APLICAÇÃO DE ALGORITMOS QUÂNTICOS VARIACIONAIS PARA FATORAÇÃO DE INTEIROS

Oscar Martins Wanderley Filho

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Franklin de Lima Marquezino

Rio de Janeiro
Fevereiro de 2024

APLICAÇÃO DE ALGORITMOS QUÂNTICOS VARIACIONAIS PARA FATORAÇÃO DE INTEIROS

Oscar Martins Wanderley Filho

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientador: Franklin de Lima Marquezino

Aprovada por: Prof. Celina Miraglia Herrera de Figueiredo
Prof. Luis Antonio Brasil Kowada

RIO DE JANEIRO, RJ – BRASIL
FEVEREIRO DE 2024

Wanderley Filho, Oscar Martins

Aplicação de Algoritmos Quânticos Variacionais para Fatoração de Inteiros/Oscar Martins Wanderley Filho. – Rio de Janeiro: UFRJ/COPPE, 2024.

XVII, 87 p.: il.; 29, 7cm.

Orientador: Franklin de Lima Marquezino

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2024.

Referências Bibliográficas: p. 71 – 78.

1. Fatoração de Inteiros. 2. VQF. 3. QAOA. 4. QWOA. 5. Computação Quântica. I. de Lima Marquezino, Franklin. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

"Eu te considero desafortunado
porque nunca viveste um
infortúnio. Passaste pela vida
sem um adversário — ninguém
jamais pode saber do que és
capaz, nem mesmo tu." (Sêneca)

Agradecimentos

Gostaria de agradecer, primeiramente, à minha família. Aos meus pais, Oscar e Virgínia, por todo o sacrifício que realizaram durante toda a vida para que eu pudesse ter as melhores oportunidades possíveis. Dizem que nunca se deve conhecer os seus heróis, pois pode vir a se decepcionar. Eu tive o privilégio de ter sido criado pelos meus e, ao invés de me decepcionar, fui inspirado. Nunca serei capaz de retribuir tudo o que fizeram e ainda fazem por mim, mas ainda assim sempre irei tentar. Devo tudo a vocês. À minha esposa Isabel, pelo apoio contínuo, por acreditar sempre em mim e pela paciência devido aos períodos ausentes durante essa jornada. É tentando me tornar o homem que você enxerga que me torno um indivíduo melhor.

Ao professor Franklin, meu mentor e guia nessa jornada. Agradeço por todo o apoio e incentivo destinados a mim e por ter acreditado na minha capacidade no início de todo o processo. Trilhar o caminho se torna menos árduo quando aqueles melhores que você já o fizeram e te guiam durante todo o trajeto.

Ao meu companheiro de jornada, Guilherme, que me ajudou em tantos momentos difíceis e com quem dividi o fardo desta caminhada. Não é atoa que dizem que, se quiser ir mais longe, deve ir acompanhado. Como diz o velho ditado militar: "As amizades forjadas nas agruras da selva jamais fenecem".

Por fim, e não menos importante, agradeço aos meus antigos professores e chefes, e também amigos, Stefan, Anderson e Isaac, por todos os ensinamentos e orientações, por acreditarem em mim e por me agraciarem com essa oportunidade. Existem indivíduos que, independente do período ou momento, marcam a sua vida de forma definitiva. Os senhores são três deles.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

APLICAÇÃO DE ALGORITMOS QUÂNTICOS VARIACIONAIS PARA FATORAÇÃO DE INTEIROS

Oscar Martins Wanderley Filho

Fevereiro/2024

Orientador: Franklin de Lima Marquezino

Programa: Engenharia de Sistemas e Computação

O algoritmo VQF (*Variational Quantum Factoring*), que consiste em uma abordagem híbrida de pré-processamento clássico seguido da aplicação do QAOA (*Quantum Approximate Optimization Algorithm*), foi proposto de modo a verificar o problema de fatoração de inteiros no cenário de dispositivos quânticos atuais. Foi então testada, neste trabalho, a performance desse algoritmo e de uma variação proposta baseada no algoritmo QWOA (*Quantum walk-assisted Optimization Algorithm*), uma generalização do QAOA assistido por CTQW (*continuous-time quantum walk*). Foram realizadas simulações em ambientes ideal e ruidoso, identificando o impacto altamente negativo do ruído na probabilidade de se medir os fatores corretos. Foram comparados diferentes otimizadores clássicos, BFGS e L-BFGS-B, e dois métodos de inicialização de parâmetros distintos, o original de busca em grade e um segundo baseado na superfície de custo da função objetivo. Foi identificada ainda a possibilidade de desenvolvimento de heurísticas de inicialização de parâmetros mais eficientes. Foi verificado, a partir da análise de relações numéricas, que propriedades como simetria $p \leftrightarrow q$ e presença de bits de transporte, nos fatores primos, tendem a promover um aumento da taxa de sucesso do algoritmo. Foi analisado ainda o efeito do processo de transpilação sobre as propriedades dos circuitos lógicos dos algoritmos, tendo sido constatado que a restrição imposta pelo mapa de conectividade do processador quântico supercondutor ocasiona um aumento expressivo de portas $CNOT$ e profundidade do circuito e, portanto, de ruído. Por fim, verificou-se que a variação testada (com quantum-walk mixer) apresentou resultados melhores para determinados conjuntos de instâncias (de tamanho até 6 qubits), embora tenha escalado de forma pior que a versão original (com transverse-field mixer) para outros conjuntos de instâncias (tamanho de 8 qubits).

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

APPLICATION OF VARIATIONAL QUANTUM ALGORITHMS FOR INTEGER FACTORIZATION

Oscar Martins Wanderley Filho

February/2024

Advisor: Franklin de Lima Marquezino

Department: Systems Engineering and Computer Science

The VQF algorithm (*Variational Quantum Factoring*), which consists of a hybrid approach of classical preprocessing followed by the application of QAOA (*Quantum Approximate Optimization Algorithm*), was proposed in order to verify the integer factoring problem in today's quantum device scenario. This work then tested the performance of this algorithm and a proposed variation based on the QWOA algorithm (*Quantum walk-assisted Optimization Algorithm*), a generalization of QAOA assisted by CTQW (*continuous-time quantum walk*). Simulations were carried out in ideal and noisy environments, identifying the highly negative impact of noise on the probability of measuring the correct factors. Different classical optimizers were compared, BFGS and L-BFGS-B, and two different parameter initialization methods, the original grid search and a second based on the cost surface of the objective function. The possibility of developing more efficient parameter initialization heuristics was also identified. It was verified, from the analysis of numerical relationships, that properties such as symmetry $p \leftrightarrow q$ and the presence of carry-bits in the prime factors tend to promote an increase in the algorithm's success rate. The effect of the transpilation process on the properties of the algorithms' logic circuits was also analyzed, and it was found that the restriction imposed by the connectivity map of the superconducting quantum processor causes a significant increase in $CNOT$ gates and circuit depth, and therefore in noise. Finally, it was found that the variation tested (with quantum-walk mixer) showed better results for certain groups of instances (up to 6 qubits), although it scaled worse than the original version (with transverse-field mixer) for other sets of instances (8 qubits in size).

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O problema da fatoração de inteiros (IFP) constitui a base de algoritmos criptográficos, como o RSA (*Rivest-Shamir-Adleman*) [8], que, por sua vez, estão presentes em diversos protocolos de encriptação, como o TLS/SSL (Transport Security Layer/-Secure Socket Layer), utilizados em diversas aplicações e sistemas de comunicação atuais [9, 10].

Nesse contexto, o algoritmo de Shor, publicado em artigo por Peter Shor em 1999, surge como uma ameaça quântica aos algoritmos de criptografia baseados na dificuldade de resolução do IFP, uma vez que consegue resolvê-lo, para diversos casos, superpolinomialmente mais rápido, em um computador quântico, do que o melhor algoritmo clássico sendo executado em um computador clássico [11]. Dessa forma, caso um computador quântico que pudesse executá-lo de forma eficiente existisse, a maioria dos sistemas de critografia de chave pública atuais estariam ameaçados [12, 13].

Entretanto, o cenário atual de dispositivos quânticos disponíveis não permite a sua implementação para casos práticos. Denominado NISQ era (*Noisy Intermediate-Scale Quantum*) [14, 15], ou era quântica de ruído de escala intermediária, é caracterizado por dispositivos com número limitado de qubits, presença de ruído e ausência de correção de erros. Sendo assim, no curto/médio prazo não é possível implementar o algoritmo de Shor para fatorar números criptograficamente relevantes, uma vez que é estimado que seriam necessários mais de 20 milhões de qubits para fatorar inteiros RSA de 2048 bits em 8 horas [16]. Outra estimativa realizada, baseada na existência de memória espacial adequada, mostra uma redução no requisito de qubits para 13 436 qubits, porém com o contraponto de aumento considerável de tempo (177 dias) [17]. Sendo assim, os maiores inteiros já fatorados pelo algoritmo Shor em um computador quântico real são 15 e 21, visto que os requisitos de circuito são também proibitivos [18]. Trabalhos recentes estimam ainda que seriam necessárias 2.78 x 10^{11} portas lógicas quânticas e uma profundidade de circuito[1] de 2.24 x 10^{11}

[1]Número de portas lógicas quânticas no caminho mais longo que um qubit precisa percorrer,

para fatorar inteiros de 1024 bits [18]. Já para o caso de inteiros de 2048 bits seriam necessárias 2.23 x 10^{12} portas lógicas quânticas e profundidade de 1.8 x 10^{12}. Essa alta demanda de recursos ocorre devido à presença de ruído, fator que tenta-se corrigir com a utilização de correção quântica de erros, porém essa solução demanda milhares de qubits, o que é inviável no cenário atual [19].

Faz-se necessário então o desenvolvimento de estratégias de modo a utilizar esses dispositivos NISQ para possivelmente alcançar vantagem quântica. Com esse objetivo em foco surgiram os algoritmos quânticos variacionais (*Variational Quantum Algorithms* - VQA) [1], estratégia promissora fundamentada no uso de um otimizador clássico para otimizar um circuito quântico parametrizado. Realiza-se assim um trade-off entre recursos clássicos e quânticos.

Com esse cenário em foco, foram propostas algumas abordagens, inspiradas em diferentes estratégias. Uma abordagem utilizada foi uma combinação entre *Quantum Annealing* e geometria algébrica computacional (especificamente Bases de Gröbner) [20]. Já outra, também baseada em *Quantum Annealing*, utilizou modelos HUBO (*higher-order unconstrained optimization*) e QUBO (*quadratic unconstrained binary optimization*) [21] para modelar as funções de custo. Houve ainda abordagem utilizando o algoritmo generalizado de Grover para fatorar tetra e pentaprimos (números formados pelo produto de quatro e cinco primos, respectivamente) [22]. Nos casos citados, houve também a transformação do problema de fatoração de inteiro em um problema de otimização binária combinatória.

Ainda no campo dos VQAs, um dos algoritmos mais promissores é o QAOA (*Quantum Approximate Optimization Algorithm*) [23], utilizado para a resolução de problemas de otimização, algoritmo esse que foi posteriormente generalizado de modo a permitir a alternância entre famílias mais gerais de operadores, culminando no *Quantum Alternating Operator Ansatz*, de mesma sigla [24]. Uma abordagem baseada nesse algoritmo é uma combinação híbrida do uso do QAOA para otimizar o algoritmo clássico de fatoração de *Schnorr* [25].

Seguindo nesse caminho, Anschuetz et al. propuseram um algoritmo híbrido baseado no QAOA, o VQF (*Variational Quantum Factoring*) [2], abordagem que consiste em um pré-processamento clássico seguido da aplicação do referido algoritmo. Outros trabalhos deram continuidade ao original, testando a sua resiliência a erros de portas lógicas quânticas (erro de leitura) e o efeito de novas transformações do hamiltoninano de custo [26], comparando a sua performance e utilização de recursos variando o QAOA pelo VQE (*Variational Quantum Eigensolver*) [19] e, por fim, analisando a resiliência a diferentes tipos de ruído do VQF original e a sua performance quando executado em um processador quântico supercondutor real [4].

Sendo assim, visando testar novas variações e possivelmente encontrar resultados

assim como de outras portas com as quais interage, desde a entrada até a saída do circuito.

promissores, esse trabalho se propôs a analisar a performance do algoritmo VQF com uma nova variação, substituindo o algoritmo QAOA pelo algoritmo QWOA (*Quantum walk-assisted Optimization Algorithm*) [27]. O QWOA é considerado uma generalização do QAOA assistida por CTQW (*continuous-time quantum walk*) [28], e a sua aplicação já foi numericamente analisada em problemas como Roteamento de Veículos Capacitados [29] e Otimização de Portfólio [30]. Cabe ressaltar que, devido à dificuldade de se obter resultados analíticos para a análise do algoritmo, sua performance deve ser caracterizada de forma experimental [19].

Dessa forma, pode-se elencar as principais contribuições do trabalho:

1. Análise da performance do VQF com a variação baseada em QWOA e da versão original. Compara-se as duas variações em relação às métricas de Taxa de Sucesso, valor mínimo da função objetivo e número de avaliações da função objetivo necessárias até o determinado otimizador clássico utilizado convergir.

2. Investigação das hipóteses de relações numéricas deixadas pelos autores no artigo original. No artigo original é levantada a hipótese de que a presença de bits de transporte (*carry-bits*) e de simetria nos fatores que constituem o biprimo a ser fatorado influenciam na probabilidade deste ser fatorado com sucesso. Tal hipótese é testada e verificada no trabalho realizado.

3. Teste das variações de algoritmos clássicos de otimização e de inicialização de parâmetros. Testa-se dois algoritmos clássicos distintos, BFGS e L-BFGS-B, demostrando que o segundo aproxima bem o primeiro, além de constituir uma boa opção para utilização eficiente de recursos; e é observado e testado um comportamento para a superfície de custo da função objetivo que pode permitir a construção de heurísticas de inicialização de parâmetros de modo a reduzir substancialmente a complexidade do algoritmo.

4. Análise das performances das duas variações sobre a presença de ruído. Verifica-se o impacto sobre a taxa de sucesso do algoritmo, para as duas versões, do ruído característico de um processador quântico supercondutor.

5. Análise das propriedades dos circuitos lógicos quânticos utilizados e do efeito da transpilação para execução no simulador que simula o processador quântico supercondutor. São analisadas as propriedades dos circuitos como número de portas lógicas quânticas, seus tipos, profundidade e como escalam com o número de camadas do algoritmo. É analisado também o efeito do processo de transpilação sobre tais propriedades.

Pode-se agora, por fim, explicitar a organização desta dissertação: No capítulo 2 é realizada uma contextualização do cenário atual para a fatoração quântica e das

soluções propostas baseadas no arcabouço dos algoritmos quânticos variacionais. Já no capítulo 3 é apresentado, em detalhes, o funcionamento do VQF, assim como da sua variação baseada no QWOA e da métrica de sucesso utilizada. Em sequência, no capítulo 4, são detalhadas todas as simulações realizadas, tanto em ambiente ideal como com ruído, e os resultados obtidos, além de explicar, detalhadamente, o processo de construção dos circuitos lógicos utilizados e dos circuitos transpilados para execução no processador quântico supercondutor. Finalmente, no capítulo 5 são desenhadas as conclusões e elencados possíveis trabalhos futuros.

Capítulo 2

Contextualização do Problema

2.1 Criptografia e sistemas de comunicação

A história da criptografia acompanha a própria história da humanidade. Embora a criptologia [31] seja uma ciência relativamente nova, há relatos do uso de criptografia, para esconder mensagens secretas, há milhares de anos.

2.1.1 Uma breve história da criptografia

Acredita-se que a primeira aplicação de criptografia ocorreu há aproximadamente 4000 anos, no Egito. Foram identificadas, na cidade de Menet Khufu, na tumba do nobre Khnumhotep II, inscrições de hieróglifos que foram escritos com um número incomum de símbolos de modo a confundir ou esconder o real significado das inscrições [32]. Embora não tenha sido uma aplicação padrão de criptografia, o texto continha um de seus componentes essenciais, a modificação do texto.

Outro exemplo, ainda na antiguidade, consiste no dispositivo criptográfico inventado pelos Espartanos, em 5 A.C., para enviar e receber mensagens secretas. Tal dispositivo, denominado Cítala (*Scytale*), consistia em um bastão cilíndrico e um pedaço de couro ou papiro. O remetente então enrolava o bastão ou o papiro no bastão, escrevia a mensagem e então enviava apenas o material no qual havia sido escrita, de modo que exibia apenas uma sequência de caracteres, aparentemente aleatórios, que não apresentavam qualquer significado. O receptor então recebia o material, o enrolava em um dispositivo cilíndrico de mesmo diâmetro e assim conseguia ler a mensagem. Foi criado assim o primeiro código de cifra de transposição, ou seja código em que os caracteres são mantidos, porém a sua ordem é trocada. Fica caracterizada ainda uma das principais aplicações da criptografia ao longo da história: como arma de guerra.

Com o passar dos séculos, e até milênios, outras técnicas foram criadas. Pode ser citada a Cifra de César, que foi a primeira cifra de substituição (simples) utilizada,

ou seja em que caracteres são substituídos, individualmente, por outro de acordo com um alfabeto de substituição. Outro caso é o do criptógrafo francês Blaise de Vigenère que, baseado no trabalho de cifras de sistemas poli alfabéticos desenvolvido pelo italiano Leon Battista Alberti, concebeu o primeiro sistema prático dessa classe, que ficou conhecido como Cifra de Vigenère. Durante séculos esta técnica foi considerada inquebrável.

Entretanto, posteriormente foi desenvolvida uma técnica de ataque estatístico, baseada na frequência de caracteres em uma determinada linguagem, que quebrou a Cifra de Vigenère. Essa técnica ficou conhecida como o Teste de Kasiski devido ao seu desenvolvedor, o prussiano Kasiski. Por outro lado, é dito que o primeiro a desenvolver a técnica foi o inglês Charles Babbage, por volta de 1946 a 1954, tal que já conseguia decriptar mensagens encriptadas pelo uso da Cifra de Vigenère. Esse fato foi possivelmente mantido em segredo devido ao uso do seu trabalho pelos ingleses para decifrar as mensagens trocadas na Guerra da Criméia [32].

Percebe-se que, assim como a tecnologia no geral, a guerra foi sempre uma grande promotora de inovações no campo da criptologia. Um caso emblemático consiste na máquina ENIGMA, utilizada pelos alemães na 2ª Guerra Mundial e o esforço de guerra do ULTRA (codenome inglês para SIGINT - *Signal Intelligence*), denominação dada pelos ingleses para as informações de inteligência de alto nível do Eixo decriptografadas e, em específico, das mensagens geradas pela ENIGMA.

Munidos dos resultados dos trabalhos desenvolvidos pelos poloneses na década anterior, em especial pelo matemático Marian Rejewski, os ingleses conseguiram desenvolver Bombes (dispositivo mecânico desenvolvido pelos poloneses para extrair a chave da ENIGMA) mais eficientes e métodos para extrair as chaves diárias da ENIGMA, feito primordial para a vitória dos aliados e fim acelerado da guerra. Grande parte do trabalho original realizado pela equipe de criptoanalistas inglesa pode ser atribuída ao matemático inglês Alan Turing, conhecido como o pai da ciência da computação moderna.

Dessa forma, até a 2ª Guerra Mundial e nos anos posteriores, grande parte dos avanços na área da criptologia foi incentivada pelas aplicações em guerras. O autor David Kahn, em sua obra *The Code Breakers* [33], desenvolve com profundidade todos os exemplos citados anteriormente, além de inúmeros outros, demonstrando como a criptografia e, posteriormente, a criptoanálise, e seus agentes, moldaram a história da humanidade, desde o Egito antigo até o mundo atual; atuando em guerras, diplomacia, espionagem etc.

