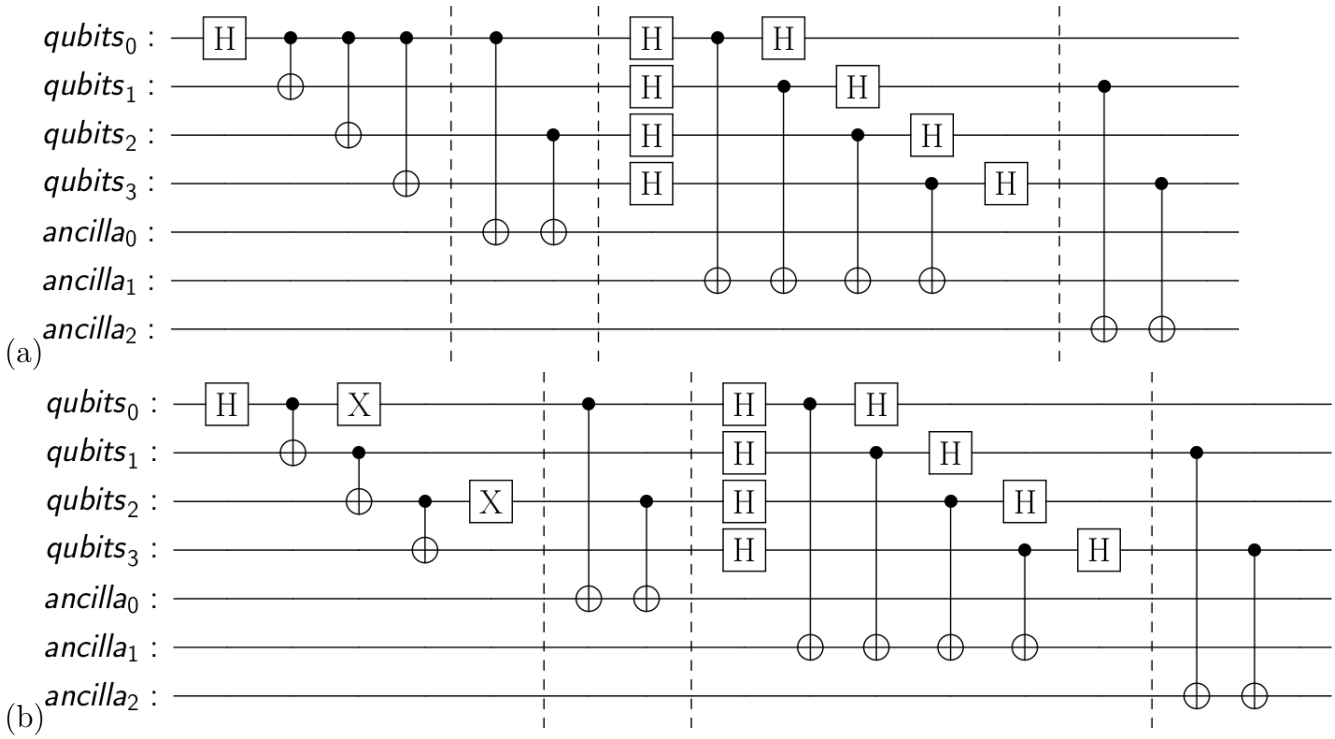


Figura 64 – Os circuitos (a) e (b) são uma versão alternativa dos circuitos apresentados em 62.

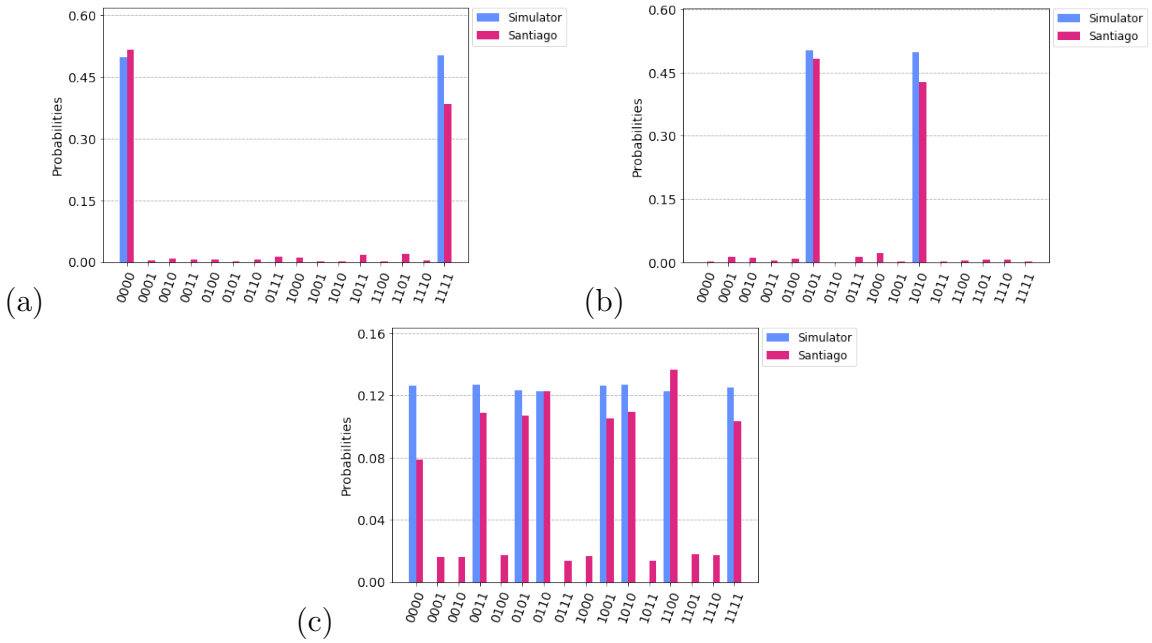


Figura 65 – Histogramas dos resultados de: (a) Preparação do estado (4.95); (b) Preparação do estado (4.96) e (c) Medição do estado (4.95) na base de estados de σ_x no computador quântico Santiago da IBM usando os circuitos (c) e (d) da Figura 63.

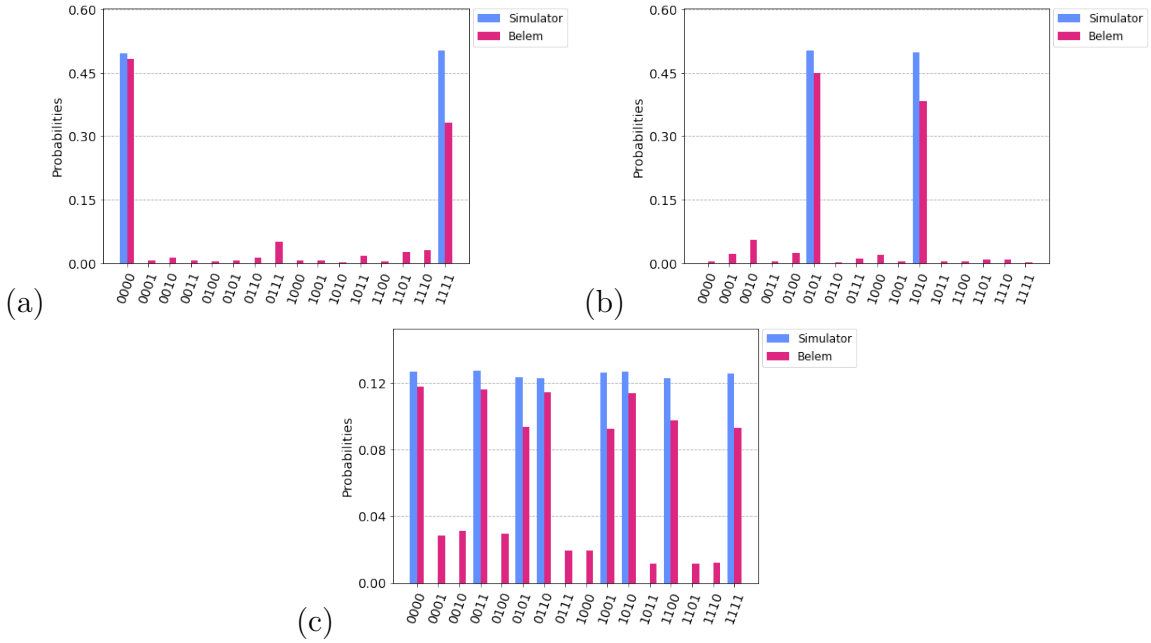


Figura 66 – Histogramas dos resultados de: (a) Preparação do estado (4.95); (b) Preparação do estado (4.96) e (c) Medição do estado (4.95) na base de estados de σ_x no computador quântico Belém da IBM usando os circuitos (c) e (d) da Figura 63.

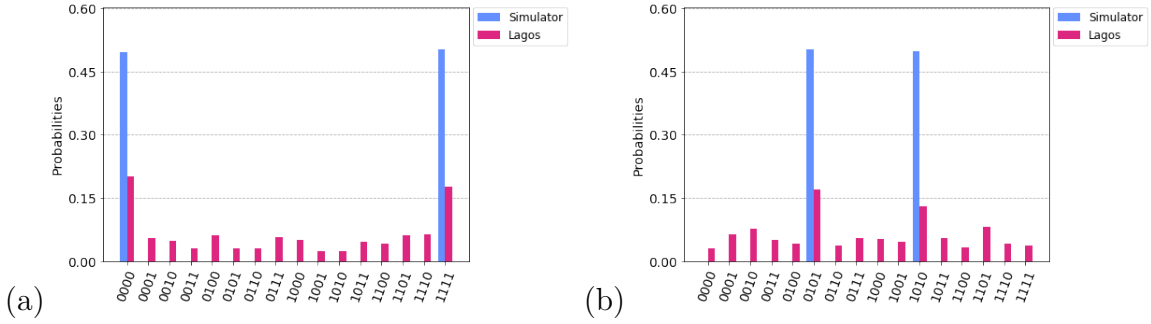


Figura 67 – Histogramas dos resultados de: (a) Preparação do estado (4.95); (b) Preparação do estado (4.96) e (c) Medição do estado (4.95) na base de estados de σ_x no computador quântico Lagos da IBM usando os circuitos (c) e (d) da Figura 62.

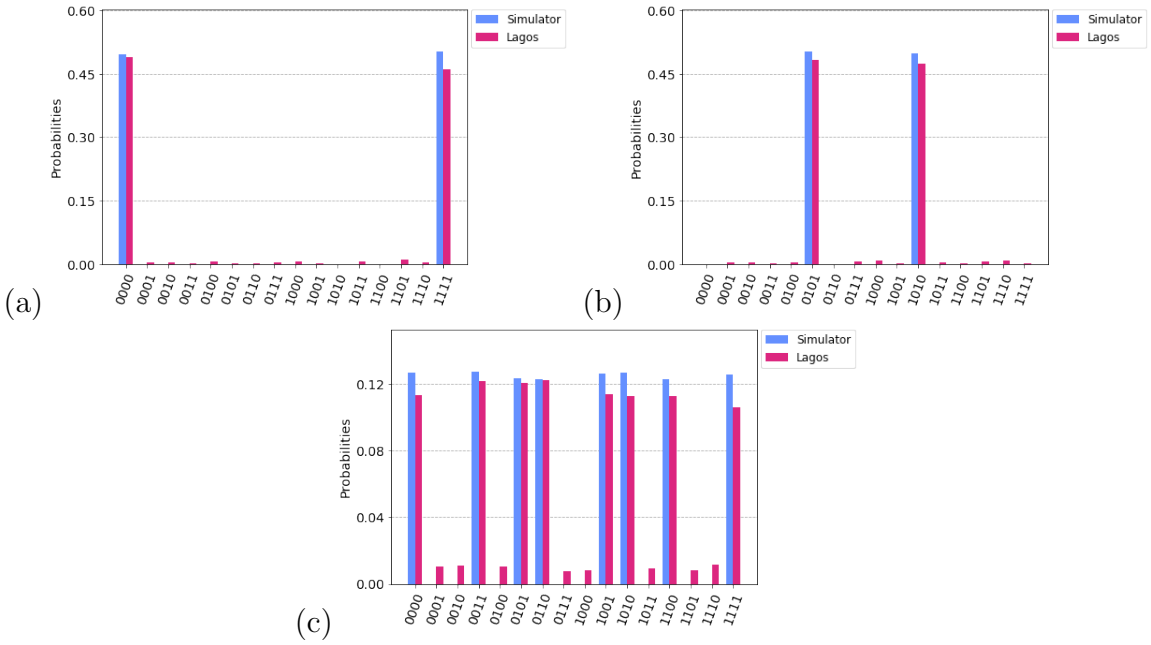


Figura 68 – Histogramas dos resultados de: (a) Preparação do estado (4.95); (b) Preparação do estado (4.96) e (c) Medição do estado (4.95) na base de estados de σ_x no computador quântico Lagos da IBM usando os circuitos (c) e (d) da Figura 63.

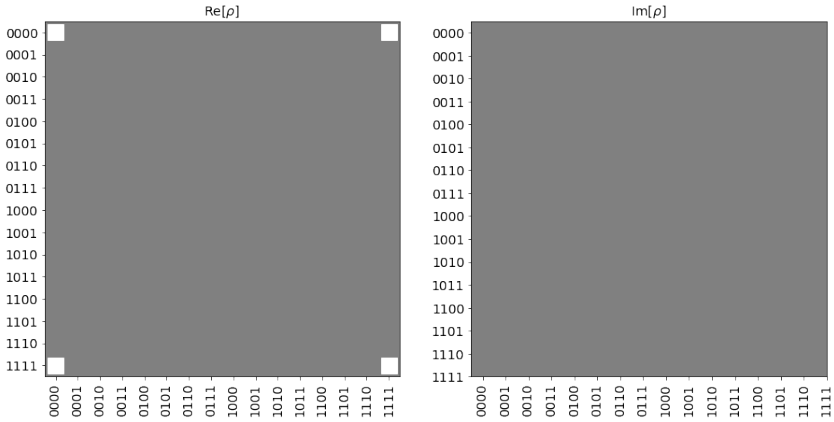


Figura 69 – Representação gráfica da matriz de densidade do estado de referência (4.95).

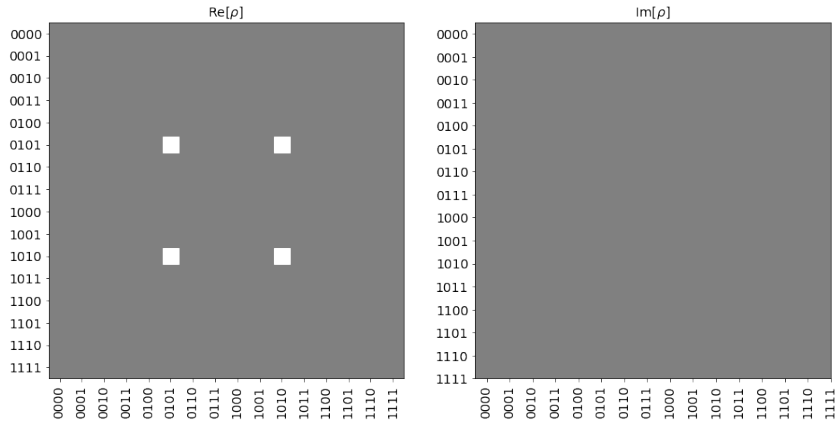


Figura 70 – Representação gráfica da matriz de densidade do estado de referência (4.96).

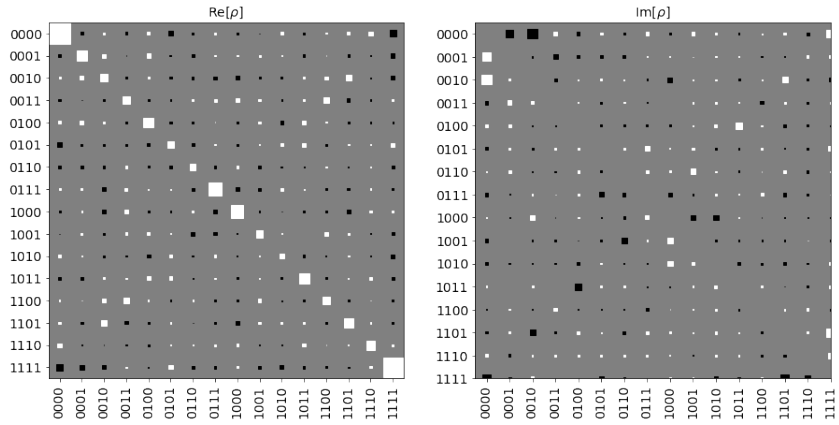


Figura 71 – Representação gráfica da matriz de densidade obtida através da tomografia de estado quântico realizada no circuito (c) da Figura 62.

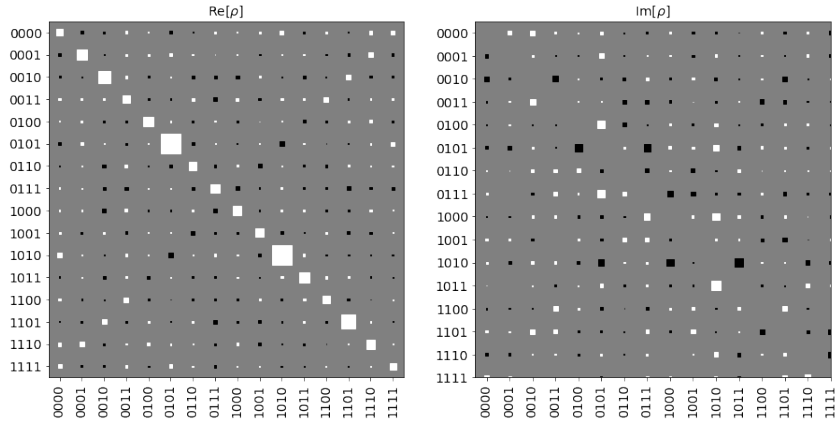


Figura 72 – Representação gráfica da matriz de densidade obtida através da tomografia de estado quântico realizada no circuito (d) da Figura 62.