2.1.2 Advento do mundo digital

A partir dos anos 1980 e, principalmente, a partir do final da década de 90, o mundo evoluiu para uma era digital. Diversos meios de comunicação surgiram e/ou foram digitalizados, como a *Internet*, redes de rádio, redes de telefones móveis etc; além de diversas aplicações e serviços, fazendo que não só empresas, como também usuários finais passassem a se preocupar com a segurança dos seus dados e informações.

Nesse contexto, foram desenvolvidos novos algoritmos de criptografia de modo a serem utilizados em protocolos de comunicação e para proteção de dados fixos. Atualmente, a NIST (*National Institute of Standards and Technology*) [34], agência americana não reguladora que promove a inovação por meio do avanço da ciência, padrões e tecnologia de medição, define os padrões recomendados a serem utilizados para cada de tipo algoritmo de criptografia, dependendo do objetivo que visa-se alcançar. Esses algoritmos aprovados, por sua vez, são divididos em três grandes classes, que são definidas pelo número ou tipo de chaves criptográficas que são utilizadas com os mesmos [35]: funções de *Hash*, algoritmos de chave simétrica e algoritmos de chave assimétrica.

As funções de *Hash* criam um pequeno resumo (valor de *Hash*) a partir de um grande volume de dados por meio de um processo de caminho único, ou seja cujo caminho inverso é computacionalmente muito difícil. São normalmente utilizadas em soluções de gerenciamento de chaves criptográficas e provisão de serviços como verificação de assinaturas digitais, geracão de códigos de autenticação de mensagens (MACs), geração de números aleatórios determinísticos etc. Como exemplos pode-se citar o SHA (*Secure Hash Standard*) e o SHA3 (*Secure Hash Standard 3*).

Já os algoritmos de chave simétrica transformam dados de modo que seja extremamente difícil ler o seu conteúdo sem a posse da chave criptográfica secreta utilizada no processo; chave essa denominada simétrica pois é utilizada tanto para encriptação quanto para a decriptação. São utilizados para prover confidencialidade de dados, geração de MACs (criação e validação), estabelecer chaves durante processos de estabelecimento de chaves etc. Tem-se o AES (*Advanced Encryption Standard*) e o TDEA (*Triple Data Encryption Algorithm*) como exemplos.

Por fim, os algoritmos de chave assimétrica utilizam um par de chaves (uma pública e outra privada) para executar o determinado objetivo em foco, sendo frequentemente denominados de algoritmos de chave pública. A chave pública é de conhecimento geral, enquanto a chave privada é de conhecimento apenas de quem gerou o par. Caso o objetivo seja garantir a confidencialidade da mensagem, utiliza-se a chave pública para criptografar os dados e apenas quem possuir a chave privada será capaz de decriptá-los e ler o seu conteúdo. Entretanto, caso o objetivo seja garantir a autenticidade de quem escreveu a mensagem, utiliza-se a chave privada

para criptografar os dados e, ao utilizar a chave pública, será possível identificar o correto emissor. São utilizados para gerenciamento de identidades, estabelecimento de chaves entre partes e computar assinaturas digitais. Como exemplo, pode-se citar o DSA (*Digital Signature Algorithm*), RSA (*Rivest-Shamir-Adleman*) e ECDSA (*Elliptic Curve Digital Signature Algorithm*).

2.1.3 A ameaça quântica

Em 1994, Peter Shor, professor no MIT (*Massachusetts Institute of Technology*), apresentou em conferência [36] um algoritmo, oficialmente publicado em artigo em 1999, conhecido como Algoritmo de Shor [11], capaz de realizar fatoração de inteiros e encontrar logaritmos discretos em tempo polinomial, sendo superpolinomialmente mais rápido que o melhor algoritmo clássico existente executado em um computador clássico.

Surgiu assim uma ameaça aos algoritmos de criptografia baseados na dificuldade de se fatorar inteiros e encontrar logaritmos discretos, como o RSA, DSA e ECDSA, citados anteriormente [12]. Dessa forma, passou a constitituir uma possível ameaça à criptografia de chave pública em geral, assumindo que um dispositivo quântico em que pudesse ser executado realmente existisse.

Nesse contexto, consistiu também em uma ameaça aos meios e sistemas de comunicação em geral, visto que diversos dos protocolos de encriptação utilizados em comunicação utilizam os referidos algoritmos de criptografia de chave pública mencionados.

Sendo assim são ou já foram utilizados com o TLS/SSL (*Transport Security Layer/Secure Socket Layer*) de modo a garantir a segurança de comunicações entre dois indivíduos, como por exemplo entre o lado cliente (*navegadores web*) com servidores (aplicações web), com PGP (*Pretty Good Privacy*) para encriptar mensagens e assiná-las digitalmente, principalmente no caso de e-mails. Também são ou já foram utilizados em VPNs (*Virtual Private Network*), provendo canais seguros criptografados ponta a ponta, assim como com SSH (*Secure Shell*) , funcionando de forma similar à VPN, de modo a permitir a conexão remota segura à computadores, servidores, dentre outros [9, 10].

Entretanto, conforme será visto na seção seguinte, o cenário dos dispositivos quânticos existentes, devido a uma série de limitações de hardware, não permite a execução do algoritmo de Shor de forma prática para problemas condizentes com os cenários reais encontrados. Devido a isso, possíveis soluções alternativas foram propostas e estão sendo estudadas e testadas de modo a responder se a criptografia de chave pública atual está segura em relação a ataques quânticos baseados no algoritmo de Shor até que a sua implementação eficiente seja possível ou se o curto/médio prazo

já apresenta ameaças possivelmente relevantes.

2.2 Fatoração na computação quântica

Conforme mencionado na seção anterior, os algoritmos de Shor demonstraram a capacidade de um computador quântico em resolver o problema da fatoração de inteiros (IFP), o problema do logaritmo discreto de campo finito (DLP) e o problema do logaritmo discreto de curva elíptica (ECDLP) [12] superpolinomialmente mais rápido que um computador clássico. Este trabalho será focado no primeiro dos problemas mencionados (IFP) e, portanto no algoritmo RSA [8].

Entretanto, um ponto importante a ser levantado, conforme mencionado brevemente na introdução, é o estado atual da computação quântica, conhecido como *Noise Intermediate-Scale Quantum era* (NISQ era). Nesse cenário, os dispositivos quânticos disponíveis possuem número de qubits limitado, não são tolerantes a falhas, apresentam alta sensibilidade a ruído e não permitem a implementação de correção de erro necessária. Sendo assim, a implementação da fatoração inteira de números criptograficamente relevantes via algoritmo de Shor ainda não é possível, uma vez que este requer um elevado número de qubits para correção de erro e uma grande profundidade de circuito para ser implementado, o que, por sua vez, torna-se inviável devido ao ruído gerado.

Sendo assim, baseado no arcabouço dos VQAs, que será descrito na seção 3.22, foram desenvolvidas diversas propostas de soluções, conforme citado na introdução, buscando tornar a fatoração viável utilizando os meios disponíveis no curto/médio prazo e determinar se os sistemas de comunicação baseados em esquemas de chave pública estão completamente seguros ou se já há a possibilidade de ameaças antes da implementação eficiente do algoritmo de Shor ser possível.

Dessa forma, devido a sua fundamentação no QAOA, algoritmo altamente promissor no âmbito dos VQAs, optou-se por estudar e testar diferentes implementações para o VQF. Este, por sua vez, consiste em uma abordagem híbrida clássico/quântica, baseada em um pré-processamento clássico seguido da aplicação do *Quantum Approximate Optmization Algorithm* (QAOA), de acordo com a menção breve realizada na introdução.

Inicialmente, o algoritmo mapeia o problema de fatoração em um modelo de Hamiltoniano de Ising, reduzindo o número de qubits necessário, com o uso de esquemas de pré-processamento clássico, sempre que possível. Em seguida, é treinada a ansatz [1] QAOA para o respectivo hamiltoniano, por meio do uso de algoritmos de otimização clássicos.

[1] Circuito quântico parametrizado utilizado para criar um estado inicial experimental que precisa ser especificado para o problema em foco.

Por fim, o problema é transformado em um problema de otimização combinatória de minimização, em que o objetivo é encontrar o estado que minimiza o valor esperado do hamiltoniano, que corresponde a encontrar o autovalor mínimo, ou seja o seu estado fundamental de energia. É nesse estado que está codificada a resposta para o problema original [37]. Segue-se então, finalmente, para a caracterização do arcabouço dos VQAs.

2.3 Algoritmos Quânticos Variacionais

Como discutido na seção anterior, os dispositivos quânticos disponíveis atualmente apresentam diversas limitações. Nesse contexto, faz-se necessário o desenvolvimento de estratégias de modo a utilizar esses dispositivos NISQ para possivelmente alcançar vantagem quântica.

Sendo assim, uma estratégia promissora com esse objetivo consiste nos Algoritmos Quânticos Variacionais (*Variational Quantum Algorithm* - VQA), grupo a que pertence o VQF apresentado anteriormente. Cerezo et al. [1] apresentam um guia extenso para o cenário atual dos VQAs, abordando desde os blocos construtores dos algoritmos (função de custo, ansatz, otimizador clássico etc), como também diferentes aplicações para as quais já há propostas de algoritmos (otimização, encontrar estados fundamentais e excitados de Hamiltonianos, compilação, correção de erro etc), discutindo ainda sobre os desafios enfrentados atualmente (treinamento e inicialização de parâmetros, e construção de ansatz específica para o problema), e até mesmo abordam possíveis desdobramentos futuros (possibilidade de alcance de vantagem quântica, aplicações em química e ciências dos materiais, em física nuclear e de partículas, em aprendizado de máquina etc).

Basicamente, os VQAs utilizam um otimizador clássico para treinar um circuito quântico parametrizado, de modo a promover um trade-off entre recursos clássicos e quânticos. Dessa forma, serão discutidos, de acordo com o apresentado pelos autores, os blocos construtores e conceitos básicos, que podem ser observados na Fig.2.1, de modo a introduzir o arcabouço sobre o qual é baseado o VQF, que, por sua vez, será abordado de forma mais profunda no Cap.3.

Nesse contexto, ainda em relação ao diagrama apresentado na Fig. 2.1, tem-se como entradas para o algoritmo uma função de custo $C(\theta)$ parametrizada, um conjunto θ de parâmetros que codificam a solução para o problema e que devem ser treinados, uma ansatz cujos parâmetros são treinados de modo a minimizar (ou maximizar) a função de custo e, por fim, há a possibilidade de um conjunto de dados de treinamento $\{\rho_k\}$. Cabe ressaltar que no VQF apenas os três primeiros estão presentes. A ansatz é representada como um circuito quântico parametrizado, que é análogo a uma rede neural.

A cada iteração do laço do algoritmo, o computador quântico é utilizado para estimar a função de custo. Em seguida, o resultado é passado para um computador clássico que utiliza um otimizador clássico para percorrer a superfície de custo e assim resolver o problema de otimização. Há diferentes formas possíveis para as saídas, as quais dependem do problema específico em foco: estado quântico, distribuição de probabilidade, bitstring, sequência de portas lógicas ou até mesmo um operador quântico. No caso do VQF, a saída procurada é dada por bitstrings. Pode-se agora apresentar de forma mais detalhada os elementos, mencionados anteriormente, que estão presentes no algoritmo em foco.

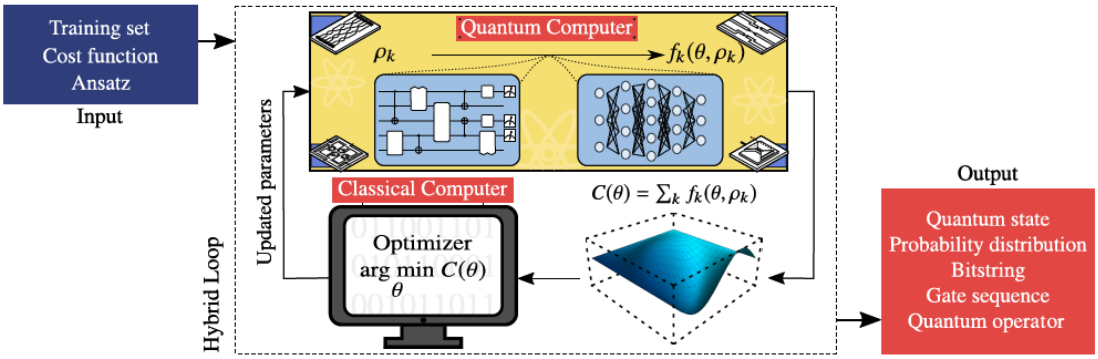


Figura 2.1: Diagrama esquemático do VQA. Figura retirada de Cerezo et al. [1].

Primeiramente, a partir do tipo de problema a ser tratado, deve-se desenvolver uma função de custo C que codifica a solução para o mesmo. Em seguida, define-se uma ansatz parametrizada em função de um conjunto de parâmetros θ que serão treinados, ou seja otimizados para o caso em foco. Dessa forma, a ansatz é treinada em um laço híbrido clássico-quântico de modo a resolver o problema de otimização:

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg\min_{\theta} C(\boldsymbol{\theta}) \text{ ou} \\ \boldsymbol{\theta}^* &= \arg\max_{\theta} C(\boldsymbol{\theta}) \end{aligned} \tag{2.1}$$

Sendo assim, o VQA utiliza o computador quântico para estimar a função de custo $C(\theta)$, enquanto utiliza um otimizador clássico para treinar o conjunto de parâmetros θ (busca os parâmetros ótimos de acordo com o objetivo). Pode-se então definir, de forma genérica, a função de custo como:

$$C(\boldsymbol{\theta}) = f\left(\{\rho_k\}, \{O_k\}, U(\boldsymbol{\theta})\right) \tag{2.2}$$

onde f é uma determinada função, $U(\theta)$ é um unitário parametrizado, θ é um

conjunto de parâmetros, $\{\rho_k\}$ são estados de entrada de um conjunto de treinamento e $\{O_k\}$ é um conjunto de observáveis físicos.

Portanto, a função de custo mapeia valores, a partir dos parâmetros θ, para os reais. Pode-se ainda observar que a função de custo define uma hipersuperfície, denominada também de superfície de custo, a qual é navegada pelo otimizador clássico de modo a identificar mínimos ou máximos globais (de acordo com o objetivo do problema sendo trabalhado).

Em seguida, os autores definem critérios desejáveis que a função de custo deve apresentar:

1. A função de custo deve ser fiel no sentido de que o mínimo de $C(\theta)$, no caso de um problema de minimização, deve corresponder à solução do problema (o inverso ocorrendo para um problema de maximização).

2. Deve ser possível estimar a função de custo de forma eficiente, em um computador quântico, por meio da realização de medições e possivelmente com a aplicação de pós-processamento.

3. A função de custo deve ainda ter significado operacional, ou seja menores valores devem indicar uma melhor qualidade de solução, no caso de problemas de minimização (o inverso deve ocorrer para problemas de maximização).

4. Por fim, a função de custo deve ser treinável, ou seja deve ser possível otimizar de forma eficiente o conjunto de parâmetros θ do qual depende.

Após a definição da função de custo, pode-se passar para a ansatz. A sua forma define o que são os parâmetros θ e como podem ser treinados. Como exemplo, conforme será visto nas seções 3.2 e 3.3, estes representam o tempo de execução na evolução adiabática trotterizada, no primeiro caso, e da realização de uma quatum-walk (caminhada quântica), no segundo. Usualmente, a sua estrutura específica irá depender do problema que se pretende resolver, embora tal fato não seja obrigatório, visto que existem estruturas mais genéricas e/ou independentes do problema.

Sendo assim, os parâmetros θ podem ser codificados em um operador unitário $U(\theta)$ do qual depende a função de custo $C(\theta)$, assim como visto anteriormente na Eq.(2.2). Este operador, por sua vez, é aplicado nos estados de entrada do circuito. Pode-se agora definir matematicamente, de forma genérica, o operador $U(\theta)$ como o produto de L unitários aplicados de forma sequencial:

$$U(\boldsymbol{\theta}) = U_{\boldsymbol{L}}\left(\boldsymbol{\theta}_{\boldsymbol{L}}\right) \cdots U_2\left(\boldsymbol{\theta}_2\right) U_1\left(\boldsymbol{\theta}_1\right) \tag{2.3}$$

tal que:

$$U_l(\theta_l) = \prod_m e^{-i\theta_m H_m} W_m \tag{2.4}$$

onde W_m é um unitário não parametrizado, H_m é um operador Hermitiano e θ_l é o l-ésimo elemento em θ.

Finalmente, definidos os conceitos básicos relacionados ao VQA, arcabouço no qual o VQF é baseado, pode-se avançar para a descrição detalhada do algoritmo em si, conforme será visto no Cap.3.

Capítulo 3

Fatoração Quântica Variacional (VQF)

Conforme mencionado brevemente na seção 2.2, o algoritmo VQF consiste em uma abordagem híbrida clássico/quântica, baseada em um pré-processamento clássico seguido da aplicação do *Quantum Approximate Optmization Algorithm* (QAOA). Nesse contexto, esse capítulo tem como objetivo descrever com profundidade cada uma de suas etapas, iniciando pelo pré-processamento clássico.

3.1 Pré-processamento clássico

O pré-processamento clássico representa a etapa inicial do algoritmo. Em um primeiro momento, a fatoração de inteiros, no caso deste trabalho a decomposição de números biprimos (semiprimos, ou seja formados pelo produto de dois números primos) nos seus fatores, é modelada como um problema de otimização binária.

Em seguida, é associada uma cláusula a cada equação do sistema de equações binárias obtido no passo anterior. Com o objetivo de reduzir o número de cláusulas e variáveis binárias como um todo, é então aplicado um esquema de regras de pré-processamento clássico. Finalmente, o sistema final de cláusulas simplificadas é mapeado para um Hamiltoniano de Ising, em cujo estado fundamental estão codificadas as soluções dos fatores. Nas seções 3.1.1-3.1.3 cada um desses passos é detalhado.

3.1.1 Fatoração como otimização binária

Burges [3] demonstrou que o problema de fatoração de inteiros pode ser mapeado em um problema de otimização combinatória binária de minimização. Nesse contexto, pode-se observar a fatoração do biprimo $m = p.q$, em que p e q são os fatores

primos que o compõem. Primeiramente, deve-se colocar m, p e q nas suas expansões binárias:

$$m = \sum_{k=0}^{n_m-1} 2^k m_k,\ p = \sum_{k=0}^{n_p-1} 2^k p_k,\ q = \sum_{k=0}^{n_q-1} 2^k q_k \tag{3.1}$$

onde $m_k \in \{0,1\}$ é o k-ésimo bit de m e n_m é o número de bits do biprimo m. Analogamente define-se p_k, q_k, n_p e n_q.

A partir das expressões definidas na Eq. (3.1), pode-se realizar a multiplicação binária $m = p.q$ e obter o seguinte sistema de equações:

$$0 = \sum_{j=0}^{i} q_j p_{i-j} + \sum_{j=0}^{i} z_{j,i} - m_i - \sum_{j=1}^{n_m-1} 2^j z_{i,i+j} \tag{3.2}$$

$\forall\ 0 \leq i < n_m$, onde $z_{i,j} \in \{0,1\}$ representa o carry-bit da posição i para a posição j e $z_{i,i+j} = 0$, onde $(i+j) > n_m - 1$. O carry-bit (bit de transporte) representa o bit "vai um" de uma coluna para a outra na tabela de multiplicação binária, conforme Fig. 3.1.