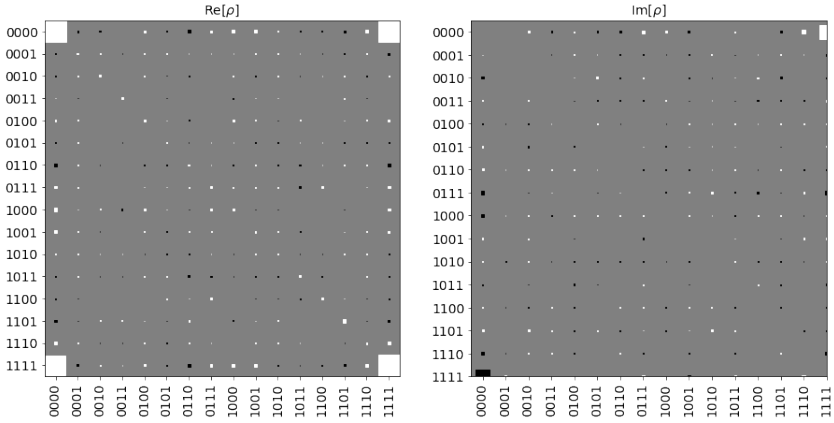


Figura 73 – Representação gráfica da matriz de densidade obtida através da tomografia de estado quântico realizada no circuito (c) da Figura 63.

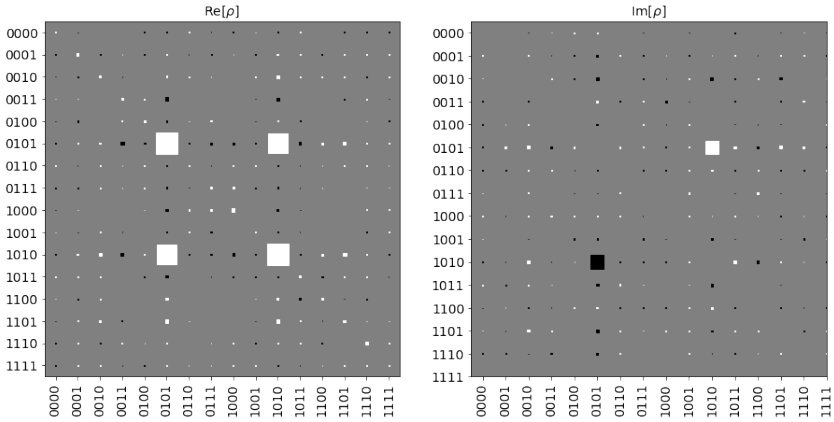


Figura 74 – Representação gráfica da matriz de densidade obtida através da tomografia de estado quântico realizada no circuito (d) da Figura 63.

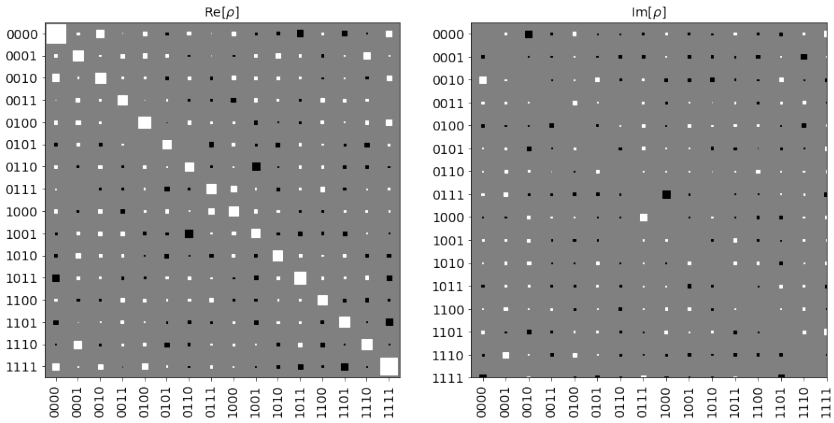


Figura 75 – Representação gráfica da matriz de densidade obtida através da tomografia de estado quântico realizada no circuito (a) da Figura 64.

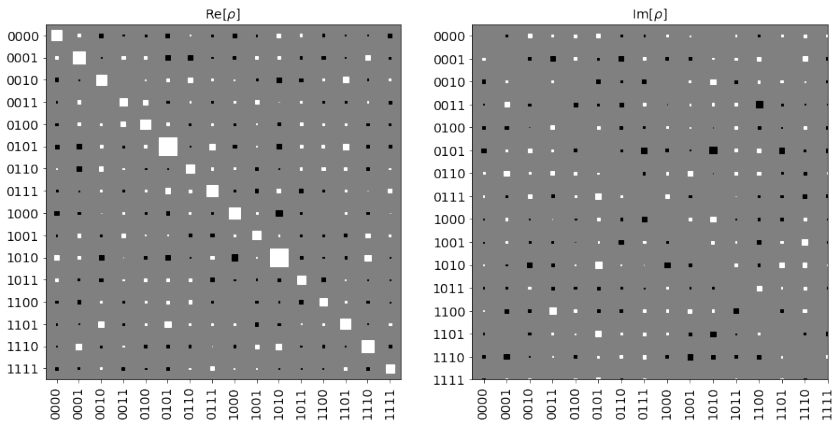


Figura 76 – Representação gráfica da matriz de densidade obtida através da tomografia de estado quântico realizada no circuito (b) da Figura 64.

5 Conclusão

Ao longo do estudo apresentado neste trabalho, vimos que as leis da mecânica quântica impõe algumas dificuldades para a criação de códigos quânticos de correção de erros, pois o teorema da não clonagem nos impede de criar uma copiadora universal de estados quânticos, o que cria uma dificuldade para criar redundância da informação, pois como foi visto no caso clássico, esta é criada a partir de cópias da mensagem, no contexto quântico vimos que o emaranhamento nos permite ter a redundância necessária para realizar a proteção da informação contra erros. Além disso, sabemos que o processo de medição na mecânica quântica causa um colapso na função de onda do sistema físico, isso implica que não podemos observar diretamente os qubits para verificar se algum erro ocorreu, pois causaríamos danos a informação quântica codificada no estado, este comportamento é muito distinto do que é observado classicamente, uma vez que podemos observar diretamente os bits, essa dificuldade é superada através do uso de qubits auxiliares e estabilizadores, pois este conjunto permite obtermos informações sobre o estado quântico através de medidas indiretas. Dessa forma é possível concluir que o emaranhamento, os estabilizadores e os qubits auxiliares são ferramentas essenciais que viabilizam os códigos quânticos, estes se dividem em várias classes e aqui demos mais foco aos *surface codes* devido a sua capacidade de ganhar escala experimentalmente, mostramos que nos *hardware* da era NISQ é possível visualizar uma trajetória promissora de implementação dessa classe de código, pois dos resultados exibidos no capítulo 4, foi possível ver que alguns dispositivos apresentam resultados que se aproximam do ideal, além disso também foi observado que a conectividade entre os qubits tem alto impacto sobre a qualidade dos resultados obtidos, dado que testamos o código [[4,1,2]] em uma QPU com mapa de acoplamento distinto do proposto em [17] e chegamos a resultados bem abaixo dos apresentados em [17] no cenário em que os qubits auxiliares faziam parte do circuito. Apesar disso, verificamos que algumas das QPUs disponíveis atualmente já apresentam um bom nível de taxa de erros, pois no teste realizado com a porta X variando a profundidade do circuito, foi possível observar que alguns qubits apresentavam comportamento muito próximo do ideal.

Para futuros trabalhos poderíamos explorar problemas envolvendo o aumento de qubits codificados nos códigos quânticos com distância $d \geq 3$, além disso poderiam ser feitos outros *benchmarks* para avaliar a performance dos códigos nos *hardwares* atuais ou buscar criar novos códigos que levam a diferentes arquiteturas como [64]. Este trabalho também abre a possibilidade de explorar o ZX-Calculus como ferramenta para entender a propagação de erros ao longo do circuito e na elaboração de códigos [65], visando atingir uma forma tolerante a erros, outro problema importante que ainda está em aberto é sobre o processo de decodificação como indicado em [58]. Um estudo sobre implementações de QRAM [62]

e como implementar um código de correção de erros dentro desta para que ela seja robusta a erros, como apontado em [63], também poderia ser uma futura direção de trabalho. Também é possível uma abordagem visando aplicar os conceitos de códigos quânticos que discutimos aqui em problemas relacionados a correspondência AdS/CFT como sugerido nas referências [66, 67, 68, 69]. Outra futura direção de trabalho seria o estudo da dinâmica de sistemas quânticos abertos através de operadores de Kraus e equações mestras, buscando um entendimento melhor sobre a natureza dos erros e da descoerência nos computadores quânticos.