Entretanto, pensando em implementação, deve-se ter um ponto de partida para os tamanhos de p e q. Sendo assim, primeiramente, deve-se notar a seguinte propriedade:

Em qualquer fatoração de um inteiro positivo T em dois fatores F_1 e F_2, com $F_1 \geq F_2$, então $n_{F_2} \leq \left\lceil \frac{n_T}{2} \right\rceil$ [3].

Dessa forma, quando p e q são desconhecidos (o que ocorre quando se recebe apenas um número m para ser fatorado), pode-se assumir, sem perda de generalidade, que $p \geq q$, $n_p = n_m$ e $n_q = \left\lceil \frac{n_m}{2} \right\rceil$[2].

Pode-se então reescrever a Eq. (3.2) obtendo a nova equação:

$$0 = \sum_{j=0}^{i} q_j p_{i-j} + \sum_{j=0}^{i} z_{j,i} - m_i - \sum_{j=1}^{n_c-1} 2^j z_{i,i+j} \tag{3.3}$$

$\forall\ 0 \leq i < n_c$, onde $n_c = n_p + n_q - 1$ e $z_{i,j} \in \{0,1\}$ representa o carry-bit da posição i para a posição j. Tem-se ainda que $m_i = 0$, onde $i > n_m - 1$ e $z_{i,i+j} = 0$, onde $(i+j) > n_c - 1$.

Sendo assim, tem-se que o sistema expresso pela Eq. (3.3) possui $n_c = n_p + n_q - 1$ equações, tal que $n_c \in O(n_m)$ [2]. Deve-se ressaltar ainda que as Eqs. (3.2) e (3.3) são obtidas a partir da soma, por coluna, dos valores na tabela de multiplicação binária, como pode ser visto no exemplo dado pela Fig. 3.1, que demonstra a multiplicação binária do biprimo 15 = 3 x 5, com as considerações anteriores.

		p_2	p_1	p_0
$\times$			q_1	q_0
		p_2q_0	p_1q_0	p_0q_0
$+$	p_2q_1	p_1q_1	p_0q_1	
$+$	$z_{0,3}$	$z_{0,2}$	$z_{0,1}$	$-2z_{0,1}$
$+$	$z_{1,3}$	$z_{1,2}$	$-2z_{1,2}$	$-4z_{0,2}$
$+$	$z_{2,3}$	$-2z_{2,3}$	$-4z_{1,3}$	$-8z_{0,3}$
	1	1	1	$1.$

Figura 3.1: Tabela exemplificando o caso da multiplicação binária de 15 = 3 x 5, cuja forma binária é dada por $15_{bin} = 1111$. As equações que compõem o sistema expresso pelas Eqs. (3.2) e (3.3) são obtidas a partir da soma dos temos em cada coluna, respectivamente. - Foi considerado, nesse esquema, que os tamanhos de p e q eram previamente conhecidos de modo a reduzir o número total de bits. Figura retirada de Anshuetz et al. [2].

Em seguida, pode-se associar uma cláusula C_i, sobre os inteiros ($\mathbb{Z}$), para cada equação, de tal forma que:

$$C_i = \sum_{j=0}^{i} q_j p_{i-j} + \sum_{j=0}^{i} z_{j,i} - m_i - \sum_{j=1}^{n_c-1} 2^j z_{i,i+j} \tag{3.4}$$

Dessa forma, a fatoração pode ser representada como encontrar as atribuições das variáveis binárias $\{p_i\}$, $\{q_i\}$ e $\{z_{ij}\}$ que resolvem o sistema de cláusulas:

$$\sum_{i=0}^{n_c-1} C_i^2 = 0 \tag{3.5}$$

visto que é demonstrado por Borges [3] que $F(m) = \sum_{i=0}^{n_c-1} C_i^2$ possui um mínimo global em $F(m) = 0$, sob essas condicões.

Sendo assim, tem-se que o problema de fatoração de inteiros foi transformado em um problema de otimização combinatória de minimização com variáveis binárias, tal que seu modelo é dado por:

$$\begin{gathered} \min \{F\} \\ p_i, q_i, z_{i,j} \in \{0,1\}, \forall\ i,j \end{gathered} \tag{3.6}$$

onde $F = \sum_{i=0}^{n_c-1} C_i^2$ e todas as soluções são viáveis (ou seja, irrestrito).

Pode-se então perceber a semelhança com problemas de satisfatibilidade (SAT).

Por outro lado, embora seja, de certa forma, um problema de satisfatibilidade, ele difere dos problemas SAT padrão, uma vez que equações SAT são definidas em um campo finito binário (e expressas usualmente na forma normal conjuntiva ou disjuntiva), enquanto no caso em foco as equações são sobre os inteiros [3].

Definido então o problema de fatoração de inteiros como um problema de otimização binária, pode-se passar para o próximo passo, que consiste em tentar simplificar o sistema de cláusulas obtido.

3.1.2 Simplificação de cláusulas

Com o objetivo de simplificar o sistema de cláusulas obtido no passo anterior, pode-se aplicar relações (regras) de modo a resolver diretamente para um subconjunto das variáveis que tenham fácil resolução clássica, conforme discutido por Dattani et al. [37]. Esse conjunto de regras será denominado, no contexto deste trabalho, de esquema de pré-processamento clássico. Um exemplo é apresentado e utilizado por Anschuetz et al. [2], conforme descrito a seguir:

$$\begin{aligned} xy - 1 = 0 &\Longrightarrow x = y = 1, \\ x + y - 1 = 0 &\Longrightarrow xy = 0, \\ a - bx = 0 &\Longrightarrow x = 1, \\ \sum_i x_i = 0 &\Longrightarrow x_i = 0, \forall\ i \\ \sum_{i=1}^{a} x_i - a = 0 &\Longrightarrow x_i = 1, \forall\ i. \end{aligned} \tag{3.7}$$

onde $x, y \in \{0, 1\}$ são variáveis binárias e $a, b \in \mathbb{Z}^+$ são constantes inteiras positivas.

Neste trabalho, foi utilizado o esquema de pré-processamento clássico descrito pelas Eqs. (3.7) em conjunto com as regras adicionais apresentadas pelas Eqs. (4.1), na seção 4.1 [1]. Nesse contexto, o esquema adotado é aplicado de modo iterativo em cada cláusula C_i definida na Eq. (3.4), um número constante de vezes.

Como último detalhe na simplificação do sistema de claúsulas, existe a possibilidade de truncar o somatório do último termo na Eq. 3.4 [2]. Para este fim, pode-se notar que se o termo 2^j, presente no somatório, for maior que o máximo resultado possível da soma dos outros três termos, para uma mesma cláusula, ou seja, se:

$$2^j > \max\left\{\sum_{j=0}^{i} q_j p_{i-j} + \sum_{j=0}^{i} z_{j,i} - m_i\right\} \tag{3.8}$$

[1] Ressalta-se que existem ainda outros esquemas de pré-processamento clássico, conforme apresentado por Karamlou et al. [4].

para cada i, tal que $0 \leq i < n_c$, então o termo $z_{i,i+j}$ que o acompanha não poderá ser 1. Caso contrário, o lado direito da equação em Eq. 3.2 seria sempre negativo, para todas as atribuições possíveis das demais variáveis, de modo que a equação nunca seria satisfeita. Sendo assim, $\forall\ i, j$, tal que a condição expressa pela Eq. (3.8) é satisfeita, faz-se $z_{i,i+j} = 0$, limitando-se assim o número de elementos do somatório presente no último termo. Dessa forma, limita-se a magnitude da Eq. (3.4), para cada i, a $O(n_m)$ [2].

Em relação à complexidade computacional deste procedimento, primeiramente deve-se atentar para a identidade $n_c = n_p + n_q - 1$, onde $n_p \leq n_m$ e $n_q \leq \lceil \frac{N_m}{2} \rceil$ [3]. Dessa forma, tem-se que $n_c \in O(n_m)$. Lembra-se ainda que o procedimento de pré-processamento clássico é aplicado em cada uma das n_c cláusulas um número constante de vezes, logo em $O(n_m)$ cláusulas.

Por outro lado, ao analisar cada cláusula dada pela Eq. (3.4), percebe-se que há $O(n_c)$ termos. Isso pode ser percebido devido ao fato de, em cada cláusula, o último somatório apresentar $n_c - 1$ termos e ainda haver o termo relativo ao biprimo m, m_i, e, no pior caso, ou seja para $i = n_c - 1$ cada um dos somatórios restantes apresentar $n_c - 1$ termos. Sendo assim, o procedimento passa por $O(n_m)$ cláusulas, cada uma contendo $O(n_m)$ termos, resultando em uma complexidade de $O(n_m^2)$ [2].

Por fim, denomina-se C_i' a cláusula C_i após a aplicação do procedimento de simplificação descrito. Tendo sido simplificado o sistema de cláusulas, resta agora mapeá-lo para um Hamiltoniano de Ising que tem codificado no seu estado fundamental as soluções das variáveis remanescentes.

3.1.3 Codificação em um Hamiltoniano de Ising

O Modelo de Ising, desenvolvido por Ernest Ising, foi inicialmente desenvolvido para o estudo de transição de fase de diferentes materiais ferromagnéticos [38]. Os blocos construtores do Modelo de Ising são partículas quânticas binárias de dois níveis, assim como os elétrons com spin $\pm 1/2$. O modelo é então formado por grupos destes elétrons (ou outras partículas) em uma configuração específica em que os spins (níveis) de elétrons vizinhos (adjacentes) afetam uns aos outros em uma maneira específica [39].

Nesse contexto, esse modelo pode ser representado matematicamente por um Hamiltoniano (Hamiltoniano de Ising), que consiste em um observável físico que quantifica a energia total de um sistema. Uma história da evolução do Modelo de Ising pode ser analisada em diversos trabalhos [40, 41].

Sendo assim, foi visto anteriormente que o problema de fatoração de inteiros foi transformado em um problema de otimização combinatória de minimização com variáveis binárias. Foi mostrado ainda, em trabalhos anteriores [2, 4, 20, 26, 37], que

esse novo problema pode ser codificado no estado fundamental de um Hamiltoniano de Ising.

Para perceber esse fato, primeiramente deve-se notar que a função $F = \sum_{i=0}^{n_c-1} C_i'^2$ é a função de energia clássica [2], tal que:

$$E = \sum_{i=0}^{n_c-1} C_i'^2 \tag{3.9}$$

onde C_i', $\forall\, i$, são as cláusulas após o processo de simplificação descrito na seção 3.1.2.

Dessa forma, as soluções das equações do sistema simplificado $C_i' = 0$ correpondem a minimizar a função de energia clássica dada pela Eq. (3.9).

Pode-se agora mapear cada cláusula simplificada C_i' para um termo $\hat{C}_i$ em um Hamiltoniano de Ising H_I [4]. Para este fim, utiliza-se a seguinte transformacão:

$$b_k \mapsto \frac{1}{2}\left(1 - Z_k\right) \tag{3.10}$$

onde $b_k \in \{p_i, q_i, z_{i,j}\}$ e Z_k é o operador de Pauli Z aplicado no k-ésimo qubit. Logo, os autovalores ± 1 de Z correspondem aos spins positivo e negativo dos elétrons do modelo (níveis das partículas) [42]. Ou seja, cada bit é mapeado para um operador qubit e pode-se afirmar que a função clássica de energia foi quantizada.

Portanto, tem-se que o problema de fatoração foi mapeado para o problema de encontrar o estado fundamental (de energia mínima) de um Hamiltoniano de Ising dado por:

$$H_I = \sum_{i=0}^{n_c-1} \hat{C}_i^2 \tag{3.11}$$

que possui um um estado fundamental de energia $E_0 = 0$ bem definido [4].

Por fim, Cao et al. [38] demonstraram a viabilidade de modelar o Hamiltoniano de Ising em computadores quânticos da IBM, que é o que foi realizado na seção 4.3.

3.2 Algoritmo de otimização quântica aproximada (QAOA)

Conforme dito no início do capítulo, o algoritmo VQF consiste em uma abordagem híbrida clássico/quântica, baseada em um pré-processamento clássico seguido

da aplicação do algoritmo QAOA. Este, por sua vez, é um algoritmo híbrido clássico/quântico para ambiente NISQ que resolve, aproximadamente, problemas clássicos de otimização [23]. Finalizada então a descrição do pré-processamento clássico na seção anterior, deve-se agora entrar na parte referente ao QAOA.

Sendo assim, o QAOA consiste no principal componente do esquema VQF [2], sendo um resolvedor de estado fundamental quântico aproximado para o Hamiltoniano descrito na Eq. (3.11). Com esse objetivo em mente, deve-se então definir a função de custo e a *ansatz* a serem utilizadas, assim como discutido na seção 2.3.

Para definir a função de custo a ser utilizada no algoritmo, primeiramente deve-se lembrar da equação de Schrödinger, que descreve a evolução no tempo de uma função de onda que é a caracterização, na mecânica quântica, de um sistema físico isolado [43, 44]. Na sua forma básica (independente do tempo) é dada por:

$$H|\psi\rangle = E|\psi\rangle \tag{3.12}$$

onde H é o Hamiltoniano que representa a quantidade física de interesse (no escopo deste trabalho, a energia do sistema quântico), $|\psi\rangle$ é um estado específico do sistema quântico (função de onda) e E é o valor da energia no determinado estado $|\psi\rangle$.

Percebe-se, portanto, que a função de onda neste caso é um autovetor do Hamiltoniano H, $|\psi\rangle$, de modo que a energia E correspondente ao estado representado é dado pelo seu autovalor associado.

Sendo assim, pode-se manipular a equação Eq. (3.12) de modo a isolar a energia:

$$H|\psi\rangle = E|\psi\rangle \Rightarrow \langle\psi|H|\psi\rangle = E \Rightarrow E = \langle\psi|H|\psi\rangle \tag{3.13}$$

Conclui-se, a partir da Eq. (3.13), que a energia correspondente a determinado estado $|\psi\rangle$ é igual ao valor esperado do Hamiltoniano, que representa a energia do sistema, no respectivo estado.

Sendo assim, o objetivo agora é encontrar qual desses estados corresponde ao menor valor de energia para o sistema, ou seja o seu estado fundamental. Com esse objetivo em foco, faz-se uso do método variacional [45], que consiste em um método aproximado para encontrar estimativas aproximadas da energia do estado fundamental. Essa técnica é baseada no Princípio Variacional (Teorema Variacional) [46]:

Seja $|\psi\rangle$ uma solução experimental (teste) adequadamente normalizada para a Eq. (3.12), onde H é um hamiltoniano conhecido independente do tempo. O Princípio Variacional então afirma que a energia do estado fundamental, E_0, é sempre

menor ou igual ao valor esperado de H calculado com a solução experimental:

$$E_0 \leq \langle\psi|H|\psi\rangle \tag{3.14}$$

Dessa forma, por meio da variação de $|\psi\rangle$ até que o valor esperado de H seja minimizado, é possível obter aproximações para a solução e para a energia do estado fundamental.

Finalmente, pode-se assim definir a funcão de custo a ser minimizada pelo QAOA modificando a Eq. (3.13) para ser parametrizada:

$$E(\theta) = \langle\psi(\theta)|H|\psi(\theta)\rangle \tag{3.15}$$

Pode-se agora seguir para a definição da ansatz a ser utilizada no QAOA. Foi visto anteriormente que o objetivo é encontrar o estado fundamental do Hamiltoniano que representa a energia do sistema, denominado Hamiltoniano de Custo no QAOA (H_C). Entretanto, este costuma ser complexo e o referido cálculo extremamente complicado ou até inviável.

Nesse contexto, introduz-se um novo hamiltoniano H_a, denominado hamiltoniano admixing (mixing ou mixer), H_a, cujo estado fundamental é conhecido e é eficientemente calculado. Farhi et al. [23] fazem então uso do Teorema Adiabático Quântico [47]: como o hamiltoniano H_a introduzido anteriormente é escolhido de modo a satisfazer os requisitos de Perron-Frobenius [48, 49], pode-se evoluir o sistema, que está inicialmente no autovetor de menor autovalor de H_a (estado de menor energia, ou seja fundamental), sobre a influência de um hamiltoniano que interpola lentamente de H_a para H_c por um longo período de tempo T, de modo que o estado final do sistema é o de autovetor de menor autovalor do hamiltoniano H_C, conforme desejado [50]. Seja então a interpolação linear:

$$H(t) = \frac{t}{T}H_C + \left(1 - \frac{t}{T}\right)H_a \tag{3.16}$$

onde $t \in [0, T]$.

Entretanto, tem-se que a interpolação anterior é realizada em um intervalo contínuo, sendo necessário discretizá-lo para permitir a sua simulação. Com esse objetivo, considera-se novamente a equação de Schrödinger, porém agora dependente do tempo:

$$|\psi(t)\rangle = e^{-iHt}|\psi(0)\rangle \tag{3.17}$$

onde H, no caso em foco, é o Hamiltoniano que representa a energia do sistema quântico, $|\psi(0)\rangle$ é o estado inicial do sistema (no tempo $t = 0$) e $|\psi(t)\rangle$ é o estado em que estará o sistema em um determinado tempo t.

Sendo assim, realiza-se uma Trotterização [51] na evolução temporal em s passos temporais, seguida de uma nova Trotterização em cada termo resultante, de modo a culminar na evolução de estado [50]:

$$|\vec{\beta}, \vec{\gamma}\rangle = e^{-i\beta_s H_a} e^{-i\gamma_s H_C} \dots e^{-i\beta_1 H_a} e^{-i\gamma_1 H_C} |\psi(0)\rangle \tag{3.18}$$

onde $|\psi(0)\rangle$ corresponde ao estado fundamental de H_a e as $2s$ variáveis $\vec{\beta} = (\beta_1, \dots \beta_s)$ e $\vec{\gamma} = (\gamma_1, \dots, \gamma_s)$ são parâmetros de otimização cujos valores ótimos correspondem a um caminho de evolução que replica o de $H(t)$ (Eq. (3.16)) o mais próximo que o espaço de parâmetros permite.

Pode-se agora, finalmente, definir especificamente a função de custo e a ansatz utilizadas para o QAOA. Os autores [23] introduzem o hamiltoniano admixing $H_a = \sum_0^{n-1} X_i$, tal que o seu estado fundamental é dado por uma superposição uniforme sobre os estados base computacionais:

$$|\psi(0)\rangle = \frac{1}{\sqrt{2^n}} \sum_z |z\rangle \tag{3.19}$$

onde i indica o qubit em que o operador de Pauli X está atuando e n o número de qubits do sistema.

Sendo assim, a ansatz (estado de teste parametrizado) utilizado é dado por:

$$|\vec{\beta}, \vec{\gamma}\rangle = \prod_{i=1}^{s} (e^{-i\beta_i H_a} e^{-i\gamma_i H_C}) |+\rangle^{\otimes n} \tag{3.20}$$

onde $|+\rangle^{\otimes n}$ é a superposição uniforme sobre os estados base computacionais.

Já a função de custo parametrizada a ser minimizada é obtida a partir da Eq. (3.15), atualizando o estado parametrizado pela ansatz do QAOA com os respectivos parâmetros e o hamiltoniano H pelo hamiltoniano de custo do problema, H_C, tal que $H_C = H_I$, que é o Hamiltoniano de Ising dado pela Eq. (3.11):

$$E(\vec{\beta},\vec{\gamma}) = \langle \vec{\beta},\vec{\gamma}|H_C|\vec{\beta},\vec{\gamma}\rangle \tag{3.21}$$

Sendo assim, o QAOA utiliza um algoritmo clássico de otimização para otimizar os parâmetros β e γ, para um número fixo s de camadas, e assim minimizar a função de custo.

Ressalta-se ainda que, para $s \to \infty$, $E(\vec{\beta},\vec{\gamma})$ atinge o valor mínimo quando a fidelidade entre $|\vec{\beta},\vec{\gamma}\rangle$ e o verdadeiro estado fundamental de H_C é 1 [2].

Pode-se ainda discutir a função de cada operador na Eq. (3.20). Tem-se os dois operadores:

$$U_H(\gamma) = e^{-i\gamma H_C} \tag{3.22}$$

$$U_a(\beta) = e^{-i\beta \sum_0^{n-1} X_i} = \prod_0^{n-1} e^{-i\beta X_i} \tag{3.23}$$

onde $U_H(\gamma)$ é conhecido como operador separador de fase, sendo responsável por aplicar uma fase de emaranhamento de acordo com o hamiltoniano de custo H_C. Ou seja, aplica uma mudança relativa de fase entre os estados, introduzindo viés de acordo com a sua qualidade (estados com menor energia têm melhor qualidade).

Já $U_a(\beta)$ é conhecido como operador *transverse-field* mixer (mixing ou admixing), sendo responsável por gerar interferência entre os estados com o objetivo de aumentar a amplitude das soluções (estados) de alta-qualidade, ou seja aumentar a probabilidade das soluções de melhor qualidade serem medidas [52].

Por último, cabe mencionar que Hadfield et al. [24] apresentaram uma generalização do QAOA original, denominada *Quantum Alternating Operator Ansatz* (mantendo a mesma sigla), de modo a permitir a alternância entre famílias mais gerais de operadores. A motivação consistiu na importância demonstrada pelos autores do preparo de operadores e de estados iniciais específicos para cada problema.

3.2.1 Inicialização de parâmetros

Como passo final do VQF e, como parte do QAOA, se faz necessário fornecer um bom ponto inicial para o otimizador clássico a ser utilizado, de modo que este possa otimizar os parâmetros do circuito.

O método escolhido pelos autores é o de grid search (busca em grade), que consiste em uma busca por força bruta, camada por camada, em uma grade formada a partir de números igualmente espaçados em um intervalo de busca, prédefinido para cada parâmetro, definidos por um tamanho de grade escolhido.

Sendo assim, para otimizar os parâmetros β e γ, inicialmente realiza-se essa busca em cada par de parâmetros (γ_i, β_i), onde $1 \leq i \leq s$. Faz-se necessário então definir o tamanho de grade a ser utilizado e o intervalo de busca.

O tamanho de grade introduzido é dado por $O(n_c^2 n^4)$ (resolução final de grade resulta em $O(n_c^2 n^4)$ x $O(n_c^2 n^4)$), onde n é o número de qubits (variáveis, ou seja, bits remanescentes após o pré-processamento clássico). Isso garante que o ótimo encontrado pela grid search difere do ótimo verdadeiro por uma constante e garante ainda uma escala polinomial da grade de busca [2].

Como o Hamiltoniano de Ising do problema (H_I) e o transverse-field mixer (H_a), conforme definido nas seções 3.1.3 e 3.2, possuem apenas autovalores inteiros, os intervalos $\gamma \in [0, 2\pi]^s$ e $\beta \in [0, \pi]^s$ garantem que a busca produzirá um resultado eficiente [23]. O resultado dessa busca é então fornecido ao algoritmo de otimização global escolhido, BFGS [53]. Dessa forma, para s camadas, tem-se que são realizadas sg^2 avaliações do circuito quântico parametrizado, onde g é o tamanho de grade escolhido, e, consequentemente, da função objetivo. Portanto, esse passo tem uma complexidade total de $O(sn_c^4 n^8)$.

Por fim, tem-se que o restante da complexidade do algoritmo é dado pelo procedimento de otimização dos parâmetros realizado pelo algoritmo de otimização clássico. Os resultados numéricos do artigo original [2], assim como os obtidos na seção 4.2.1 e indicados na Fig. 4.6 deste trabalho, sugerem que a escala de complexidade da execução do algoritmo clássico de otimização até a convergência, seja para um mínimo local ou global, é aproximadamente independente do tamanho da instância do problema e dependente linearmente do número de camadas do circuito QAOA.

Sendo assim, para um circuito QAOA de s camadas, tem-se que o custo total de execução do VQF é dado por $O(s^2 n_c^4 n^8)$. Entretanto, é ressaltado pelos autores que, numericamente, esse limite parece solto, uma vez que tamanhos de grade significativamente menores (produzem resoluções de grade mais grossas ou espaçadas) são suficientes para o algoritmo. Ressalta-se ainda que não há garantia de que o algoritmo sempre irá gerar a solução global ótima.

3.2.2 Taxa de sucesso

Após a execução do algoritmo de otimização clássico, conforme descrito na seção 3.2.1, tem-se que os parâmetros foram otimizados, tendo sido encontrados valores ótimos locais ou globais. Com esse resultado em mãos, pode-se finalmente preparar o circuito parametrizado final $|\vec{\gamma}_{opt}, \vec{\beta_{opt}}\rangle$ no computador quântico e medi-lo.

Realiza-se então a amostragem do circuito preparado e verifica-se se o resultado obtido (string de bits medida) satisfaz o sistema de cláusulas $C_i' = 0$. Em caso positivo, pode-se então mapear os bits medidos para as variáveis $\{p_i\}$, $\{q_i\}$ e $\{z_{ij}\}$

remanescentes após o pré-processamento clássico, procedimento este denominado de pós-processamento clássico.

Dessa forma, o sucesso do VQF é medido pela probabilidade da string medida codificar os fatores corretos. Entretanto, pode-se observar um fato interessante: ainda que o objetivo seja que a bitstring medida satisfaça todas as cláusulas, na realidade basta que codifique corretamente as variáveis remanescentes $\{p_i\}$ e $\{q_i\}$, correspondentes aos fatores p e q. Ou seja, ainda que os valores medidos correspondentes às variáveis $\{z_{ij}\}$ (carry-bits) estejam incorretos, se os valores medidos correspondentes às variáveis $\{p_i\}$ e $\{q_i\}$ estiverem corretos, os fatores p e q continuarão sendo medidos corretamente.[2]

Dessa forma, define-se então o conjunto $M_s = \{b_j\}$ formado por todas as bitstrings amostradas do computador quântico que codificam as variáveis remanescentes $\{p_i\}$ e $\{q_i\}$ corretamente. Em seguida, define-se a taxa de sucesso, $s(\vec{\gamma}, \vec{\beta})$, como a proporção de bitstrings amostradas do computador quântico com o estado final parametrizado (circuito) $\mid \vec{\gamma}_{opt}, \vec{\beta_{opt}}\rangle$ que codificam corretamente as variáveis $\{p_i\}$ e $\{q_i\}$ em $C_i' = 0$:

$$\begin{aligned} s(\vec{\gamma}, \vec{\beta}) &= \frac{|M_s|}{|M|}, \\ |M_s| &= \{b_j \in M \mid p_i = p_i^f, q_i = q_i^f, \forall\ p_i \in P', \forall\ q_i \in Q'\} \end{aligned} \tag{3.24}$$

onde $|M_s|$ é o número de amostras que codificam as variáveis $\{p_i\}$ e $\{q_i\}$ corretamente, $|M|$ é o número total de medições amostradas, $P_f = \{p_i^f\}$ e $Q_f = \{q_i^f\}$ correspondem aos valores corretos, nos fatores originais em forma binária, nas respectivas posições i, e $0 \leq s(\vec{\gamma}, \vec{\beta}) \leq 1$. Tem-se ainda que $P' = \{p_i\}$ e $Q' = \{q_i\}$ são os conjuntos das variáveis binárias dos fatores p e q, respectivamente, remanescentes após o processo clássico de simplificação de cláusulas.

Sendo assim, bastaria que uma única bitstring, dentre todas as amostras, atendesse aos critérios descritos acima para que os fatores corretos fossem obtidos. Entretanto, quanto menor a frequência de bitstrings satisfatórias, menor é a taxa de sucesso e, portanto maior é o número necessário de preparos e medições do estado de teste $\mid \vec{\gamma}_{opt}, \vec{\beta_{opt}}\rangle$.

Nesse contexto, como se trata de um processo estocástico e, portanto não determinístico, o objetivo é obter valores maiores para a taxa de sucesso, preferencialmente $s(\vec{\gamma}, \vec{\beta}) = 1$, o que indicaria que os fatores estão codificados corretamente em todas as bitstrings amostradas. Cabe ressaltar que a taxa de sucesso se aproxima de 1 na medida que a função objetivo definida pela Eq. (3.21) se aproxima de 0.

[2]Tal fato ocorre, pois, na implementação, as variáveis correspondentes aos fatores $\{p_i\}$ e $\{q_i\}$, ao serem eliminadas, nunca são colocadas em função das variáveis $\{z_{ij}\}$. Apenas o inverso ocorre.

Ou seja:

$$\begin{aligned} E(\vec{\beta},\vec{\gamma}) = 0 \Rightarrow s(\vec{\gamma},\vec{\beta}) = 1, \\ 0 \leq s(\vec{\gamma},\vec{\beta}) \leq 1,\ E(\vec{\beta},\vec{\gamma}) \geq 0 \end{aligned} \tag{3.25}$$

3.3 Algoritmo de otimização assistido por caminhada quântica (QWOA)

De modo a seguir a metodologia proposta para o trabalho de testar variações para cada etapa, procurou-se testar outro algoritmo variacional de otimização e assim comparar os resultados obtidos pela variação com o original e possivelmente gerar novos benchmarks para o algoritmo.

Nesse contexto, escolheu-se testar o *Quantum walk-assisted optimization algorithm* (QWOA) [27]. O QWOA é considerado uma generalização do QAOA, sendo portanto uma escolha natural, baseada na alternância entre caminhadas quânticas de tempo-contínuo (c*ontinuous-time quantum walks* - CTQW) [54] sobre soluções viáveis que podem ser identificadas de modo eficiente, e mudanças de fase dependentes da qualidade das soluções.

A caminhada quântica de tempo-contínuo, de modo similar às caminhadas aleatórias (*random walks*), descreve, em um grafo, o movimento de um caminhante sobre um conjunto de caminhos predeterminados. Entretanto, ao invés de jogar uma moeda a fim de decidir qual caminho percorrer em cada ponto, o caminhante quântico pode pegar vários caminhos ao mesmo tempo [55]. Sendo assim, enquanto a *random walk* é determinística, no sentido de que, em cada ponto, sabe-se exatamente onde está, a *quantum walk* é não-determinística, podendo estar em vários pontos ao mesmo tempo, devido à superposição quântica.

Segue-se agora para definir a ansatz e a função de custo utilizadas no QWOA. Ao contrário do QAOA que, conforme visto anteriormente, utiliza como arcabouço a evolução adiabática trotterizada, o QWOA utiliza a CTQW como ansatz [27], de modo que a evolução do sistema quântico é dado pela alternância destas com as mudanças de fase dependentes da qualidade das soluções:

$$|\vec{\gamma},\vec{t}\rangle = U_W(t_s)\, U_{H_C}(\gamma_s) \ldots U_W(t_1)\, U_{H_C}(\gamma_1)\, |\psi\rangle \tag{3.26}$$

onde $|\psi\rangle$ é o estado inicial definido por uma superposição uniforme entre todas as soluções combinatórias viáveis, $U_{H_C}(\gamma)$ é o operador mudança de fase assim como definido para o QAOA na seção 3.2, $U_W(t)$ é o operador quantum-walk mixer que

aplica uma CTQW por um tempo t sobre o domínio combinatório e, por fim, os $2s$ parâmetros $\vec{t} = (t_1, \ldots t_s)$ e $\vec{\gamma} = (\gamma_1, \ldots, \gamma_s)$ são utilizados para a otimização da função de custo, assim como também no QAOA.

Dessa forma, o operador $U_{H_C}(\gamma)$ aplica uma mudança de fase em cada solução de acordo com a sua qualidade, enquanto o operador $U_W(t)$ realiza uma quantum-walk no grafo, misturando as amplitudes entre as soluções. Sendo assim, a alternância de aplicação entre os dois operadores provoca um aumento da amplitude dependente da qualidade das soluções, ou seja aumenta a probabilidade de medição de soluções de melhor qualidade (no escopo deste trabalho, de menor energia associada) [5].

Cabe agora definir matematicamente o operador $U_W(t)$, uma vez que o operador $U_{H_C}(\gamma)$ já está definido pela Eq. (3.22) [5]:

$$U_W(t) = e^{-it\mathcal{L}} \tag{3.27}$$

onde $t \geq 0$ pode ser entendido como o tempo de duração da quantum-walk, conforme já dito, e $\mathcal{L}$ é a matriz Laplaciana do grafo circulante que conecta todas as soluções viáveis do problema.

A escolha de um grafo circulante [56] (como exemplo o grafo completo ou um ciclo) se deve à preservação de simetria e ao fato de todo grafo circulante ser diagonalizado pela Transformada de Fourier. Dessa forma, a Transformada de Fourier Quântica [57] (*Quantum Fourier Tansform* - QFT) pode ser utilizada para diagonalizar qualquer escolha de grafo circulante, enquanto a estrutura do circuito permanece, em grande parte, inalterada [27].

Nesse contexto, Bennet et al. [5] propõem uma estrutura altamente eficiente para executar a simulação acelerada de quantum-walk em grafos completos conectando todas as soluções viáveis. A escolha do grafo completo $\mathbb{K}_M$, onde M é a cardinalidade do conjunto de soluções viáveis, se deve ao fato de já ter sido mostrado que a QFT módulo-arbitrário consiste em uma abordagem eficiente para a realização de quantum walk nesse grafo [27].

Entretanto, é conhecido que a QFT é sensível a ruído o que, para o escopo NISQ em que este trabalho se foca, é altamente indesejável. Dessa forma, na estrutura proposta pelos autores, o Laplaciano $\mathcal{L}$ presente na Eq. (3.27), pode ser expresso da seguinte forma:

$$\mathcal{L} = M(\mathbb{I} - |\psi\rangle\langle\psi|) \tag{3.28}$$

onde M é a cardinalidade do conjunto de soluções viáveis, conforme já dito, $\mathbb{I}$ é a

matriz Identidade e $|\psi\rangle$ é a superposição uniforme entre as soluções viáveis.

De posse da expressão Eq. (3.28), pode-se reescrever o operador quantum walk, até uma fase global, na forma:

$$U_W(t) = e^{-it\mathcal{L}} = \mathbb{I} + \left(-1 + e^{iMt}\right)|\psi\rangle\langle\psi| \tag{3.29}$$

Tem-se ainda que a Eq. (3.29) expressa a rotação do sistema em torno de um estado específico $|\psi\rangle$ [58].

Outro detalhe que ainda não foi especificado é o estado inicial $|\psi\rangle$. Como o problema abordado de fatoração como otimização binária constitui um problema irrestrito, conforme discutido na seção 3.1.1, tem-se que o estado inicial é dado pela superposição uniforme de todas as soluções do espaço solução, uma vez que todas as soluções são viáveis. Sendo assim, tem-se que o estado $|\psi\rangle$ é dado por:

$$|\psi\rangle = \frac{1}{\sqrt{M}} \sum_{x \in \mathbb{S}} |x\rangle \tag{3.30}$$

onde $M = |\mathbb{S}| = 2^n$, $\mathbb{S}$ é o conjunto das soluções e n é o número de qubits do sistema quântico.

Pode-se agora então, finalmente, reescrever a equação (3.26) com as novas expressões detalhadas:

$$|\vec{\gamma}, \vec{t}\rangle = \prod_{i=1}^{s} \{[\mathbb{I} + \left(-1 + e^{iMt}\right)|+\rangle^{\otimes n}\langle+|^{\otimes n}]e^{-i\gamma_i H_C}\}|+\rangle^{\otimes n} \tag{3.31}$$

onde $|+\rangle^{\otimes n}$ é a superposição uniforme sobre os estados base computacionais.

Em continuação, procura-se agora definir a função de custo, conforme desejado. Nesse caso, é análoga à do QAOA, em que procura-se minimizar o valor esperado para o Hamiltoniano que representa a energia do sistema, substituindo apenas a ansatz (estado de teste parametrizado) do QAOA pelo que acabou-se de definir para o QWOA:

$$E(\vec{\gamma}, \vec{t}) = \langle \vec{\gamma}, \vec{t}|H_C|\vec{\gamma}, \vec{t}\rangle \tag{3.32}$$

Bennet et al. [5] apresentam ainda os pré-requisitos, sendo compostos por quatro critérios, para que um problema seja um bom candidato para o QWOA:

1. A cardinalidade do espaço de soluções viáveis (únicas) M deve ser eficiente-

mente computável, dada uma instância específica do problema:
No caso em foco, todas as soluções são viáveis (problema irrestrito), logo M é dado diretamente pelo espaço de Hillbert para o número n de qubits do sistema: $M = 2^n$.

2. Deve existir um algoritmo eficiente de indexação e desindexação do espaço solução:
Não se aplica ao caso em foco, uma vez que todo o espaço solução é composto de soluções viáveis, que já estão codificadas em strings de n-bits, em ordem lexicográfica. Exemplos de aplicação do processo de indexação e desindexação podem ser vistos nos trabalhos [27, 29, 30, 50], para diferentes tipos de problemas e objetos combinatórios.

3. A qualidade de qualquer solução arbitrária deve ser eficientemente computável:
Foi visto anteriormente na seção 3.2 que a qualidade de uma determinada solução (estado) é o autovalor associado ao correspondente autovetor. Sendo assim, como cada autovetor corresponde a um estado da base computacional, o seu autovalor pode ser obtido de forma eficiente a partir da multiplicação do Hamiltoniano de Ising relativo ao sistema pelo respectivo autovetor. Isso corresponde a multiplicar uma matriz (n x n), onde n é o número de qubits, por um dos seus respectivos autovetores (dimensão n), que é conhecido, o que resulta no autovetor multiplicado pelo respectivo autovalor. Nesse ponto, basta apenas verificar qual termo é diferente de zero. Essa operação pode ser realizada em $O(n^2)$.

4. O problema deve estar na classe NPO-PB, ou seja, a classe dos problemas de otimização em que a função objetivo é limitada polinomialmente no tamanho da instância do problema [50]:
Pode-se lembrar, da Eq. (3.9), que a função objetivo é obtida a partir da soma dos quadrados de cada clásula (que podem ser reduzidas pelo esquema de pré-processamento clássico). Foi visto ainda, na seção 3.1.2, que a magnitude da Eq. (3.4) é limitada a $O(n_m)$. Como os termos do somatório são quadráticos e tem-se $O(n_m)$ cláusulas no somatório, pode-se concluir que a função objetivo $F \in O(n_m^3)$, sendo polinomial no tamanho da entrada m.

Conclui-se portanto que a fatoração como problema de otimização combinatória binária atende aos pré-requisitos. Entretanto, uma grande vantagem do QWOA consiste na redução do espaço de busca, por meio da preparação de estado, ou seja na capacidade de restringir o espaço solução a apenas as soluções viáveis (objetos combinatórios corretos), ganho esse que não é observado no caso em foco uma vez que é irrestrito, conforme já mencionado.

Sendo assim, a manutenção dos testes com essa variação é baseada no fato de se obter uma possível vantagem devido ao quantum-walk mixer, cujo arcabouço em grafo completo impede o viés inicial entre possíveis soluções. Dessa forma, um possível espalhamento mais rápido para soluções de melhor qualidade foi investigado. De fato, para determinadas instâncias do problema, esse comportamento foi observado, conforme descrito na seção 4.2.5.