Referências

[1] NIELSEN, Michael A.; CHUANG, Isaac. Quantum computation and quantum information. 2002. 21, 37, 45, 46, 47, 50, 56, 61, 63, 70, 71, 73, 74, 75, 76, 77, 89, 111, 117, 141, 142, 150

[2] SHANNON, Claude Elwood. A mathematical theory of communication. The Bell system technical journal, v. 27, n. 3, p. 379-423, 1948. 21, 26, 30, 32

[3] VON NEUMANN, John. Mathematical foundations of quantum mechanics. Princeton university press, 2018. 21

[4] BENIOFF, Paul. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. Journal of statistical physics, v. 22, n. 5, p. 563-591, 1980. 22

[5] FEYNMAN, Richard P. Simulating physics with computers. In: Feynman and computation. CRC Press, 2018. p. 133-153. 22

[6] DEUTSCH, David; JOZSA, Richard. 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, 1992. 22

[7] BERNSTEIN, Ethan; VAZIRANI, Umesh. Quantum complexity theory. SIAM Journal on computing, v. 26, n. 5, p. 1411-1473, 1997. 22

[8] GROVER, Lov K. A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. 1996. p. 212-219. 22, 64, 91

[9] SHOR, Peter W. Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th annual symposium on foundations of computer science. Ieee, 1994. p. 124-134. 22

[10] SHOR, Peter W. Scheme for reducing decoherence in quantum computer memory. Physical review A, v. 52, n. 4, p. R2493, 1995. 23, 152

[11] VOOL, Uri; DEVORET, Michel. Introduction to quantum electromagnetic circuits. International Journal of Circuit Theory and Applications, v. 45, n. 7, p. 897-934, 2017. 23, 104, 105

[12] ARRAZOLA, J. M. et al. Quantum circuits with many photons on a programmable nanophotonic chip. Nature, v. 591, n. 7848, p. 54-60, 2021. 23

[13] ARUTE, Frank et al. Quantum supremacy using a programmable superconducting processor. Nature, v. 574, n. 7779, p. 505-510, 2019. 23

[14] MASLOV, Dmitri et al. Quantum advantage for computations with limited space. Nature Physics, p. 1-4, 2021. 23

[15] DESHPANDE, Abhinav et al. Quantum Computational Supremacy via High-Dimensional Gaussian Boson Sampling. arXiv preprint arXiv:2102.12474, 2021. 23

[16] KANDALA, Abhinav et al. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. Nature, v. 549, n. 7671, p. 242-246, 2017. 23

[17] ANDERSEN, Christian Kraglund et al. Repeated quantum error detection in a surface code. Nature Physics, v. 16, n. 8, p. 875-880, 2020. 17, 23, 161, 162, 163, 165, 173

[18] CHEN, Zijun et al. Exponential suppression of bit or phase flip errors with repetitive error correction. arXiv preprint arXiv:2102.06132, 2021. 23

[19] RYAN-ANDERSON, C. et al. Realization of real-time fault-tolerant quantum error correction. arXiv preprint arXiv:2107.07505, 2021. 23

[20] WOOTTON, James R. Benchmarking near-term devices with quantum error correction. Quantum Science and Technology, v. 5, n. 4, p. 044004, 2020. 23

[21] WOOTTON, James R.; LOSS, Daniel. Repetition code of 15 qubits. Physical Review A, v. 97, n. 5, p. 052313, 2018. 23

[22] NGUYEN, Nhung H. et al. Demonstration of Shor encoding on a trapped-ion quantum computer. arXiv preprint arXiv:2104.01205, 2021. 23

[23] WHEELER, John Archibald. Information, physics, quantum: The search for links. CRC Press, 2018. 25

[24] DIVINCENZO, David P.; LOSS, Daniel. Quantum information is physical. Superlattices and Microstructures, v. 23, n. 3-4, p. 419-432, 1998. 25

[25] LANDAUER, Rolf. Irreversibility and heat generation in the computing process. IBM journal of research and development, v. 5, n. 3, p. 183-191, 1961. 25, 50

[26] BENNETT, Charles H. Logical reversibility of computation. IBM journal of Research and Development, v. 17, n. 6, p. 525-532, 1973. 25, 50

[27] STONE, James V. Information theory: a tutorial introduction. 2015. 26, 28, 30, 31, 32, 34, 50

[28] PRESKILL, John. Lecture notes for physics 229: Quantum information and computation. California Institute of Technology, v. 16, n. 1, p. 1-8, 1998. 30

[29] MACWILLIAMS, Florence Jessie; SLOANE, Neil James Alexander. The theory of error correcting codes. Elsevier, 1977. 38, 40, 43, 45

[30] BALATSOUKAS-STIMMING, Alexios; FILOS-RATSIKAS, Aris. On the computational complexity of blind detection of binary linear codes. In: 2019 IEEE International Symposium on Information Theory (ISIT). IEEE, 2019. p. 2449-2453. 41

[31] TERHAL, Barbara M. Quantum error correction for quantum memories. Reviews of Modern Physics, v. 87, n. 2, p. 307, 2015. 45

[32] ZETTILI, Nouredine. Quantum mechanics: concepts and applications. 2003. 58, 60, 61, 65, 66

[33] GRIFFITHS, David J. Introduction to quantum mechanics. Pearson International Edition (Pearson Prentice Hall, Upper Saddle River, 2005), 1960. 66, 96

[34] ARFKEN, George B.; WEBER, Hans J. Mathematical methods for physicists. 1999. 68, 135

[35] BONAMENTE, Massimiliano. Statistics and analysis of scientific data. Springer, 2017. 75

[36] FUJII, Keisuke. Quantum Computation with Topological Codes: from qubit to topological fault-tolerance. Springer, 2015. 89, 142, 155

[37] WILDE, Mark M. From classical to quantum Shannon theory. arXiv preprint arXiv:1106.1445, 2011. 95

[38] SCHLOSSHAUER, Maximilian A. Decoherence: and the quantum-to-classical transition. Springer Science and Business Media, 2007. 71, 73, 74, 81, 82

[39] BREUER, Heinz-Peter et al. The theory of open quantum systems. Oxford University Press on Demand, 2002. 72, 74

[40] EINSTEIN, Albert; PODOLSKY, Boris; ROSEN, Nathan. Can quantum-mechanical description of physical reality be considered complete?. Physical review, v. 47, n. 10, p. 777, 1935. 80

[41] BELL, John S. On the einstein podolsky rosen paradox. Physics Physique Fizika, v. 1, n. 3, p. 195, 1964. 80

[42] ASPECT, Alain; GRANGIER, Philippe; ROGER, Gérard. Experimental tests of realistic local theories via Bell's theorem. Physical review letters, v. 47, n. 7, p. 460, 1981. 80, 119

[43] Qiskit Textbook. IBM. Disponível em: *https* : *//qiskit.org/textbook/preface.html*. 14, 104, 105, 106, 109, 118, 121, 122, 134, 185

[44] NAUS, H. W. L.; POLINDER, H. Bell inequality violation on small NISQ computers. arXiv preprint arXiv:2006.13794, 2020. 118

[45] BLAIS, Alexandre et al. Cavity quantum electrodynamics for superconducting electrical circuits: An architecture for quantum computation. Physical Review A, v. 69, n. 6, p. 062320, 2004. 102

[46] KRANTZ, Philip et al. A quantum engineer's guide to superconducting qubits. Applied Physics Reviews, v. 6, n. 2, p. 021318, 2019. 104, 105

[47] FUJI, K.; Introduction to the Rotating Wave Approximation (RWA): Two Coherent Oscillations. arXiv:1301.3585v3 [quant-ph] 23 May 2014. 108

[48] AARONSON, Scott. Introduction to Quantum Information Science Lecture Notes. Fall 2018. 110

[49] BENNETT, Charles H.; WIESNER, Stephen J. Communication via one-and two-particle operators on Einstein-Podolsky-Rosen states. Physical review letters, v. 69, n. 20, p. 2881, 1992. 111

[50] BENNETT, Charles H. et al. Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. Physical review letters, v. 70, n. 13, p. 1895, 1993. 113

[51] YIN, Juan et al. Satellite-based entanglement distribution over 1200 kilometers. Science, v. 356, n. 6343, p. 1140-1144, 2017. 114

[52] VALIVARTHI, Raju et al. Teleportation systems toward a quantum internet. PRX Quantum, v. 1, n. 2, p. 020317, 2020. 114

[53] LUO, Yi-Han et al. Quantum teleportation of physical qubits into logical code spaces. Proceedings of the National Academy of Sciences, v. 118, n. 36, 2021. 114

[54] KAYIRAN, Onur; YIN, Jieming; Eckert, Yasuko. (Advanced Micro Devices, Inc. - AMD) "Look-ahead teleportation for realiable computation in multi-simd quantum processor". Feb. 18, 2020. Disponível em: https://www.freepatentsonline.com/20210255871.pdf. Acesso em 11/10/2021. 114

[55] CZELUSTA, Grzegorz; MIELCZAREK, Jakub. Secure quantum communication through a wormhole. arXiv preprint arXiv:2103.14996, 2021. 114

[56] FEYNMAN, Richard P. et al. The Feynman lectures on physics, Quantum mechanics. 1966. 121, 122, 123

[57] MINEV, Zlatko. Lecture notes: Introduction to quatum noise. Qiskit Global Summer School: Machine Learning. 2021. 134

[58] ROFFE, Joschka. Quantum error correction: an introductory guide. Contemporary Physics, v. 60, n. 3, p. 226-245, 2019. 134, 143, 144, 153, 155, 157, 160, 173

[59] GOTTESMAN, Daniel. Stabilizer codes and quantum error correction. California Institute of Technology, 1997. 137

[60] CHAMBERLAND, Christopher et al. Topological and subsystem codes on low-degree graphs with flag qubits. Physical Review X, v. 10, n. 1, p. 011022, 2020. 156

[61] SÁ, Nahum. Quantum State Tomography. Disponível em: https://nahumsa.github.io/n-blog/2020-06-22-tomography/ . Acesso em 14/09/2021. 162

[62] GIOVANNETTI, Vittorio; LLOYD, Seth; MACCONE, Lorenzo. Quantum random access memory. Physical review letters, v. 100, n. 16, p. 160501, 2008. 173