Do exposto nessa seção, percebe-se, para o caso de fatoração estudado neste trabalho, as similaridades entre o QAOA e o QWOA, sendo a diferença entre as variações sendo dada pelos operadores mixers aplicados (transverse-field mixer para o QAOA e quantum-walk mixer para o QWOA).

Em relação à inicialização dos parâmetros γ e t, foi utilizada a técnica de inicialização descrita na seção 3.2.1. Foi mantida ainda, para fins de comparação, a resolução de grade $O(n_c^2 n^4)$ x $O(n_c^2 n^4)$, entretanto uma análise análiticа deve ser realizada a fim de verificar se é um bom limite superior. Por outro lado, como o quantum-walk mixer não possui necessariamente valores inteiros, o intervalo dos parâmetros t não podem ser restringidos a $[0, \pi]^s$, enquanto o intervalo dos parâmetros γ podem ser mantidos em $[0, 2\pi]^s$, visto que o hamiltoniano de custo H_C é o mesmo nos dois casos [50]. Sendo assim, foi utilizado o intervalo $[0, 2\pi]^s$ para os parâmetros t, o que se mostrou eficiente pelos resultados das simulações, conforme expresso na seção 4.2.5.

Outro fato que cabe ser mencionado, é o de que, assim como no caso do QAOA, conforme discutido na seção 3.2.1, os resultados númericos expressos na seção 4.2.5 sugerem que a escala de complexidade da execução do algoritmo clássico de otimização até a convergência, seja para um mínimo local ou global, é aproximadamente independente do tamanho da instância do problema e dependente linearmente do número de camadas do circuito QWOA. Sendo assim, para um circuito QWOA de s camadas, tem-se que o custo total de execução do VQF, nesta variação, é dado por $O(s^2 n_c^4 n^8)$. Por fim, ressalta-se que as ressalvas realizadas na seção 3.2.1 em relação à solução global ótima são mantidas.

Capítulo 4

Análise Experimental

4.1 Pré-processamento clássico

O objetivo desse experimento era verificar a eficiência do pré-processamento clássico em simplificar o sistema de cláusulas por meio da redução no número de cláusulas e de variáveis. Como consequência, verifica-se a eficiência na redução do número de qubits necessários para construir o Hamiltoniano de Ising que representa o estado de energia do sistema e no qual o resultado da otimização está codificado.

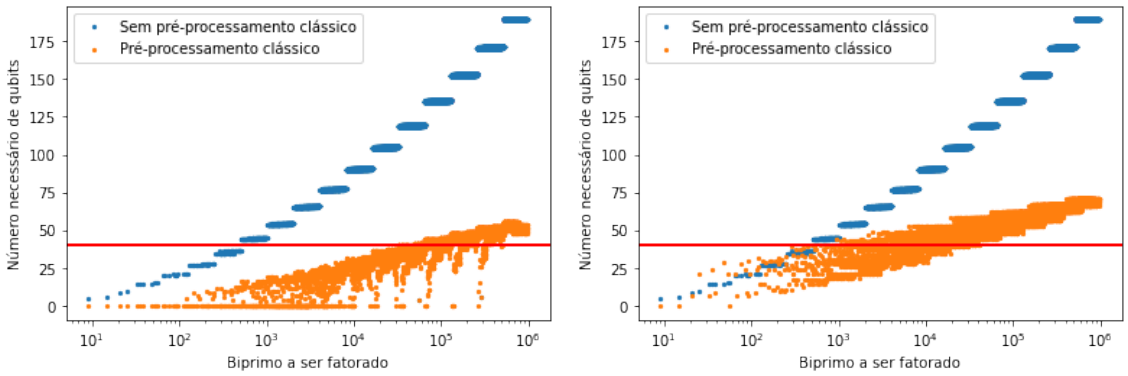
Nesse contexto, foram utilizadas as regras de pré-processamento descritas na seção 3.1.2 em conjunto com as seguintes regras de paridade, tal que $x, y, z \in \{0, 1\}$ são variáveis binárias e $a, b, c, d \in \mathbb{Z}^+$ são constantes inteiras positivas:

$$\begin{gathered} x + y - 2z = 0 \Rightarrow x = y = z \\ x + 2y - 2z = 0 \Rightarrow x = 0 \ \& \ y = z \\ x - 2z + 1 = 0 \Rightarrow x = 1 \ \& \ z = 1 \\ a + b + 2c + 2d - 2x - 4y = 0 \Rightarrow a = b \end{gathered} \tag{4.1}$$

Ressalta-se novamente que existem outras regras que podem ser aplicadas.[1] Sendo assim, com o objetivo de analisar o comportamento de escala do número de qubits necessário para o algoritmo, foi aplicado o processo de pré-processamento clássico em todos os números biprimos até 10^6, obtendo os resultados conforme apresentado nas Figs. 4.1, 4.2 e 4.3.

Pode-se então perceber, na Fig. 4.1, que o esquema de pré-processamento clássico é extremamente eficiente na tarefa de redução do número de qubits, inclusive quando os tamanhos dos fatores p e q não são previamente desconhecidos, cenário esse que condiz com quase a totalidade dos casos encontrados na prática. Segue dessa

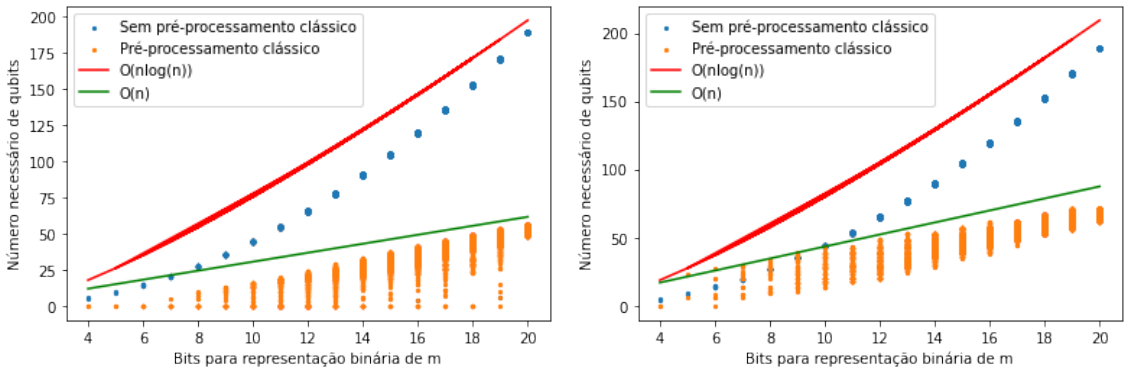
[1] Para as instâncias $m \in \{56153, 291311\}$ testadas neste trabalho, foi utilizado o esquema de pré-processamento apresentado em [37].



(a) Tamanhos dos fatores p e q conhecidos

(b) Sem conhecimento dos tamanhos dos fatores p e q

Figura 4.1: Resultado empírico da redução no número de qubits necessário para o algoritmo após a realização do pré-processamento clássico. O salto discreto observado no caso sem o pré-processamento clássico (azul) ocorre devido ao fato de números de mesma grandeza serem representados com o mesmo número de bits. Já o comportamento irregular do caso com pré-processamento clássico (laranja) ocorre devido ao efeito de eliminação das variáveis do sistema de equações; assim como observado por Anschuetz et al. [2]. (a) Foi considerado que os tamanhos dos fatores p e q eram previamente conhecidos a fim de reduzir o número de qubits necessário. (b) Resultado sem conhecimento prévio dos tamanhos de p e q, condizente com o cenário real encontrado.

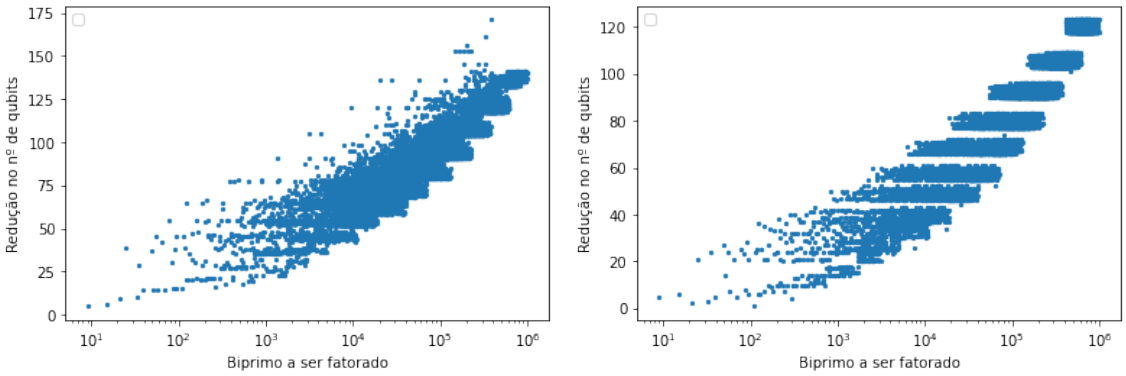


(a) Tamanhos dos fatores p e q conhecidos

(b) Sem conhecimento dos tamanhos dos fatores p e q

Figura 4.2: Observa-se que, sem a aplicação de pré-processamento clássico (azul), o número de qubits necessário para o algoritmo escala de forma empírica, aproximadamente, como $O(nlog(n))$ (vermelho). Já com a aplicação de pré-processamento clássico (laranja), este escala aproximadamente como $O(n)$ (verde); conforme previsto por Borges [3] e observado por Anschuetz et al. [2] e Karamlou et al. [4]. Onde n é o número de bits necessários para representar o biprimo m na sua forma binária $(n = n_m)$.

observação que melhorias nas regras de pré-processamento implicam diretamente na redução do número de qubits necessário para a constução do Hamiltoniano de



(a) Tamanhos dos fatores p e q conhecidos

(b) Sem conhecimento dos tamanhos dos fatores p e q

Figura 4.3: Número de qubits reduzidos pela aplicação do esquema de préprocessamento clássico em relação ao caso sem préprocessamento.

Ising e, consequentemente para a construção do circuito quântico parametrizado do QAOA e QWOA.

Por outro lado, pode-se observar, na Fig. 4.2, a escala empírica do número de qubits em relação ao número de bits necessário para a representação binária do biprimo m. Percebe-se, portanto, que o esquema de pré-processamento ocasiona uma escala linear do número de qubits.

Por fim, observa-se, na Fig. 4.3, a escala do número de qubits reduzidos ao aplicar-se o esquema de préprocessamento clássico, ou seja a diferença entre o número necessário de qubits sem o préprocessamento clássico e o número necessário de qubits após a aplicação das regras.

4.2 Simulação ideal

Como ponto de partida das simulações numéricas de otimização, optou-se por realizar sinulações sob condições ideais, ou seja, sem a presença de ruído que pudesse prejudicar os resultados encontrados. Nesse contexto, o objetivo era analisar o comportamento dos algoritmos na prática e comparar os resultados com o previsto na literatura.[2] Dessa forma, foram escolhidas as instâncias descritas na tabela 4.2.[3]

Ainda neste contexto, faz-se necessário estabelecer o conceito de simetria $p \leftrightarrow q$ considerado neste trabalho. Seja $P' = \{p_i\}$, onde $0 \leq i \leq n_p - 1$, o conjunto formado pelos bits remanescentes do fator p original, na sua forma binária, após

[2]O trabalho realizado por Michał Stęchły, presente em https://github.com/mstechly/vqf, foi utilizado como inspiração inicial para a implementação realizada nesta dissertação.

[3]Com o objetivo de reduzir o número de qubits necessário para construir o Hamiltoniano de Ising e, consequentemente para executar o algoritmo VQF, devido à limitações de hardware, foi assumido que os tamanhos dos fatores p e q eram previamente conhecidos. Essa premissa não interfere nos experimentos e seus resultados.

a aplicação do esquema de pré-processamento apresentado. Seja ainda $P^c = \{p_i^c\}$, onde $0 \leq i \leq n_p - 1$, o conjunto formado pelos bits do fator p original, na sua forma binária, já calculados por meio do esquema de pré-processamento apresentado. Analogamente define-se os conjuntos Q' e Q^c.

Afirma-se então que existe simetria $p \leftrightarrow q$ entre os dois fatores se e somente se os conjuntos P' e Q' e os conjuntos P^c e Q^c têm a mesma cardinalidade, os elementos da primeira dupla, na mesma posição, são o complementar um do outro, e os elementos da segunda dupla, na mesma posição, são iguais, ou seja, deve-se ter:

$$\begin{aligned} |P'| &= |Q'|,\ p_i = \overline{q_i} \\ |P^c| &= |Q^c|,\ p_i^c = q_i^c \end{aligned} \tag{4.2}$$

Na prática, pode-se verificar esse comportamento na instância $m = 56153$, por exemplo, que foi utilizada nas simulações. Após a aplicação do esquema de pré-processamento clássico, obtém-se o seguinte sistema de cláusulas:

$$\begin{aligned} p_3 + q_3 - 1 &= 0 \\ p_4 + q_4 - 1 &= 0 \\ p_4 q_3 + p_3 q_4 - 1 &= 0 \end{aligned} \tag{4.3}$$

Neste momento, os fatores, em suas respectivas formas binárias, estão representados de acordo com a tabela 4.1.

Fatores	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
p	1	1	1	p_4	p_3	0	0	1
q	1	1	1	q_4	q_3	0	0	1

Tabela 4.1: Representação binária dos fatores p e q da instância $m = 56153$ após a aplicação do esquema de pré-processamento clássico.

De modo a encontrar os fatores corretos que formam a instância, isto é, 233 e 241, há as seguintes possibilidades para os bits p_i e q_i:

$$p_3 = 0, p_4 = 1, q_3 = 1, q_4 = 0 \tag{4.4}$$

$$p_3 = 1, p_4 = 0, q_3 = 0, q_4 = 1 \tag{4.5}$$

Percebe-se, portanto, que as duas possibilidades expressas pelas Eqs. (4.4) e (4.5) e os bits já calculados para os fatores, expressos de acordo com a Tabela 4.1, atendem ao deerminado pela Eq. (4.2), concluindo-se assim que os fatores 233 e 241

apresentam simetria $p \leftrightarrow q$.

A consequência deste fato consiste em que ambas as equações implicam no cálculo correto dos fatores, apenas invertendo-os. Ou seja, no primeiro caso temos $p = 241$ e $q = 233$ e no segundo caso temos $p = 233$ e $q = 241$.

Já os tamanhos de grade utilizados seguem os limites definidos na seção 3.2.1, embora, como já dito anteriormente, resultados numéricos tenham indicado que tamanhos de grade menores são suficientes [2]. Fato este que também foi observado neste trabalho.

Finalmente, após a definição das instâncias, pode-se caminhar para as simulações de fato.

Biprimo "m"	**Número de qubits**	**Número de "carry-bits"**	**Simetria p ↔ q**	**Tamanho de grade**
1981 = 7 x 283	2	0	Não	6 x 6
319 = 11 x 29	6	3	Não	24 x 24
69169 = 263 x 263	8	5	Não	36 x 36
2363 = 17 x 139	3	1	Não	9 x 9
56153 = 233 x 241	4	0	Sim	12 x 12
291311 = 523 x 557	6	0	Sim	24 x 24

Tabela 4.2: Tabela descrevendo as instâncias utilizadas nas simulações ideais. Primeira coluna: Indica os números biprimos utilizados na simulação ideal. Segunda coluna: Expressa o número total de qubits necessário para executar o algoritmo VQF após a aplicação das regras de pré-processamento descritas pelas Eqs. (3.7) e (4.1). Terceira coluna: Representa, dentro do número total de qubits necessário, o número de carry-bits. Quarta coluna: Indica se existe simetria entre os fatores p e q após a aplicação das regras de pré-processamento clássico. Quinta coluna: Tamanho de grade utilizado na inicialização de parâmetros por grid search.

4.2.1 Otimização via QAOA

Nesta simulação, realizou-se o teste do algoritmo QAOA, descrito na seção 3.2.[4] Utilizou-se o método de grid search para a inicialização dos parâmetros a serem fornecidos ao algoritmo clássico de otimização, o algoritmo de otimização clássico BFGS e as instâncias detalhadas na tabela 4.2.[5]

No estudo, foram analisados os comportamentos da taxa de sucesso do algoritmo, do valor da função objetivo e do número de avaliações da função objetivo pelo algoritmo de otimização clássico até a convergência, de acordo com o número de camadas do QAOA.

[4]Para a realização das simulações em ambiente ideal, utilizou-se o framework *Python Qiskit* da IBM, com o backend *Aer Simulator*. Já os otimizadores clássicos são implementados via classe *minimize*, presente na biblioteca *Python open-source SciPy*.

[5]Foi considerada uma tolerância de $tol = 10^{-5}$ para o algoritmo de otimização clássico e o número de amostras $v = 1000$. As barras verticais de erro nos gráficos indicam, cada uma, um desvio padrão em três instâncias do problema.

Nesse contexto, a Fig. 4.4 representa a taxa de sucesso de encontrar os fatores corretos p e q em função do número de camadas s do QAOA. Pode-se perceber que, para $m = 1981, 2363, 56153$ a taxa de sucesso cresce com o aumento do número de camadas, conforme esperado, convergindo ainda para o valor máximo. Isso significa que os fatores corretos estavam codificados em todas as bitstrings amostradas.

Já para as instâncias $m = 319, 69169, 291311$ a taxa de sucesso não escala tão bem, ocorrendo um aumento marginal. Uma análise inicial sugere que essa discrepância de resultados apresentados pelos dois grupos de instâncias está relacionado à diferença no número de qubits necessário para executá-las no VQF, assim como à presença ou não de carry-bits e simetria $p \leftrightarrow q$, hipótese essa levantada pelos autores originais [2]. Uma análise numérica mais profunda acerca desses fatores é realizada na seção 4.2.4.

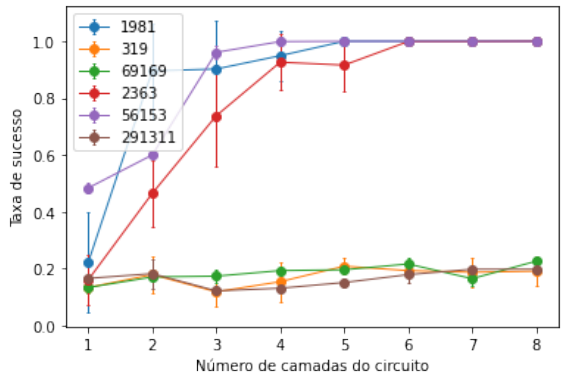


Figura 4.4: A taxa de sucesso do algoritmo VQF para cada instância do problema.

Outra análise importante, como dito anteriormente, foi do comportamento do valor da função objetivo em relação ao número de camadas do algoritmo, conforme representado na Fig. 4.5. Percebe-se que, para as instâncias $m = 1981, 2363, 56153$, o valor converge para zero, atingindo o valor mínimo. Já para as demais instâncias, o valor da função objetivo é minimizado, porém de modo menos eficiente, indicando que o algoritmo de otimização clássica ficou preso em um mínimo local ($m = 291311, 69169$) ou que é necessário um número maior de camadas até a convergência para zero ($m = 319$). No caso da instância $m = 319$, resultados numéricos mostraram que o valor da função objetivo chegou próximo de zero e o valor da taxa de sucesso atingiu 0,9 com $s = 30$ camadas do QAOA, tendo atingido o valor máximo em $s = 59$ camadas. Já para as outras duas instâncias, um aumento do número de camadas não correspondeu a uma melhora no valor da função objetivo e, consequentemente, da taxa de sucesso. Uma análise da superfície de custo percorrida pelo algoritmo de otimização clássica se faz necessária para tentar compreender melhor o comportamento do segundo grupo de instâncias. Entretanto, cabe ressaltar que o comportamento do valor da função objetivo condiz com o previsto e com o

observado no comportamento da taxa de sucesso, uma vez que, conforme discutido na seção 3.2.2, a taxa de sucesso se aproxima de 1 (valor máximo) na medida em que o valor da função objetivo se aproxima de 0 (Eq. (3.25)).

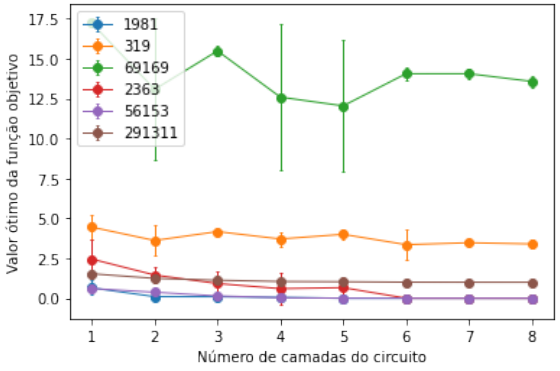
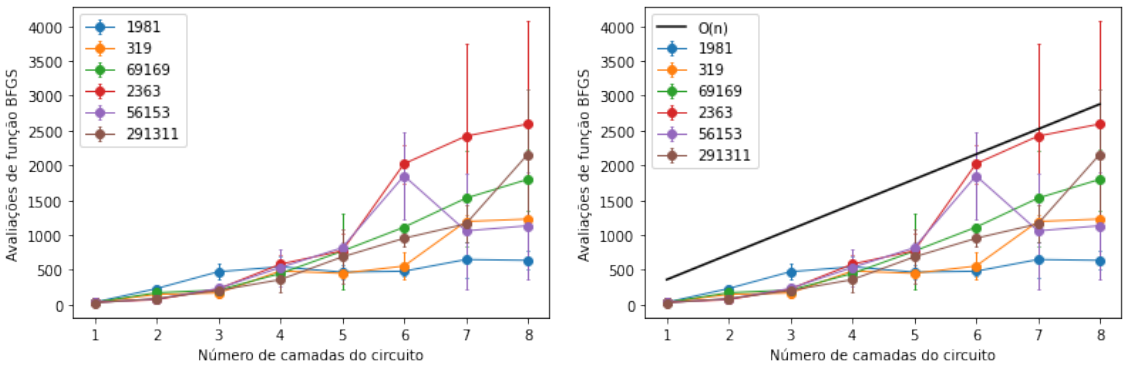


Figura 4.5: O valor da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) para cada instância do problema.

Por fim, analisou-se o número de avaliações da função objetivo realizadas pelo algoritmo de otimização clássica BFGS, sinalizando a exigência da parte clássica de otimização do VQF, resultado esse apresentado na Fig. 4.6. Os resultados indicam que a escala do número de avaliações é aproximadamente linear no número de camadas e aproximadamente independente do tamanho do problema.



(a) (b)

Figura 4.6: (a) Número de avaliações da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) até o algoritmo BFGS convergir. (b) Aproximadamente linear no número de camadas n.

4.2.2 Otimizadores clássicos

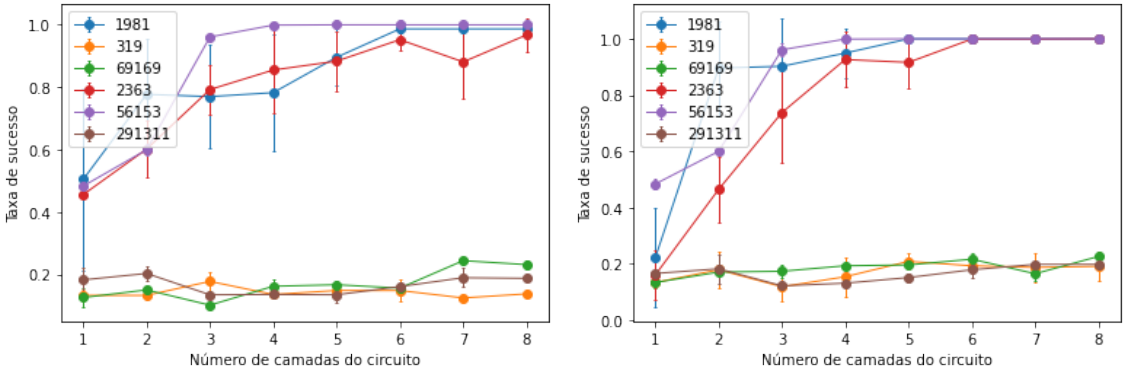
Neste estudo, buscou-se comparar o algoritmo de otimização clássico BFGS com a sua variação, o algoritmo clássico L-BFGS-B.[6]

[6]Foi considerada uma tolerância de $tol = 10^{-5}$ para o algoritmo de otimização clássico e o número de amostras $v = 1000$. As barras verticais de erro nos gráficos indicam, cada uma, um

Este algoritmo deriva do algoritmo L-BFGS (*Limited-memory* BFGS) que, por sua vez, deriva do algoritmo BFGS já citado anteriormente. O L-BFGS, assim como o seu antecessor, é um método quasi-Newton voltado para otimizações não-lineares de larga escala. Entretanto, sua vantagem consiste no fato de aproximar o BFGS utilizando uma quantidade limitada de memória computacional [59].

Nesse contexto, o algoritmo L-BFGS-B vai além, permitindo a manipulação de restrições simples da forma $l_i \leq x_i \leq u_i$, ou seja, de limites superiores e inferiores para cada variável [60]. Dessa forma, torna-se possível aplicar, na otimização clássica dos parâmetros, as restrições $\gamma \in [0, 2\pi]^s$ e $\beta \in [0, \pi]^s$, apresentadas na seção 3.2.1.

Sendo assim, repetiu-se as simulações realizadas na seção 4.2.1, substituindo o algoritmo clássico de otimização utilizado de BFGS para o L-BFGS-B. Inicialmente na Fig. 4.7, pode-se observar que o comportamento da taxa de sucesso foi mantido, com pequenas variações. Para as instâncias $m = 1981, 2363, 56153$, a taxa de sucesso permaneceu tendo uma boa escala com o aumento do número de camadas, tendo as instâncias $m = 1981, 2363$ convergido para o valor máximo mais tardiamente. Já as instâncias $m = 319, 69169, 291311$ apresentaram taxas de sucesso cujos valores se mantiveram, aproximadamente, em torno dos mesmos valores apresentados para o caso BFGS.



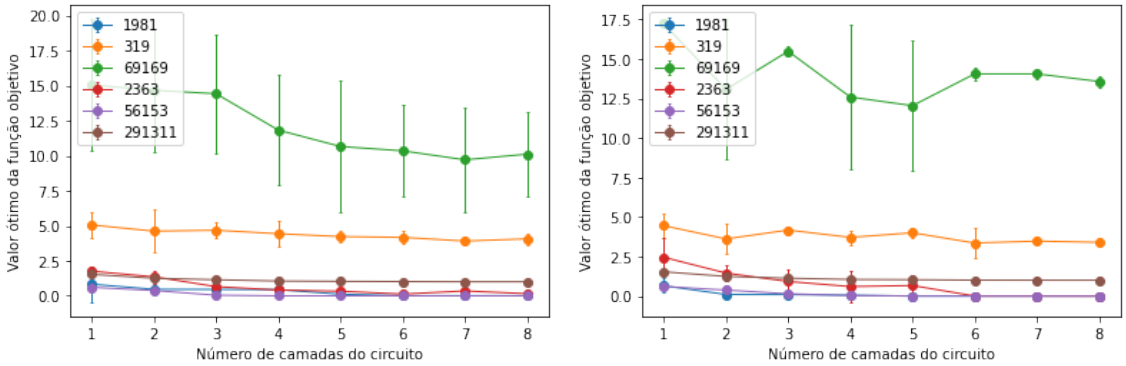
(a) (b)

Figura 4.7: (a) A taxa de sucesso do algoritmo VQF para cada instância do problema, utilizando o algoritmo de otimização clássica L-BFGS-B. (b) A taxa de sucesso do algoritmo VQF para cada instância do problema, utilizando o algoritmo de otimização clássica BFGS.

Já na Fig.4.8, verifica-se a comparação entre os valores das funções objetivo para os dois casos. Para as instâncias $m = 1981, 2363, 56153$, o valor da função objetivo convergiu para o valor mínimo em um número maior de camadas. Já para as instâncias $m = 319, 69169, 291311$, seus valores foram reduzidos, porém em menor

desvio padrão em três instâncias do problema. Foi utilizado, em ambos os casos, a inicialização de parâmetros pelo método de grid search.

escala, tendo se mantido em patamares semelhantes ao observado no caso do BFGS. Pode-se observar que, novamente, o valor da função objetivo se comportou como o esperado, sendo inversamente proporcional à taxa de sucesso.



(a) (b)

Figura 4.8: (a) O valor da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) para cada instância do problema, utilizando o algoritmo de otimização clássica L-BFGS-B. (b) O valor da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) para cada instância do problema, utilizando o algoritmo de otimização clássica BFGS.

Finalmente, na Fig. 4.9, pode-se comparar o número de avaliações da função objetivo, realizado pelo algoritmo de otimização clássico, para os dois casos. Percebe-se, portanto, que a linearidade em relação ao número de camadas permanece. Entretanto, uma discrepância percebida consiste no fato de que o algoritmo L-BFGS-B realiza um número menor de avaliações da função objetivo até a convergência, comportamento esse observado em todas as instâncias testadas, o que resulta em um menor custo computacional clássico. Uma hipótese é que esse comportamento é devido ao caráter de aproximação do L-BFGS-B em relação ao BFGS, porém uma análise mais profunda deve ser realizada para a sua compreensão.

Dessa forma, do apresentado nesta seção, pode-se concluir que os resultados observados condizem com o esperado, visto que, conforme dito anteriormente, o L-BFGS-B busca aproximar o BFGS.

4.2.3 Inicialização de parâmetros

Durante a realização das simulações numéricas, observou-se um fato bem interessante. Ao inicializar os parâmetros iniciais, a serem fornecidos para o algoritmo de otimização clássico, todos com o valor 1, os resultados obtidos foram extremamente similares e, em alguns casos, até melhores do que ao utilizar o método de grid search. Esse tipo de inicialização de parâmetros será, a partir de agora, denominada inicialização unitária de vetor.

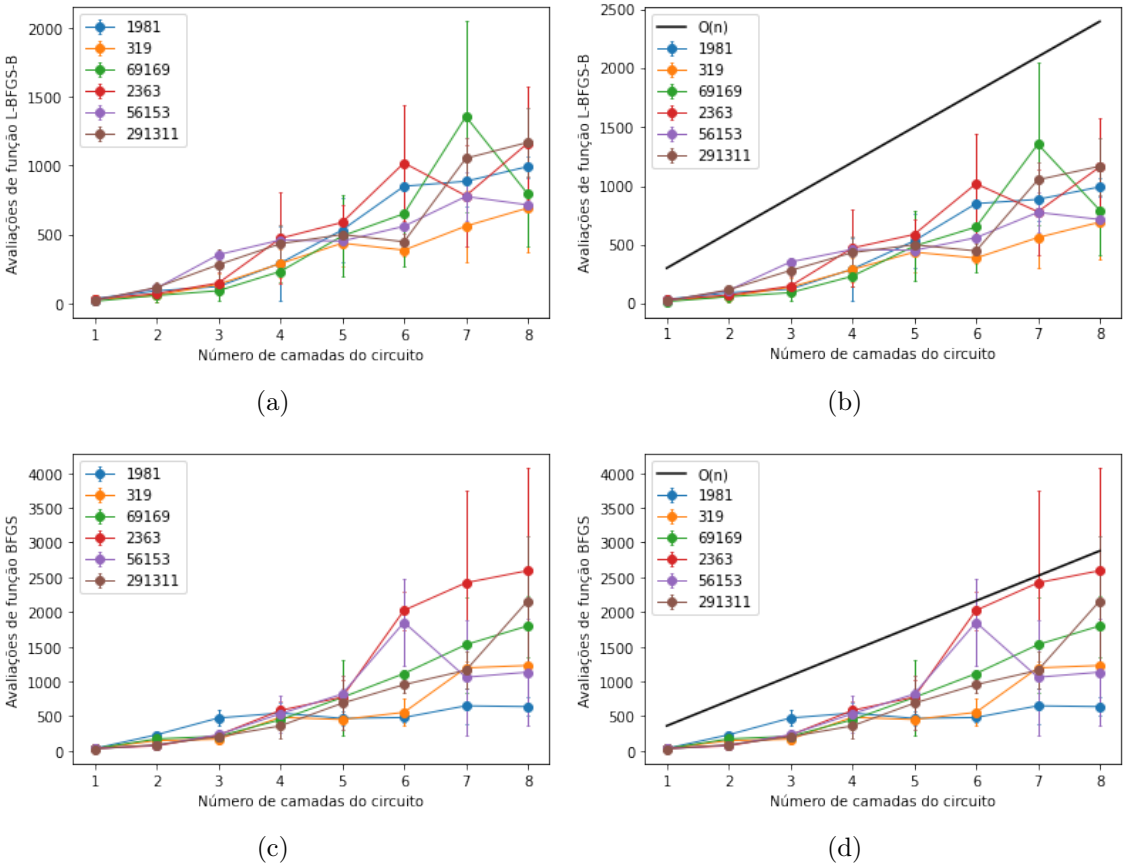


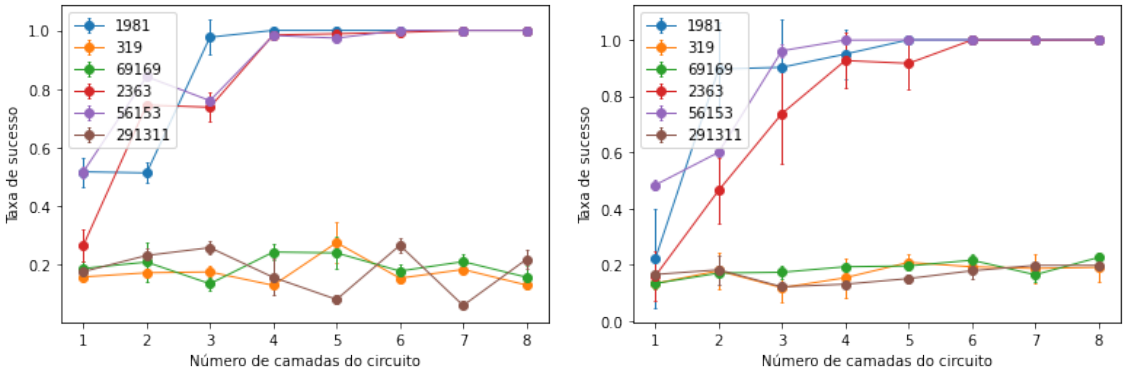
Figura 4.9: (a) Número de avaliações da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) até o algoritmo L-BFGS-B convergir. (c) Número de avaliações da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) até o algoritmo BFGS convergir. (b) e (d) Aproximadamente linear no número de camadas n.

Dessa forma, repetiu-se as simulações realizadas na seção 4.2.1, porém substituindo a inicialização de parâmetros por grid search pela inicialização de parâmetros por inicialização unitária de vetor.[7]

Nesse contexto, pode-se observar, na Fig. 4.10, que o comportamento da taxa de sucesso para as duas simulações é extremamente similar. No caso das instâncias $m = 1981, 2363, 56153$, a inicialização unitária de vetor faz com que a conversão da taxa de sucesso para o valor máximo seja mais rápida. Já para as instâncias $m = 319, 69169, 291311$, essa apresenta um comportamento mais errático, embora tenha se mantido, aproximadamente, nos mesmos valores.

Já na Fig. 4.11, pode-se perceber novamente o mesmo comportamento para o valor da função objetivo, sendo, nos dois casos, inversamente proporcional à taxa de sucesso, conforme esperado. Os valores atingidos nos dois casos tam-

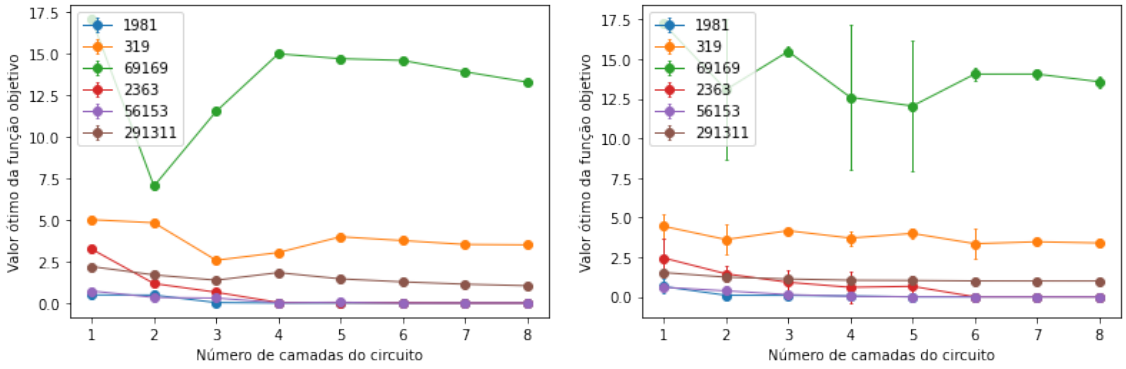
[7]Foi considerada uma tolerância de $tol = 10^{-5}$ para o algoritmo de otimização clássico e o número de amostras $v = 1000$. As barras verticais de erro nos gráficos indicam, cada uma, um desvio padrão em três instâncias do problema. Em ambas as simulações foi utilizado o algoritmo clássico de otimização BFGS.



(a) (b)

Figura 4.10: (a) A taxa de sucesso do algoritmo VQF para cada instância do problema, utilizando a inicialização de parâmetros por inicialização unitária de vetor. (b) A taxa de sucesso do algoritmo VQF para cada instância do problema, utilizando a inicialização de parâmetros por grid search.

bém foi aproximadamente igual, tendo atingido o valor mínimo, para as instâncias $m = 1981, 2363, 56153$, mais rapidamente no caso da inicialização unitária de vetor.



(a) (b)

Figura 4.11: (a) O valor da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) para cada instância do problema, utilizando a inicialização de parâmetros por inicialização unitária de vetor. (b) O valor da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) para cada instância do problema, utilizando a inicialização de parâmetros por grid search.

Finalmente, para o caso do número de avaliações da função objetivo, realizadas pelo algoritmo de otimização clássico, também foi observado o mesmo comportamento aproximadamente linear relativo ao número de camadas e independente do tamanho da instância do problema, conforme expresso na Fig. 4.12. Entretanto, foi percebido que o algoritmo de otimização clássico realiza um número menor de avaliações da função objetivo (seja por ter sido bem sucedida ou por algum critério de parada) no caso da inicialização de parâmetros por inicialização unitária de vetor.