[63] ARUNACHALAM, Srinivasan et al. On the robustness of bucket brigade quantum RAM. New Journal of Physics, v. 17, n. 12, p. 123010, 2015. 174

[64] WOOTTON, James R. Hexagonal matching codes with 2-body measurements. arXiv:2109.13308v1 [quant-ph] 27 Sep 2021. 173

[65] CHANCELLOR, Nicholas et al. Graphical structures for design and verification of quantum error correction. arXiv preprint arXiv:1611.08012, 2016. 173

[66] CAO, ChunJun; LACKEY, Brad. Approximate Bacon-Shor code and holography. Journal of High Energy Physics, v. 2021, n. 5, p. 1-110, 2021 174

[67] DYMARSKY, Anatoly; SHAPERE, Alfred. Quantum stabilizer codes, lattices, and CFTs. arXiv preprint arXiv:2009.01244, 2020. 174

[68] ALMHEIRI, Ahmed; DONG, Xi; HARLOW, Daniel. Bulk locality and quantum error correction in AdS/CFT. Journal of High Energy Physics, v. 2015, n. 4, p. 1-34, 2015. 174

[69] PASTAWSKI, Fernando et al. Holographic quantum error-correcting codes: Toy models for the bulk/boundary correspondence. Journal of High Energy Physics, v. 2015, n. 6, p. 1-55, 2015. 174

[70] IBM Research. Qiskit 0.29.0 documentation. Disponível em: https://qiskit.org/documentation/ . Acesso em: 18/08/2021. 185

[71] IBM Quantum. https://quantum-computing.ibm.com/, 2021. 185

[72] JAVADI-ABHARI, Ali.; GAMBETTA, Jay M. Qiskit and its Fundamental Elements. 2018. Disponível em: https://medium.com/qiskit/qiskit-and-its-fundamental-elements-bcd7ead80492 . Acesso em: 18/08/2021. 185

A Sistema Binário

A computação é baseada em sistemas binários, portanto é necessário conhecermos as operações binárias para entendermos os processos computacionais. Nos transistores presentes nos processadores clássicos, existe um mecanismo de controle de passagem de corrente elétrica que cria essa lógica binária, ou seja, presença e ausência de corrente elétrica, esse processo físico pode ser abstraído e mapeado no alfabeto binário $\mathcal{B} = \{0, 1\}$, para que possamos desenhar os circuitos lógicos responsáveis por desempenhar a tarefa desejada. A unidade fundamental do sistema binário é o bit, este pode assumir um dos valores presentes em $\mathcal{B}$, mas em geral é preciso mais de um bit para resolver os problemas de interesse, portanto precisamos realizar produtos cartesianos dos alfabetos binários para expandir nossa base de representação, por exemplo, se precisamos de dois bits devemos realizar o produto cartesiano representado abaixo

$$\mathcal{B}_{12} = \mathcal{B}_1 \times \mathcal{B}_2 = \{00, 01, 10, 11\}, \tag{A.1}$$

já para três bits é necessário adicionar mais um alfabeto binário ao produto cartesiano desenvolvido acima, de forma a obter

$$\mathcal{B}_{123} = \mathcal{B}_1 \times \mathcal{B}_2 \times \mathcal{B}_3 = \{000, 001, 010, 011, 100, 101, 110, 111\}. \tag{A.2}$$

Então se precisarmos de n bits, teremos que realizar o produto cartesiano de n alfabetos binários $\mathcal{B}$ e obtendo como resultado final um conjunto com 2^n *bitstrings*.

Além da representação, precisamos saber realizar conversões de uma base numérica para a outra e definir a aritmética binária para poder realizar as operações da computação. Vamos começar pela conversão de bases, aqui vamos tomar como exemplo a conversão da base decimal para a binária, mas vale ressaltar que existem outras bases de interesse para a computação, por exemplo a octal e hexadecimal. Uma das formas de converter um número é expandi-lo em termos de potências de 2, de início devemos buscar a maior potência de 2 que seja igual ou menor que o número que desejamos converter para binário, então ao encontrarmos a potência correta devemos multiplicá-la por 1, em seguida subtraímos uma unidade do expoente da potência anterior e verificamos se a soma dessas duas potências é igual ou menor que o número de interesse, se este for o caso multiplique-a por 1, caso contrário multiplique por 0. Este procedimento deve ser repetido até esgotarem as potências de 2, ou seja, até chegarmos a $2^0 = 1$. Abaixo podemos conferir dois exemplos demonstrando a forma descrita para realizar a conversão de um número na base decimal para a base binária.

$$42_{10} = 1(2^5) + 0(2^4) + 1(2^3) + 0(2^2) + 1(2^1) + 0(2^0) = 101010_2$$
$$137_{10} = 1(2^7) + 0(2^6) + 0(2^5) + 0(2^4) + 1(2^3) + 0(2^2) + 0(2^1) + 1(2^0) = 10001001_2.$$

Também é possível converter números reais para binário usando a mesma lógica, a única diferença é que após a vírgula devemos trabalhar com as potências negativas de 2, começando de -1 e indo até a potência inteira que resulte exatamente nas casas decimais do número que desejamos converter, então se escolhermos o número $10,25$ para ser convertido para base 2, teremos o seguinte resultado usando as regras conhecidas

$$10,25_{10} = 1(2^3) + 0(2^2) + 1(2^1) + 0(2^0), 0(2^{-1}) + 1(2^{-2}) = 1010,01_2. \quad \text{(A.3)}$$

Uma vez definida a conversão entre bases, podemos dar início a definição da aritmética binária, a fim de entender como um computador opera com os números em binário para realizar as atividades demandadas. De início, devemos definir como somar no sistema binário, para isso é preciso levar em conta as seguintes regras:

$$0_2 + 0_2 = 0_2 \quad \text{(A.4)}$$

$$0_2 + 1_2 = 1_2 + 0_2 = 1_2 \quad \text{(A.5)}$$

$$1_2 + 1_2 = 10_2. \quad \text{(A.6)}$$

Note que as duas primeiras assemelham-se as somas entre 0 e 1 no sistema decimal, contudo na terceira e última regra temos a propagação do dígito 1 para à esquerda, semelhante ao que ocorre quando a soma em decimal excede 9. Com essas regras em mãos, podemos realizar uma adição em binário para exemplificar como usá-las, para isso vamos usar os números 42 e 137 que anteriormente foram convertidos para binário, sendo assim ao realizarmos a soma seguindo as regras apresentadas acima devemos obter

$$10001001_2 + 00101010_2 = 10110011_2 = 179_{10}. \quad \text{(A.7)}$$

Existe outra classe de soma que pode ser definida na base binária, esta é conhecida como soma módulo 2, e se baseia nas seguintes regras

$$0_2 \oplus 0_2 = 0_2 \quad \text{(A.8)}$$

$$0_2 \oplus 1_2 = 1_2 \oplus 0_2 = 1_2 \quad \text{(A.9)}$$

$$1_2 \oplus 1_2 = 0_2, \quad \text{(A.10)}$$

note que é utilizado o símbolo $\oplus$ para indicar que estamos somando em módulo 2. Além disso, essa soma se difere da definida anteriormente por não apresentar a característica de carregar o dígito 1 quando se somam dois números 1_2, essa característica confere resultados diferentes para os dois tipos de soma, este fato fica evidente no exemplo abaixo, onde é realizada uma soma módulo 2 entre os números em binário correspondentes a 42 e 137 do exemplo anterior

$$10001001_2 \oplus 00101010_2 = 10100011_2. \quad \text{(A.11)}$$

Existem duas formas equivalentes de se realizar uma subtração no sistema binário, a primeira é através da definição abaixo

$$0_2 - 0_2 = 0_2 \tag{A.12}$$

$$0_2 - 1_2 = 1_2 - 0_2 = 1_2 \tag{A.13}$$

$$1_2 - 1_2 = 0_2, \tag{A.14}$$

onde se estabelecem os resultados básicos que norteiam a operação. Um detalhe importante a ser ressaltado, é que quando efetuamos uma subtração entre dois números binários e nos deparamos com uma situação do tipo $0 - 1$, devemos seguir a regra estabelecida acima, contudo devemos realizar uma propagação do dígito 1 no subtraendo a partir da posição do dígito onde ocorreu o primeiro $0 - 1$. Para exemplificar como realizar a subtração, vamos efetuar $137 - 42$ no sistema binário

$$10001001_2 - 00101010_2 = 1011111_2 = 95_{10}. \tag{A.15}$$

A maneira alternativa de fazer a subtração é usar a representação de números negativos com a operação de soma, ou seja, fazemos a conta $a + (-b)$ em vez de escrevermos a operação como $a - b$, para isso vamos precisar apresentar como representar um número negativo em binário, além disso serão necessários os conceitos de complemento de 1 e de 2. Para representar o sinal de um número inteiro em binário, usa-se a seguinte convenção: inclui-se um novo bit a esquerda do bit significativo mais à esquerda, caso o número inteiro em questão seja positivo, o novo bit deverá ser igual a 0, caso contrário será 1. Abaixo temos um exemplo de aplicação dessa convenção com os números 42 e -42