Cabe ressaltar que os resultados observados acima foram mantidos ao substi-

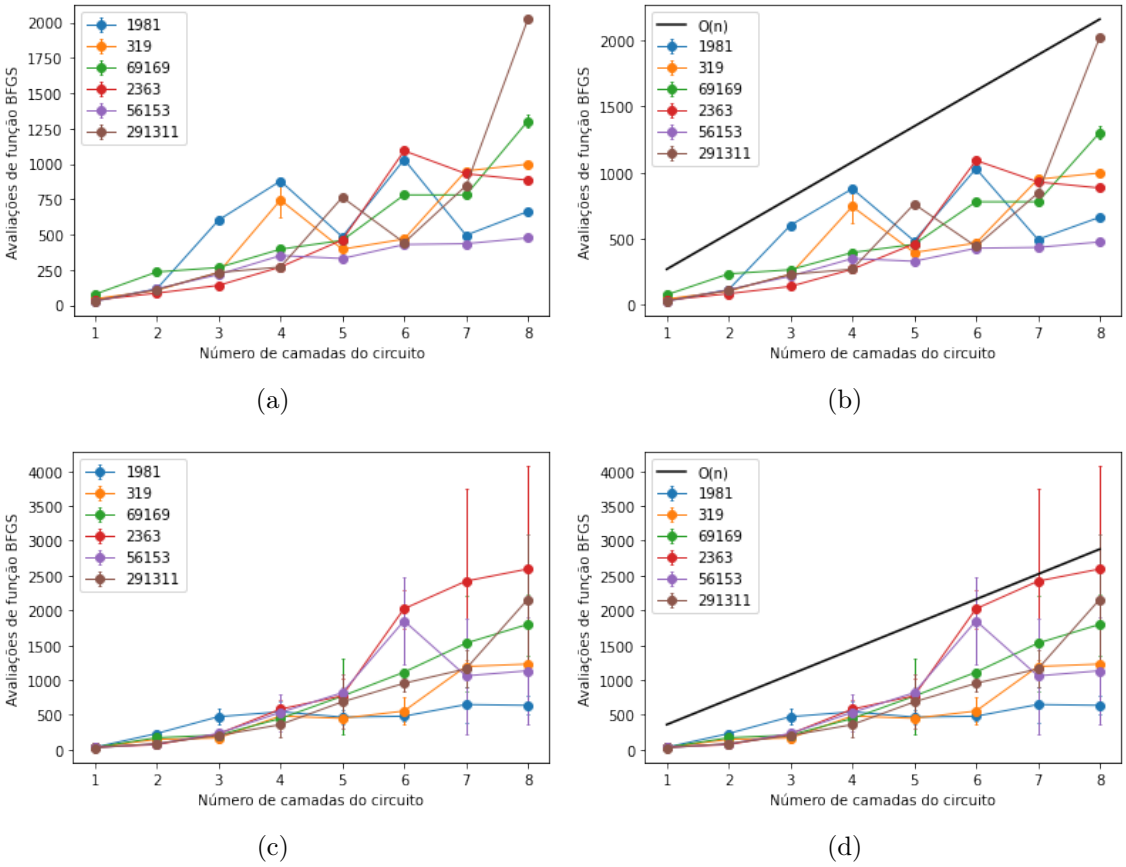


Figura 4.12: (a) Número de avaliações da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) até o algoritmo BFGS convergir, utilizando a inicialização de parâmetros por inicialização unitária de vetor. (c) Número de avaliações da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) até o algoritmo BFGS convergir, utilizando a inicialização de parâmetros por grid search. (b) e (c) Aproximadamente linear no número de camadas n.

tuir o algoritmo de otimização clássica BFGS pelo algoritmo L-BFGS-B, conforme esperado.

Para entender melhor esse comportamento, analisou-se a superfície de custo percorrida pelo algoritmo de otimização clássica L-BFGS-B em instâncias do algoritmo VQF de tamanho 3, 4 e 5 qubits, conforme detalhado por Karamlou et al. [4]. Foi observado que as superfícies de custo, para $s = (2, ..., 8)$, seguem um mesmo padrão, tanto entre camadas, como entre instâncias do problema. Essa estrutura é observada, inclusive, para circuitos de grande profundidade.

Dessa forma, analisou-se o comportamento do algoritmo, neste trabalho, de acordo com as áreas da superfície de custo apresentada na Fig. 4.13. De fato, inicializando os parâmetros com os valores da área azul, ou seja, de menor valor de energia (valor da função objetivo) provocou resultados da otimização clássica com menores valores para a função objetivo e maiores valores para a taxa de sucesso. O oposto foi observado ao inicializar os parâmetros com valores das áreas verdes e amarelas, conforme esperado.

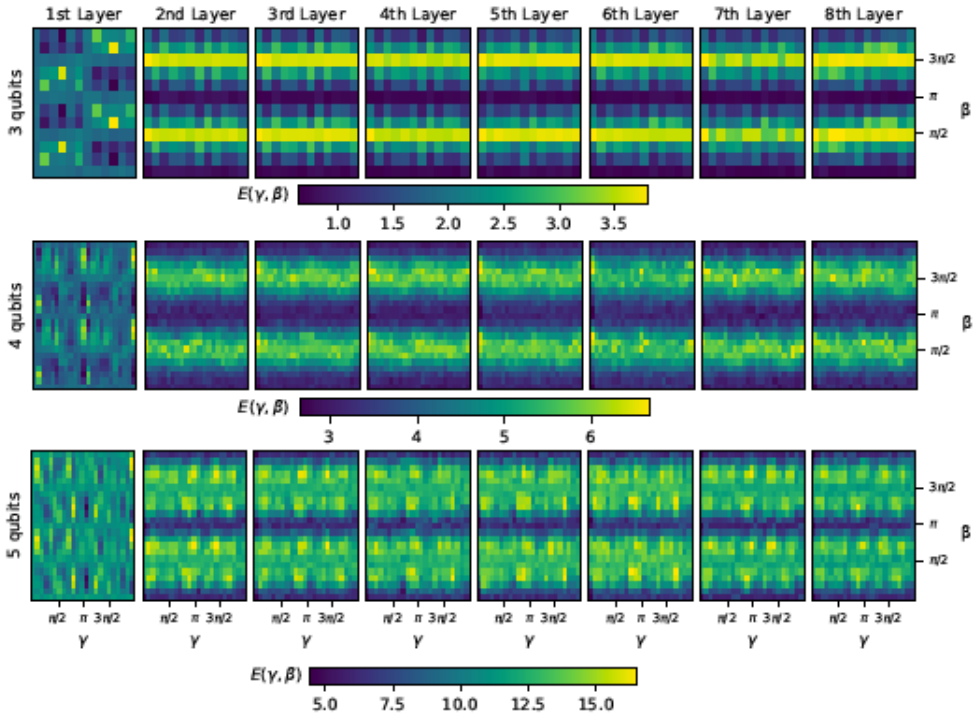


Figura 4.13: Superfície de custo para a otimização clássica do QAOA. Para cada instância do problema (linhas), é explicitado o valor da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)), em função do número de camadas do QAOA (colunas), e de acordo com as grades definidas pelos valores de $\vec{\gamma}$ e $\vec{\beta}$. - Figura obtida de Karamlou et al. [4].

Sendo assim, faz todo o sentido que o método de inicialização unitária de vetor tenha funcionado, uma vez que está em uma área de bons valores para a função objetivo, na superfície de custo, ou seja, mais próximo de um mínimo local ou

global. O mesmo comportamento foi observado para valores próximos de zero e próximos de π, para γ e β, condizente novamente com o explicitado pelas superfícies de custo.

Tal fato se torna importante, uma vez que, de posse do conhecimento da estrutura perene (entre camadas e entre instâncias) da superfície de custo, pode-se evitar a realização de $N = sg^2$ avaliações do circuito quântico parametrizado $|\gamma, \beta\rangle$, por meio de bons palpites para o ponto inicial dos parâmetos, reduzindo assim o custo total de execução do VQF. Esse resultado pode contribuir para a construção de heurísticas mais eficientes de inicialização de parâmetros.

Entretanto, o método de grid search se provou muito eficiente para produzir um ponto inicial adequado para o algoritmo de otimização clássica, o que, a priori, sem conhecimento algum do comportamento da função objetivo do problema, se torna essencial. Ressalta-se ainda que existem outros métodos de inicialização de parâmetros que podem ser testados e utilizados, como o INTERP e FOURIER [61].

4.2.4 Relações numéricas

O objetivo do estudo realizado nesta seção foi analisar possíveis relações numéricas entre os biprimos m e seus fatores p e q que pudessem implicar em comportamentos específicos, seja no número de qubits necessário para construir o Hamiltoniano de Ising após a aplicação das regras de pré-processamento clássico, seja diretamente na taxa de sucesso do algoritmo VQF como um todo. Buscou-se ainda estudar mais profundamente as hipóteses levantadas na seção 4.2.1 em relação às taxas de sucesso apresentadas pelos dois grupos de instâncias. Para este fim, utilizou-se o método de inicialização de parâmetros por inicialização unitária de vetor, o algoritmo de otimização clássico L-BFGS-B e as instâncias detalhadas na tabela 4.3.[8]

Inicialmente, procurou-se identificar alguma possível relação entre os biprimos e os seus fatores que pudesse influenciar no número de qubits necessário para a construção do Hamiltoniano de Ising. Nesse contexto, as simulações realizadas indicaram apenas que o número de qubits depende exclusivamente do esquema de pré-processamento clássico utilizado, conforme descrito na seção 4.1 Fica explicitado que quanto mais completas e abrangentes forem as regras de pré-processamento clássico utilizadas, maior será a redução no número de qubits necessário. Deve-se ressaltar, no entanto, que o esquema de pré-processamento deve ser eficiente, uma vez que ocasiona um maior custo computacional clássico. Logo, o trade-off entre recursos quânticos e clássicos deve valer a pena.

Em um segundo momento, procurou-se investigar os fatores influenciadores da

[8]Foi considerada uma tolerância de $tol = 10^{-5}$ para o algoritmo de otimização clássico e o número de amostras $v = 1000$. As barras verticais de erro nos gráficos indicam, cada uma, um desvio padrão em três instâncias do problema.

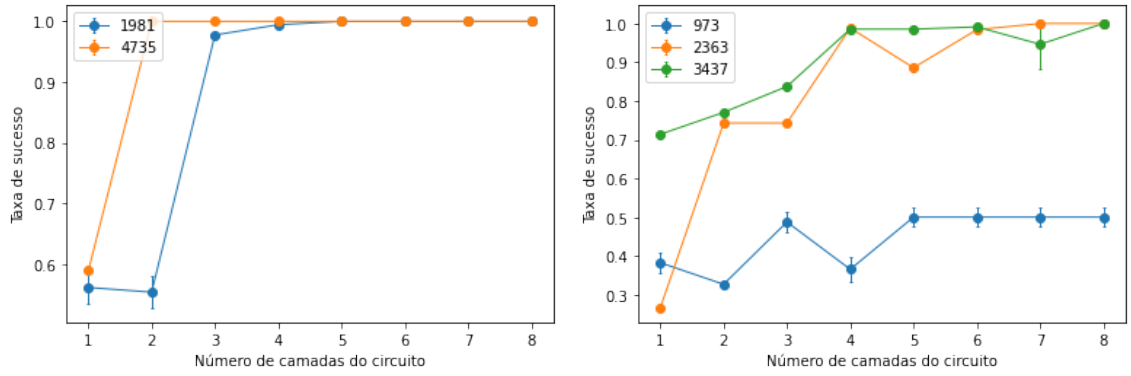
Biprimo "m"	Número de qubits	Número de "carry-bits"	Simetria p ↔ q	Tamanho de grade
1981 = 7 x 283	2	0	Não	6 x 6
4735 = 5 x 947	2	1	Não	6 x 6
973 = 7 x 139	3	0	Não	9 x 9
2363 = 17 x 139	3	1	Não	9 x 9
3437 = 7 x 491	3	2	Não	9 x 9
56153 = 233 x 241	4	0	Sim	12 x 12
1165 = 5 x 233	4	1	Não	12 x 12
1895 = 5 x 379	4	2	Não	12 x 12
291311 = 523 x 557	6	0	Sim	24 x 24
319 = 11 x 29	6	3	Não	24 x 24
961 = 31 x 31	6	4	Não	24 x 24

Tabela 4.3: Tabela descrevendo as instâncias utilizadas nas análises numéricas. Primeira coluna: Indica os números biprimos utilizados. Segunda coluna: Expressa o número total de qubits necessário para executar o algoritmo VQF após a aplicação das regras de pré-processamento descritas pelas Eqs. (3.7) e (4.1). Terceira coluna: Representa, dentro do número total de qubits necessário, o número de carry-bits. Quarta coluna: Indica se existe simetria entre os fatores p e q após a aplicação das regras de pré-processamento clássico. Quinta coluna: Tamanho de grade utilizado na inicialização de parâmetros por grid search.

taxa de sucesso do VQF e, dentro desse contexto, as hipóteses levantadas na seção 4.2.1. Sendo assim, foram realizadas diversas simulações numéricas, alterando gradualmente o número de qubits, o número de carry-bits e a existência ou não de simetria $p \leftrightarrow q$, entre as instâncias utilizadas, com o objetivo de encontrar padrões.

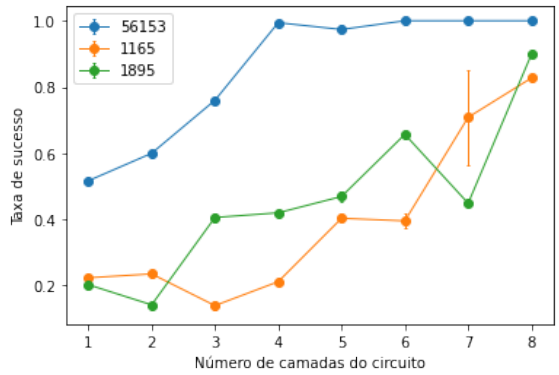
Dessa forma, na Fig. 4.14(a), comparou-se as taxas de sucesso para o VQF no caso das instâncias $m = 1981, 4735$, ambas com 2 qubits no total, e foi possível perceber que, no caso da segunda, essa convergiu mais rápido para o valor máximo. Até o momento, a única diferença presente consiste no fato de a segunda instância apresentar um carry-bit dos dois qubits totais, enquanto a primeira não apresenta nenhum.

Já na Fig. 4.14(b) comparou-se a taxa de sucesso para o VQF no caso das instâncias $m = 973, 2363, 3437$, todas com 3 qubits no total. Constatou-se que a instância $m = 973$ apresentou a pior taxa de sucesso, seguida pela instância $m = 2363$ e, por último, por $m = 3437$, tendo as duas últimas convergido para o valor máximo. Pode-se notar que a única diferença na disposição das três instâncias é o aumento gradual do número de carry-bits presentes no número total de qubits de cada uma: zero carry-bits no caso da primeira, um no caso da segunda e dois no caso da terceira. De posse desse resultado e do descrito anteriormente, inicia-se a percepção de um padrão: o aumento do número de carry-bits, dentro do número total de qubits da instância, promove um aumento da taxa de sucesso do algoritmo como um todo. Por fim, o resultado desse grupo de instâncias é inferior ao apresentado pelo grupo anterior, indicando que um aumento no número de qubits total promove uma redução na taxa de sucesso alcançada.

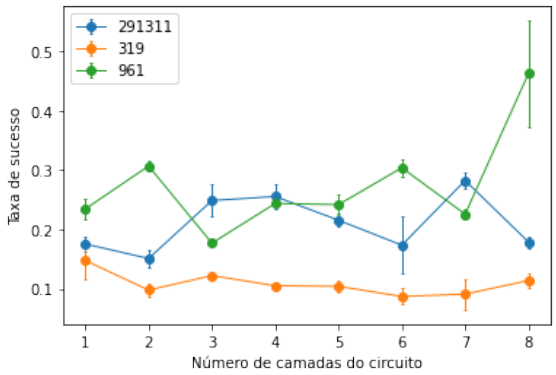


(a) Instâncias com 2 qubits

(b) Instâncias com 3 qubits



(c) Instâncias com 4 qubits



(d) Instâncias com 6 qubits

Figura 4.14: Relações numéricas - taxa de sucesso do algoritmo VQF em relação ao número de camadas do QAOA.

Em seguida, na Fig. 4.14(c), analisou-se a taxa de sucesso no caso das instâncias $m = 56153, 1165, 1895$, com 4 qubits no total, cada uma. A instância $m = 56153$ apresentou o melhor resultado para a taxa de sucesso, exibindo uma boa escala, tendo convergido rapidamente para o valor máximo. Já as instâncias $m = 1165, 1895$ apresentaram uma escala menos eficiente para a taxa de sucesso, atingindo valores próximos do máximo. Observa-se ainda que a primeira instância é a única, dentre as três, que apresenta simetria $p \leftrightarrow q$ e não possui carry-bits. Já as instâncias $m = 1165, 1895$ possuem, respectivamente, um e dois carry-bits, tendo a segunda apresentado resultados melhores que a primeira. Sendo assim, fica reforçado o padrão do efeito da presença de carry-bits mencionado anteriormente e inicia-se a percepção de um novo padrão: a existência de simetria $p \leftrightarrow q$ promove um aumento significativo da taxa de sucesso.

Por fim, na Fig. 4.14(d), verificou-se o comportamento da taxa de sucesso para as instâncias $m = 291311, 319, 961$, que possuem 6 qubits. A instância $m = 961$ apresentou o melhor resultado para a taxa de sucesso, seguida por $m = 291311$ e $m = 319$, em sequência. Como visto anteriormente, o resultado relativo entre as

instâncias $m = 961$ e $m = 319$ é justificado pelo maior número de carry-bits. Já o resultado apresentado pela instância $m = 291311$, em relação à instância $m = 319$ é justificado pela presença de simetria $p \leftrightarrow q$, reforçando o novo padrão observado anteriormente. Como $m = 961$ convergiu para valores maiores que $m = 291311$, pode-se perceber que, a partir de certo ponto, o número de carry-bits pode ser mais relevante que a presença de simetria $p \leftrightarrow q$.

Sendo assim, dos resultados númericos acima, pode-se concluir que o aumento do número de qubits para representar o Hamiltoniano de Ising e, consequentemente para a construção do circuito quântico parametrizado, tende a ocasionar uma redução na taxa de sucesso do algoritmo VQF. Tal resultado é esperado, uma vez que aumenta a complexidade e profundidade do circuito quântico.

Finalmente, pode-se ainda concluir que a presenca de simetria $p \leftrightarrow q$ e de carry-bits parece promover um aumento da taxa de sucesso. No primeiro caso, o fenômeno observado faz sentido, uma vez que a simetria faz com que um maior número de estados medidos codifiquem os valores corretos remanescentes dos fatores p e q, já que os seus bits remanescentes são complementares e os já calculados pelo pré-processamento clássico são iguais, conforme detalhado na seção 4.2. Já no segundo caso, a presença de carry-bits também promove um aumento do número de estados que codifica corretamente as variáveis restantes p_i e q_i, visto que as variáveis z_{ij}, devido à natureza da implementação, são irrelevantes em termos de medição de estados, conforme descrito na seção 3.2.2.

4.2.5 Otimização via QWOA

A simulação realizada nesta seção tem como objetivo comparar os comportamentos do algoritmo QWOA com o QAOA . Para este fim, foi utilizado o método de inicialização de parâmetros por grid search, o algoritmo clássico de otimização L-BFGS-B e as instâncias já descritas na tabela 4.2.[9]

Sendo assim, analisou-se a taxa de sucesso, o valor da função objetivo e o número de avaliações da função objetivo realizada pelo otimizador clássico para os dois casos. Percebe-se, na Fig. 4.15, que, para as instâncias $m = 1981, 2363, 56153$, a taxa de sucesso no caso do QWOA escala bem, convergindo para o valor máximo, assim como no caso do QAOA. Entretanto, nota-se ainda que, para o primeiro algoritmo, esse patamar foi atingido em um número menor de camadas para $m = 2363, 56153$; tendo escalado melhor para $m = 1981$. Já para o caso das instâncias $m = 319, 69169, 291311$, a taxa de sucesso não escalou tão bem, apresentando melhoras marginais. Cabe ressaltar ainda que apresentou valores maiores que o

[9]Foi considerada uma tolerância de $tol = 10^{-5}$ para o algoritmo de otimização clássico e o número de amostras $v = 1000$. As barras verticais de erro nos gráficos indicam, cada uma, um desvio padrão em três instâncias do problema.