$$42_{10} = 0101010_2 \text{ e } -42_{10} = 1101010_2. \tag{A.16}$$

Outra forma de representar um número negativo em binário é através do complemento de 2, mas antes de darmos a definição dele, precisamos definir um passo anterior a ele, que é o complemento de 1, este é definido como o número binário que possui a ordem contrária de 0 e 1 do número binário original, por exemplo sabemos que $42_10 = 101010_2$, logo seu complemento de 1 deve ser igual a

$$010101_2 \text{ (complemento de 1)}. \tag{A.17}$$

Uma vez definido o complemento de 1, podemos ir para a definição do complemento de 2, esta consiste na seguinte frase: o complemento de 2 de um número binário é o resultado da adição de 1 ao algarismo mais à direita do complemento de 1 do número em questão. Para mostrar o que essa definição quer dizer, vamos usar novamente o número 42 como exemplo, então fazendo uso do resultado acima e realizando a adição abaixo, obtemos

$$010101_2 + 000001_2 = 010110_2 \text{ (complemento de 2)}. \tag{A.18}$$

Uma vez definido o complemento de 2 e sabendo realizar uma adição no sistema binário, estamos prontos para mostrar que somar um número binário com o complemento de 2 do número a ser subtraído é equivalente a operação de subtração. Para mostrar isso na prática, vamos retornar ao exemplo da subtração $137 - 42$, que agora será escrita como $137 + (-42)$, para isso devemos adequar o número de algarismos do subtraendo em binário e determinar seu complemento de 2, como mostrado abaixo

$$00101010_2 \ \rightarrow \ 11010110_2 \text{ complemento de 2,} \tag{A.19}$$

com o resultado acima em mãos, basta realizar a soma dela com o número em binário equivalente a 137, dessa forma vamos produzir o seguinte resultado

$$10001001_2 + 11010110_2 = 1011111_2 = 95_{10}. \tag{A.20}$$

Note que a soma dos algarismos mais à esquerda de ambos os números binários é igual a 10_2, portanto pela regra da adição deveria haver uma propagação de um dígito 1, mas no caso de subtração por soma com o complemento de 2 isso não se aplica, então o que se faz é ignorar a propagação para chegar ao resultado correto.

A última operação que vamos introduzir será a multiplicação de números binários, para realizá-la devemos levar em conta os seguintes resultados básicos

$$0_2 \cdot 0_2 = 0_2 \tag{A.21}$$

$$0_2 \cdot 1_2 = 1_2 \cdot 0_2 = 0_2 \tag{A.22}$$

$$1_2 \cdot 1_2 = 1_2, \tag{A.23}$$

note que estes são idênticos as multiplicações na base decimal envolvendo apenas os números 0 e 1. Um exemplo de como realizar essa operação é dado abaixo, onde fazemos a multiplicação de 42 por 2

$$101010_2 \cdot 10_2 = 1010100_2 = 84_{10}. \tag{A.24}$$

B Qiskit

O Qiskit é um kit de desenvolvimento de software de código aberto, cujo objetivo é trabalhar com computadores quânticos no nível de pulsos, circuitos e algoritmos. Seu desenvolvimento é feito pelo setor de pesquisa da IBM e pela comunidade do Qiskit [70]. A ferramenta permite executar circuitos quânticos de maneira remota nos computadores quânticos disponibilizados pela IBM em sua plataforma *IBM Quantum Experience* ou em simuladores em um computador local. O objetivo principal do Qiskit é tornar acessível o desenvolvimento de software para computadores quânticos, independente do nível de habilidade ou da área de interesse de quem estiver programando. Neste apêndice vamos apresentar a linguagem e mostrar as funções básicas para criação de algoritmos quânticos, caso o leitor queira se aprofundar no tema recomenda-se que este busque as referências [43, 70, 71].

B.1 Conceitos Básicos

A estrutura do Qiskit é dividida em quatro pilares principais: Qiskit Terra, Qiskit Aqua, Qiskit Ignis e Qiskit Aer. O primeiro faz referência ao elemento terra, fazendo assim uma alusão ao caráter de fundação do software, ou seja, o Terra é responsável por providenciar toda a sustentação para a criação de programas quânticos ao nível de circuitos e pulsos, permitindo a otimização destes para as arquiteturas particulares de cada dispositivo e a realização de experimentos através de acesso remoto. Em segundo lugar temos o elemento água, o elemento da vida, que é responsável pelas aplicações de computação quântica no mundo real, ou seja, aqui devemos encontrar os algoritmos que lidam com problemas de química, otimização e inteligência artificial, que contém a capacidade de rodar em computadores NISQ. Outro ponto importante é que o Aqua torna acessível esses algoritmos para profissionais de outras áreas, pois eles não precisam se preocupar em criar os algoritmos do zero, podem utilizar os que já estão implementados. Nosso terceiro elemento é o Ignis, o fogo, este é dedicado ao estudo e ao combate dos erros, nos fornecendo elementos para realizar uma melhor caracterização dos erros, o melhoramento de portas, realização de processos de tomografia, computação na presença de ruídos e o desenvolvimento de códigos quânticos de correção de erros. O quarto e último elemento é chamado de Aer, o ar, ele permeia todos os outros elementos, pois este é dedicado a construção de simuladores, emuladores e *debbugers*, o que torna possível desenvolver algoritmos e testá-los sem a necessidade de um computador quântico real, além disso nos ajuda a entender os limites da computação clássica [72].

Em geral, o processo de desenvolvimento e execução de algoritmos quânticos

usando o Qiskit pode ser dividido em quatro etapas:

- Construção;
- Compilação;
- Execução;
- Análise.

Na etapa de construção nós desenhamos o circuito quântico que representa o problema considerado, já a compilação prepara o circuito para rodar em um computador quântico específico ou em um simulador. Uma vez que a compilação é terminada, inicia-se o processo de execução, cujo próprio nome já define sua ação, ou seja, é nesta etapa que o algoritmo é executado em um simulador ou computador quântico real. Por fim, mas não menos importante, temos a análise dos dados gerados no processo de execução, aqui colhe-se os resultados gerados na execução e os apresenta na forma de estatística.

A estrutura apresentada acima, mostra o caminho que um algoritmo quântico percorre no processo de computação, porém não é o suficiente para compreender o desenvolvimento do algoritmo, para este fim, vamos definir, nas subseções a seguir, os seis passos apresentados abaixo, para a criação de um programa:

- Importação de pacotes;
- Inicialização de variáveis;
- Adição de portas;
- Visualização do circuito;
- Simulação do experimento;
- Visualização dos resultados.

B.1.1 Importação de Pacotes

Para dar início ao desenvolvimento de um programa que irá gerar o circuito quântico de interesse, precisamos importar os pacotes que possuem as ferramentas necessárias para o trabalho. Abaixo temos um exemplo de código que representa o passo inicial.

Listing B.1 – Exemplo de um trecho de código que realiza a importação dos pacotes a serem utilizados.

```
from qiskit.circuit import ClassicalRegister, QuantumRegister, QuantumCircuit
from qiskit import Aer
from qiskit.tools.visualization import plot_histogram
```

```
import numpy as np
```

Note que no trecho de código acima, temos dois comandos importantes, **"from"** e **"import"**, o primeiro nos diz de onde vamos importar as funções a serem usadas, o segundo nos informa os pacotes e funções que serão importados, portanto do exemplo acima podemos dizer que de **"qiskit.circuit"** estamos importando **"ClassicalRegister"**, **"QuantumRegister"** e **"QuantumCircuit"**, também podemos ver que o pacote de cálculo numérico de python, **"numpy"**, é importado e além disso temos o comando **"as"** que determina como vamos chamar o pacote no programa.

B.1.2 Inicialização de Variáveis

Uma vez feita a importação dos pacotes necessários para a construção do circuito quântico, devemos definir as variáveis que representarão os qubits e os bits e com elas definir a variável do circuito como exemplificado no código abaixo.

Listing B.2 – Código exemplificando a definição das variáveis necessárias para a criação de um circuito quântico.

```
qubits=QuantumRegister(n,name='qubits')
bits=ClassicalRegister(m,name='bits')
qc=QuantumCircuit(qubits,bits)
```

Vale ressaltar que a quantidade de qubits n e a de bits m não precisam ser iguais. Além disso, é possível ter mais de um conjunto de variáveis de qubits e bits dentro de um circuito e os bits não são uma componente essencial, pois em alguns casos não é necessário realizar medidas, portanto há a possibilidade de criar circuitos quânticos apenas com qubits. Abaixo temos uma forma de criação de circuito sem a utilização de variáveis para os qubits e bits.

Listing B.3 – Uma versão alternativa da definição do circuito quântico.

```
qc=QuantumCircuit(n,m)
```

B.1.3 Adição de Portas

Dado que o Qiskit é construído em python, que é uma linguagem de programação orientada a objetos, temos que os circuitos quânticos são considerados objetos e dentro de sua definição existem atributos e métodos, dessa forma as portas quânticas podem ser encaradas como métodos que podem ser aplicados na variável que carrega o circuito. Em geral, as portas necessitam apenas dos qubits, mas existem algumas que exigem parâmetros, por exemplo ângulos, como o caso da porta que representa o operador (3.6). Abaixo temos um exemplo de como adicionar as portas a um circuito, no caso em questão

temos um *half-adder*, ou seja, um circuito que executa a adição de dois bits de entrada com resultado contendo dois bits, este é um circuito clássico, mas pode ser realizado no contexto quântico e uma vez que estamos nesse contexto vamos usar a superposição de estados para realizar a adição de todas as combinações de dois bits em paralelo, para isso vamos precisar adicionar uma porta Hadamard (h), duas *controlled-NOT* (cx), uma *multicontrolled-NOT* (mct) e duas de medida (measure).

Listing B.4 – Código que cria um circuito de *half-adder* cuja entrada é a superposição de estados que contêm dois qubits.

```
qc.h(qubits[0:2])
qc.barrier()
qc.cx(qubits[0],qubits[2])
qc.cx(qubits[1],qubits[2])
qc.mct(qubits[0:2],qubits[3])
qc.barrier()
qc.measure(qubits[2:4],bits)
```

Note que a porta Hadamard exige apenas os qubits em que ela deve ser aplicada, contudo nesse exemplo existem portas controladas, estas necessitam de dois argumentos, os qubits de controle e os alvos, já na porta de medição precisamos passar os qubits que queremos medir e os bits que vão ser usados para o armazenamento do resultado.

B.1.4 Visualização do Circuito

Até aqui vimos como definir e modificar um circuito quântico, mas eventualmente podemos querer visualizar o circuito para conferir se tudo está nos conformes, para isso devemos lançar mão de ferramentas visuais, estas são importantes pois facilitam o processo de correção de instruções erradas e a leitura do que está sendo feito no processo de computação. Para utilizar a ferramenta de visualização devemos fazer uso do método **"draw()"**, nele podemos introduzir um argumento ou não, este é responsável por selecionar a forma que o circuito será apresentado, no primeiro exemplo que vamos ver abaixo, temos o caso em que selecionamos o pacote matplotlib para criar a imagem do circuito, um exemplo pode ser visto na Figura 77.

Listing B.5 – Código para gerar uma figura do circuito através do matplotlib.

```
qc.draw(output='mpl')
```

A forma mostrada acima é a que possui maior quantidade de recursos gráficos, as outras que serão apresentadas aqui são um pouco rudimentares, contudo conseguem realizar a tarefa de exibir o circuito de forma clara. No caso de uso do argumento **"text"** ou na ausência de um, obtemos a Figura 78.

Listing B.6 – Código que imprime o circuito na forma de texto.

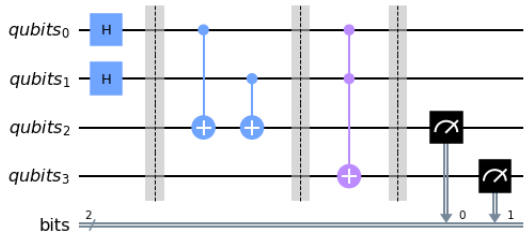


Figura 77 – Representação gráfica do circuito *half-adder* no formato matplotlib.

```
qc.draw(output='text')
```

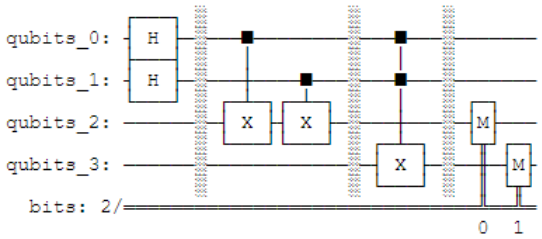


Figura 78 – Representação gráfica do circuito *half-adder* no formato texto.

Outra forma de obter o resultado acima é através do uso da função **print()** passando como parâmetro o circuito que desejamos desenhar.

Listing B.7 – Código alternativo para imprimir o circuito na forma de texto.

```
print(qc)
```

Por fim, temos a representação que é pensada para textos escritos na linguagem LaTeX, ela segue a linha da representação em texto, porém é um pouco mais estilizada. Abaixo podemos conferir o código e a figura gerada por esse método.

Listing B.8 – Código para gerar uma imagem do circuito usando o LaTeX.

```
qc.draw(output='latex')
```

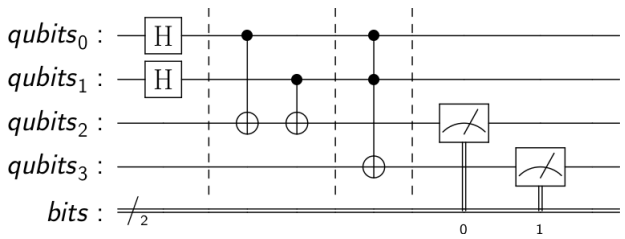


Figura 79 – Representação gráfica do circuito *half-adder* no formato LaTeX.

B.1.5 Simulação do Experimento

Uma vez que construímos o circuito e o verificamos através de ferramentas gráficas, podemos prosseguir para o passo de execução do algoritmo, aqui dispomos de duas opções: simulador e computador quântico real, contudo nesta subseção vamos nos ater apenas a primeira opção, ou seja, simuladores. Aqui podem surgir algumas questões, por que usar um simulador se existem computadores quânticos disponíveis? Quantos tipos de simulador existem? A resposta para a primeira pergunta é que no simulador podemos verificar novamente se o circuito foi fabricado corretamente, além disso podemos obter um resultado sem ruídos, o que nos permite realizar uma análise sobre a qualidade dos resultados obtidos por um computador quântico real e por fim, temos a possibilidade de testar cenários que não podemos explorar nos hardwares disponíveis no momento, devido a limitações de ruído ou quantidade de qubits.

A resposta para a segunda questão levantada no parágrafo anterior é a seguinte, no Qiskit dispomos de três simuladores, estes são chamados de *qasm simulator*, *statevector simulator* e *unitary simulator*, cada um destes possui uma função diferente, o primeiro simula um dispositivo real sem ruído, ou seja, seu resultado é uma distribuição de probabilidade dos estados medidos, cujos resultados são oriundos de multiplicações das matrizes dos operadores com a matriz do estado quântico usado na computação. O segundo nos retorna o vetor de estado resultante após a sequência de portas e o terceiro produz a matriz unitária que representa o circuito. Abaixo seguem trechos de códigos mostrando como usar cada um desses simuladores, note que há uma diferença entre o *qasm simulator* em relação aos demais, pois este necessita de um terceiro dado de entrada na função **"execute"**, isso se deve porque precisamos de várias medidas para tirar a estatística a ser analisada posteriormente, nos outros casos só é preciso calcular as matrizes que correspondem ao vetor de estado e a unitária que representa o circuito.

Listing B.9 – Código exemplificando como usar o simulador de QASM.

```
job=execute(qc,backend=Aer.get_backend('qasm_simulator'),shots=1024)
results=job.result()
counts=results.get_counts()
```

Listing B.10 – Demonstração de como usar o simulador de vetor de estado.

```
job=execute(qc,backend=Aer.get_backend('statevector_simulator'))
results=job.result()
state_vec=results.get_statevector()
```

Listing B.11 – Código que demonstra como utilizar o simulador de matriz unitária.

```
job=execute(qc,backend=Aer.get_backend('unitary_simulator'))
results=job.result()
unitary=results.get_unitary()
```

B.1.6 Execução em Computadores Quânticos

Caso nosso circuito não possua uma *depth* alta, o que pode levar a muitos erros, e não exigir muitos qubits para ser executado, podemos utilizar os computadores quânticos disponíveis na plataforma *IBM Quantum Experience* para realizar os cálculos. Para fazer tal ação devemos utilizar algumas linhas de códigos parecidas com as expostas a seguir.

Listing B.12 – Exemplo de um código para execução de circuitos quânticos nos computadores quânticos disponíveis na IBM Quantum Experience.

```
provider=IBMQ.load_account()
device=provider.get_backend('ibmq_santiago')
job=execute(qc,initial_layout=[0,1,2,3,4],backend=device,shots=8192)
job_id=job.job_id()
from qiskit.tools import job_monitor
job_monitor(job)
result=device.retrieve_job(job_id).result()
```

Note que aqui primeiro foi necessário fazer *login* na plataforma através do comando **"IBMQ.load_account()"**, depois disso escolhemos o dispositivo que irá realizar as contas e em seguida passamos o trabalho para a máquina através da função **"execute"**, note que nela temos um *input* a mais, o chamado **"initial_layout"**, que é responsável por passar para o computador quântico qual qubit do dispositivo deve ser usado para interpretar o "papel" de um qubit correspondente do circuito, esse tipo de mapeamento é extremamente importante para a fase atual da computação quântica (NISQ), pois assim conseguimos controlar a quantidade de CNOTs introduzidas no processo de compilação/transpilação, o que é algo positivo pois esse tipo de porta introduz muitos erros na computação no estágio atual. No restante do código temos funções para monitorar o status do processo de execução e recuperação dos resultados.

B.1.7 Visualização dos Resultados

Finalmente chegamos ao final da execução de nosso circuito quântico, nesta etapa buscamos formatar os resultados obtidos de uma forma que facilite sua leitura e interpretação, para isso fazemos uso de recursos visuais. Para o resultado oriundo de um simulador de *QASM* (*Quantum Assembly*) ou de um computador quântico, usamos histogramas, uma vez que o produto final é a frequência de medição dos estados quânticos, então uma vez que estamos em posse desses dados devemos usar as linhas de código apresentadas abaixo para a elaboração do histograma.

Listing B.13 – Exemplo de como fazer um histograma com os dados obtidos na medição.

```
from qiskit.visualization import plot_histogram

plot_histogram(counts)
```

Um exemplo de histograma gerado pelo trecho de código acima pode ser visto na Figura 80.

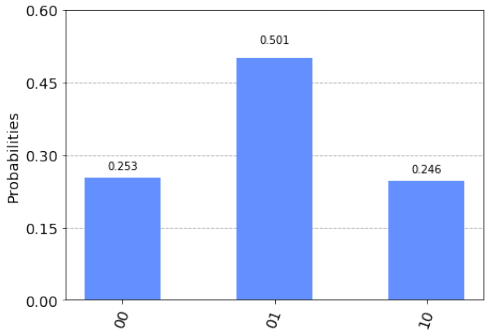
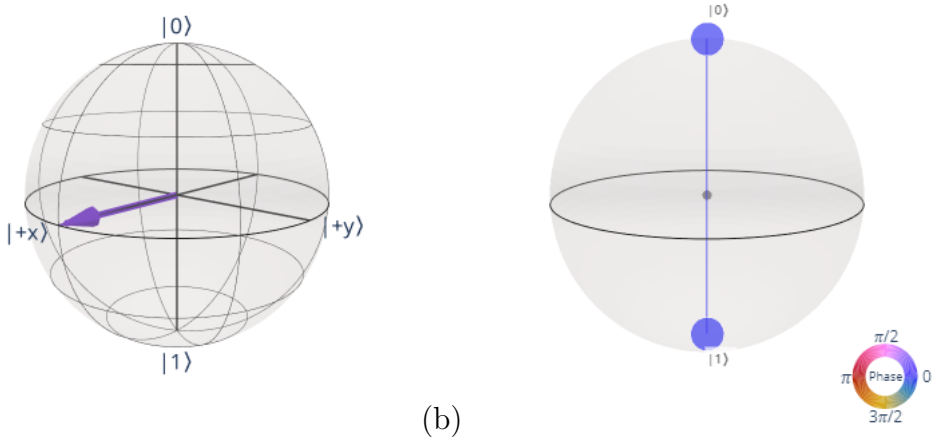


Figura 80 – Histograma representando o resultado obtido após a execução do circuito de *half-adder* com as entradas em superposição.

Caso tivermos o interesse em analisar o vetor de estado ou a matriz de densidade de um qubit, podemos fazer uso das funções **"plot_bloch_multivector()"** e **"plot_state_qsphere()"**, a primeira é útil para visualizarmos as coordenadas do vetor de Bloch do estado quântico em questão, ou seja, os valores médios das matrizes de Pauli calculados a partir do vetor de estado de interesse, além disso podemos ver se temos um estado puro ou misto, dado que podemos ver se o ponto que representa o sistema encontra-se na superfície ou no interior da esfera de Bloch. A segunda função devemos usar quando temos mais de um qubit no circuito ou quando temos o interesse de obter mais informações sobre as fases relativas que podem estar presentes no estado quântico. Abaixo encontram-se trechos de códigos mostrando como usar cada uma das funções citadas e também encontra-se a Figura 81, esta nos mostra quais são os resultados das execuções das linhas de código apresentadas.



(a) (b)

Figura 81 – (a) Exemplo de uma esfera de Bloch. (b) Exemplo de uma Qsphere.

Listing B.14 – Código para gerar uma esfera de Bloch a partir de uma matriz de densidade.

```
from qiskit.visualization import plot_bloch_multivector
from qiskit.quantum_info import DensityMatrix

dm = DensityMatrix(state_vec)

plot_bloch_multivector(dm)
```

Listing B.15 – Código exemplificando como construir uma Qsphere a partir de um vetor de estado.

```
from qiskit.visualization import plot_state_qsphere

plot_state_qsphere(state_vec)
```

C Código do Circuito do Canal de Bit-Flip

Listing C.1 – Código em Qiskit que produz o circuito que apresenta o comportamento de um canal de bit-flip Figura 49. Observação: A variável q dentro da variável environment_state presente no código na verdade representa 1-p, esta substituição foi feita devido a problemas de compilação do LaTeX na presença do caractere '-' na parte de código.

```
import random
import numpy as np
from qiskit.circuit import ClassicalRegister, QuantumRegister, QuantumCircuit

environment=QuantumRegister(1,name='environment')
qubits=QuantumRegister(3,name='qubits')
ancilla=QuantumRegister(3,name='ancilla')
bits=ClassicalRegister(3,name='bits')
ancilla_bit1=ClassicalRegister(1,name='anc_bit1')
ancilla_bit2=ClassicalRegister(1,name='anc_bit2')
ancilla_bit3=ClassicalRegister(1,name='anc_bit3')

qc=QuantumCircuit(environment,qubits,ancilla,bits,
        ancilla_bit1,ancilla_bit2,ancilla_bit3)

p=float(input('Digite a probabilidade de um bitflip ocorrer: '))

environment_state=np.array([np.sqrt(q),np.sqrt(p)])
qc.initialize(environment_state,environment)
qc.draw('mpl')

qc.barrier()
qc.h(qubits[0])
qc.cx(qubits[0],[qubits[1],qubits[2]])
qc.barrier()
qc.cx(environment,qubits[random.randint(0,2)])
qc.barrier()
qc.cx(qubits[0],ancilla[0])
qc.cx(qubits[1],ancilla[0])
qc.barrier()
qc.cx(qubits[1],ancilla[1])
qc.cx(qubits[2],ancilla[1])
qc.barrier()

qc.mct([ancilla[0],ancilla[1]],ancilla[2])
qc.cx(ancilla[2],ancilla[0:2])
qc.barrier()
```

```
qc.measure(ancilla[0], ancilla_bit1)
qc.measure(ancilla[1], ancilla_bit2)
qc.measure(ancilla[2], ancilla_bit3)
qc.barrier()
qc.x(qubits[0]).c_if(ancilla_bit1,1)
qc.x(qubits[2]).c_if(ancilla_bit2,1)
qc.x(qubits[1]).c_if(ancilla_bit3,1)
qc.barrier()
qc.cx(qubits[0],[qubits[1], qubits[2]])
qc.h(qubits[0])
qc.barrier()
qc.measure(qubits, bits)
```

D Códigos Usados no Teste do Surface Code [[4,1,2]]

Listing D.1 – Código em Qiskit que produz o circuito (c) da Figura 62. Observação: a variável minus é equivalente a -1, ela foi utilizada devido a problemas de compilação no LaTeX com o -1 dentro do código.

```
from qiskit import QuantumRegister, QuantumCircuit
import numpy as np

qubits=QuantumRegister(4,name='qubits')
ancilla=QuantumRegister(3,name='ancilla')

qc=QuantumCircuit(qubits,ancilla)

qc.ry(np.pi/2,qubits)
qc.barrier()
qc.ry(minus*np.pi/2,ancilla)
qc.cz(ancilla[1],qubits)
qc.ry(np.pi/2,ancilla[1])
qc.ry(minus*np.pi/2,qubits)
qc.barrier()
qc.cz(ancilla[0],[qubits[0],qubits[2]])
qc.cz(ancilla[2],[qubits[1],qubits[3]])
qc.ry(np.pi/2,[ancilla[0],ancilla[2]])

transpiled_qc = transpile(qc, basis_gates=['id','x','sx','cx','rz'])
```

Listing D.2 – Código em Qiskit que produz o circuito (d) da Figura 62. Observação: a variável minus é equivalente a -1, ela foi utilizada devido a problemas de compilação no LaTeX com o -1 dentro do código.

```
from qiskit import QuantumRegister, QuantumCircuit
import numpy as np

qubits=QuantumRegister(4,name='qubits')
ancilla=QuantumRegister(3,name='ancilla')

qc=QuantumCircuit(qubits,ancilla)

qc.x(qubits[0])
qc.x(qubits[2])
qc.barrier()
qc.ry(np.pi/2,qubits)
```

```
qc.barrier()
qc.ry(minus*np.pi/2,ancilla)
qc.cz(ancilla[1],qubits)
qc.ry(np.pi/2,ancilla[1])
qc.ry(minus*np.pi/2,qubits)
qc.barrier()
qc.cz(ancilla[0],[qubits[0],qubits[2]])
qc.cz(ancilla[2],[qubits[1],qubits[3]])
qc.ry(np.pi/2,[ancilla[0],ancilla[2]])

transpiled_qc=transpile(qc,basis_gates=['id','x','sx','cx','rz'])
```

Listing D.3 – Código em Qiskit que produz o circuito (c) da Figura 63.

```
from qiskit import QuantumRegister, QuantumCircuit

qubits=QuantumRegister(4,name='qubits')

qc=QuantumCircuit(qubits)

qc.h(qubits[0])
qc.cx(qubits[0],qubits[1:4])

transpiled_qc=transpile(qc,basis_gates=['id','x','sx','cx','rz'])
```

Listing D.4 – Código em Qiskit que produz o circuito (d) da Figura 63.

```
from qiskit import QuantumRegister, QuantumCircuit

qubits=QuantumRegister(4,name='qubits')

qc=QuantumCircuit(qubits)

qc.h(qubits[0])
qc.cx(qubits[0],qubits[1:4])
qc.barrier()
qc.x(qubits[0])
qc.x(qubits[2])

transpiled_qc=transpile(qc,basis_gates=['id','x','sx','cx','rz'])
```