QAOA para as instâncias $m = 319, 291311$ e menores para $m = 69169$.

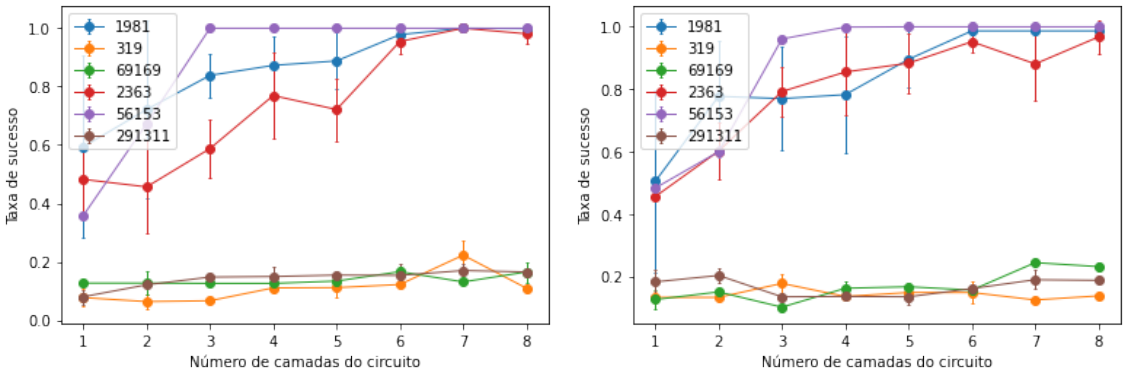
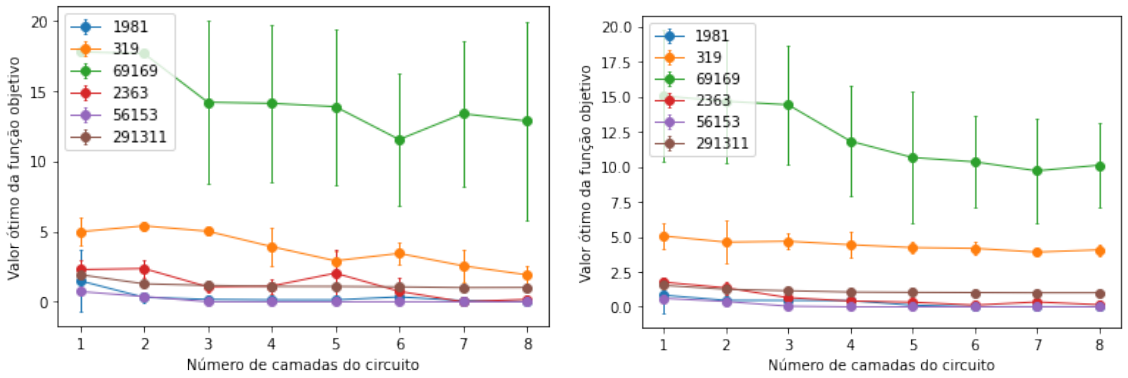


Figura 4.15: (a) O valor da taxa de sucesso para cada instância do problema, utilizando o algoritmo QWOA. (b) O valor da taxa de sucesso para cada instância do problema, utilizando o algoritmo QAOA.

Dessa forma, pode-se agora analisar o valor da função objetivo, de acordo com o definido na Eq. (3.32) (Fig. 4.16). Conforme esperado e já observado anteriormente para as demais simulações, o valor da função objetivo teve um comportamento inverso ao da taxa de sucesso. Nesse contexto, observa-se que o seu valor, no caso das instâncias $m = 1981, 2363, 56153$, converge para zero e, em relação ao QAOA, em camadas anteriores. Por outro lado, em relação ao segundo grupo de instâncias, o valor da função objetivo é minimizado, porém em escala marginal. Assim como o ocorrido no caso do QAOA, simulações numéricas indicaram que o algoritmo de otimização ficou preso em um mínimo local nas instâncias $m = 291311, 69169$ e que um número maior de camadas até a convergência é necessário para a instância $m = 319$. Entretanto, uma diferença relevante foi constatada: o valor da função objetivo convergiu para zero e, consequentemente, a taxa de sucesso convergiu para um, no caso de m = 319, em $s = 18$ camadas no algoritmo QWOA, enquanto que, no algoritmo QAOA, apresentou valores de taxa de sucesso na faixa de 0,9 nas simulações com $s = 30$ camadas, tendo convergido para o valor máximo em $s = 59$ camadas.[10]

Finalmente, examinou-se o número de avaliações da função objetivo até a convergência do algoritmo clássico de otimização, conforme disposto na Fig. 4.17. Os resultados, assim como no caso do QAOA, indicam que a escala do número de avaliações é aproximadamente linear no número de camadas e aproximadamente independente do tamanho do problema. Somado a isso, com exceção da discrepância

[10] As simulações para verificar a convergência para a instância m = 319 foram realizadas nas mesmas condições para a comparação entre os dois casos. Ambos com mesma inicialização de parâmetros e mesmo algoritmo clássico de otimização.



(a) (b)

Figura 4.16: (a) O valor da função objetivo $E(\vec{\gamma}, \vec{t})$ (Eq. (3.32)) para cada instância do problema, utilizando o algoritmo QWOA. (b) O valor da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) para cada instância do problema, utilizando o algoritmo QAOA.

observada em relação à instância m = 319, o número de avaliações é similar para os dois algoritmos.

Pode-se agora então refletir sobre os resultados obtidos. Esses indicam que o QWOA apresentou melhores taxas de sucesso para as instâncias $m = 319, 1981, 2363, 56153, 291211$ de 2, 3, 4, 6 e 6 qubits, respectivamente; tendo apresentado uma pior taxa de sucesso no caso da instância $m = 69169$, de 8 qubits.

Uma hipótese para essa ocorrência é o fato de o operador mixer continuous-time quantum-walk implementado ser beaseado em uma continuous-time quantum walk em um grafo completo (há conexão direta no grafo entre todas as soluções viáveis), enquanto o operador mixer transverse-field do QAOA, mixer original do QAOA, é baseado em um hipercubo (há conexão direta apenas entre as soluções viáveis com diferença unitária de distância de hamming); o que pode permitir que se espalhe mais rápido no grafo, alcançando as soluções de melhor qualidade mais rápido para grafos menores.

Ao analisar a estrutura do operador mixer do QWOA (continuous-time quantum walk mixer) e o operador Grover-mixer [62], percebe-se grande semelhança, tendo inclusive sido mencionado que o Grover-mixer é efetivamente uma caminhada quântica em tempo-contínuo (continuous-time quantum walk) em um grafo completo [63].

Nesse contexto, trabalhos recentes ([64], [65], [66]) estudaram o comportamento do algoritmo QAOA no âmbito de problemas Max 2-SAT e Max 3-SAT, comparando diferentes operadores mixers, estando inclusos o transverse-field mixer (mixer campo-transversal) e o Grover-mixer, já mencionados anteriormente. Sendo assim, os resultados obtidos demonstraram que o operador Grover-mixer foi melhor que o operador transverse-field mixer para instâncias dos problemas de até 7 qubits, en-

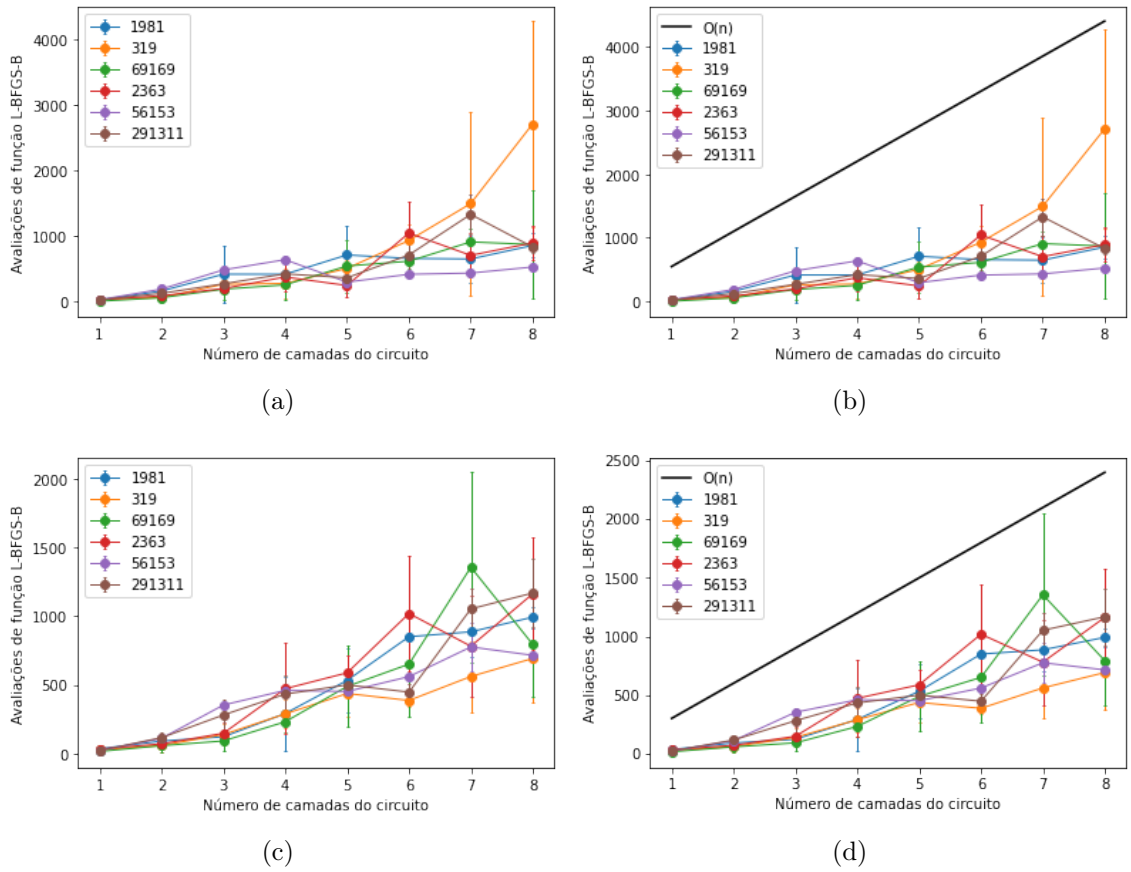


Figura 4.17: (a) Número de avaliações da função objetivo $E(\vec{\gamma}, \vec{t})$ (Eq. (3.32)) até o algoritmo L-BFGS-B convergir, utilizando o algoritmo QWOA. (c) Número de avaliações da função objetivo $E(\vec{\beta}, \vec{\gamma})$ (Eq. (3.21)) até o algoritmo L-BFGS-B convergir, utilizando o algoritmo QAOA. (b) e (c) Aproximadamente linear no número de camadas n.

quanto o inverso foi observado para instâncias a partir de 8 e até 14 qubits (maior tamanho de instâncias testadas). Tais resultados estão alinhados com o que foi observado acima para o caso deste trabalho, que trata do problema de fatoração como otimização binária.

Os resultados mencionados são importantes no contexto deste trabalho, visto que, assim como o Max 2-SAT e o Max 3-SAT, o problema de otimização combinatória tratado no VQF também é um problema irrestrito, no sentido que busca-se minimizar uma função objetivo em que, a princípio, todas as soluções são viáveis.

Tal fato, aliado à semelhança e possível equivalência entre o operador Grover-mixer e o operador continuous-time quantum walk, sugere que o mesmo comportamento observado para aqueles problemas com o operador Grover-mixer pode ser observado no problema de fatoração como otimização binária com o operador continuous-time quantum walk. No caso, os resultados obtidos neste trabalho indicam que tal comportamento de fato ocorre, entretanto simulações numéricas de

mais instâncias, com tamanhos maiores (maior número de qubits) e com um número maior de camadas são necessárias para uma melhor análise.

Sendo assim, pode-se concluir que o estudo do VQF com a variação do operador mixer utilizado no QAOA original, realizado neste trabalho, é um importante passo na compreensão do real poder do VQF e no estabelecimento deste algoritmo como referência para aplicações em computadores quânticos, de modo que permita referenciar a capacidade de um dispositivo quântico para realizar operações unitárias arbitrárias [4].

4.3 Simulação com ruído

O objetivo do estudo realizado nesta seção era analisar e entender o efeito de ruído no algoritmo VQF, assim como na variação utilizando QWOA, e com isso compreender melhor a possibilidade de escalabilidade do algoritmo em ambientes ruidosos que apresentam alta fidelidade ao encontrado na realidade atual.

Na seção 4.2 anterior, as simulações foram realizadas no modelo de caixa preta, ou seja, os estado finais parametrizados foram calculados algebricamente, de acordo com as Eqs. (3.20)($|\vec{\beta}, \vec{\gamma}\rangle$) e (3.31) ($|\vec{\gamma}, \vec{t}\rangle$), e inicializados diretamente na classe nativa *Quantum Circuit*, presente no framework *Qiskit* utilizado; assim como os valores das funções objetivos foram calculados algebricamente de acordo com as Eqs. (3.21) ($E(\vec{\beta}, \vec{\gamma})$) e (3.32) ($E(\vec{\gamma}, \vec{t})$).

Entretanto, para esta simulação, foi utilizado como backend o *FakeBoeblingenV2*, presente no módulo *Fake Provider* da IBM, que tem como objetivo simular sistemas quânticos IBM reais por meio da utilização de snapshots do sistema quântico original. Em específico, simula o processador quântico supercondutor *ibmq-boeblingen*, hoje já aposentado pela IBM. Sua escolha busca manter a estrutura de pesquisa utilizada por Karamlou et al. [4], além de possuir um número de qubits adequado para os tamanhos de instâncias utilizados.

Dessa forma, foi necessário seguir outra abordagem de construção dos circuitos para simulação, como será discutido nas próximas seções.

4.3.1 Processador Quântico Supercondutor

O processador quântico supercondutor é um processador quântico multi-qubits formado por qubits supercondutores que, por sua vez, armazenam informação nos graus quânticos de liberdade de osciladores anarmônicos nanofabricados, construídos a partir de elementos de circuito supercondutores [67]. A implementação de qubits com circuitos supercondutores integrados é discutida por Devoret et al. [68], assim como uma história evolutiva da plataforma é apresentada por Girvin [69].

Embora as simulações ideais considerem sistemas quânticos fechados, ou seja que não sofrem interações indesejadas do mundo externo, esse cenário utópico difere bastante do mundo real. No mundo real não existem sistemas completamente fechados, de modo que sistemas quânticos reais sofrem com essas interações indesejadas, que, por sua vez, aparecem como ruído nos sistemas quânticos de processamento de informações [6]. Ou seja, são fatores que podem afetar os cálculos realizados pelo computador quântico, uma vez que podem fazer com que a informação armazenada no qubit desapareça gradualmente.

Nesse contexto, o ruído pode ser separado em duas categorias: coerente e incoerente. O primeiro é descrito por operações unitárias que mantém a pureza do estado quântico na saída do circuito, sendo uma fonte comum desse tipo os erros sistemáticos originados de dispositivos não calibrados perfeitamente que, em consequência, não aplicam as portas lógicas desejadas de forma exata. Já o segundo é originado do emaranhamento entre um computador quântico e o seu ambiente externo, resultando em estados mistos, ou misturados, descrito como distribuições de probabilidade sobre diferentes estados puros. Dessa forma, implica em resultados que são aleatórios, independentemente de em qual base é realizada a medição.

Sendo assim, as principais fontes de ruído incoerente presentes no processador quântico são relaxamento de qubits (*qubit relaxation*) e decoerência, enquanto a fonte dominante de erro coerente é o acoplamento-ZZ (ZZ-*coupling*) residual entre os qubits transmon [[4], [70]]. Em relação ao erro coerente, a interação ocorre devido ao acoplamento entre os níveis mais altos de energia dos qubits, e é especialmente pronunciado em transmons devido a sua fraca anarmornicidade.

Descritas as características de ruído referentes ao processador em foco, pode-se especificar outras propriedades de grande relevância para este estudo. O *ibm-boeblingen* consiste em um sistema quântico de 20 qubits supercondutores (transmon qubits de frequência fixa), com portas lógicas quânticas single-qubit e two-qubit dirigidas por microondas, e apresenta um mapa de conectividade entre os qubits físicos confome especificado na Fig. 4.18. Essas propriedades são importantes, pois limitam os tipos de portas lógicas quânticas que podem ser utilizadas nos qubits (single-qubit e two-qubit) e entre quais qubits pode-se aplicar as portas two-qubit. Como consequência direta, tem-se a necessidade de adaptação dos circuitos quânticos lógicos para que possam ser executados no processador, conforme será visto na seção 4.3.2.

4.3.2 Construção de circuitos

Conforme discutido na seção 4.3.1, a arquitetura dos processadores quânticos supercondutores disponibilizados pela IBM limita as portas quânticas single-qubit e

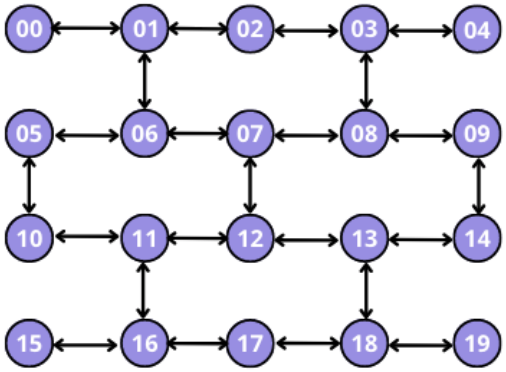


Figura 4.18: Mapa de conectividade dos qubits físicos do *FakeBoeblingenV2* (*ibm-boeblingen*). Os nós correspondem aos qubits físicos e as arestas direcionadas correspondem às portas lógicas $CNOT$ permitidas, com a origem e destino correspondendo, por sua vez, aos qubits de controle e alvo, respectivamente.

two-qubit que podem ser implementadas diretamente nos qubits físicos. As portas single-qubit implementadas diretamente são dadas pela biblioteca (ou alfabeto) de Clifford+T, que consiste de NOT, $CNOT$, H, S, $S^\dagger$, T, e $T^\dagger$, assim como pelas portas Pauli X, Y, Z e, por fim, pelas portas single-qubit um-parâmetro $U1$, dois-parâmetros $U2$ e três-parâmetros $U3$ [71].

Nesse contexto, construiu-se os circuitos para executar o algoritmo VQF original e com a variação utilizando QWOA, tendo sido necessárias adaptações para que a sua execução no *FakeBoeblingenV2* fosse possível. Sendo assim, iniciou-se pela construção dos circuitos lógicos para o algoritmo.

Primeiramente, discute-se a construção dos blocos dos circuitos que são iguais para as duas versões, ou seja, o preparo do estado inicial e a aplicação do operador de fase, dado pelas Eqs. (3.19) e (3.22). No caso do estado inicial, tanto no caso do QAOA, como no caso do QWOA (devido ao fato do problema ser irrestrito, ou seja todas as soluções são viáveis), tem-se que o estado inicial é dado por uma superposição uniforme sobre os estados base computacionais. Sendo assim, o estado pode ser obtido via aplicação da porta de Hadamard (H) em cada qubit, resultando no estado $|+\rangle^{\otimes n} = H^{\otimes n}|0\rangle^{\otimes n}$.

Pode-se agora então discutir a construção do operador de fase, a partir do Hamiltoniano de custo (de fase ou do problema) H_c, dado pela Eq. (3.11). Conforme discutido na seção 3.1.3, cada cláusula C'_i, obtida após a aplicação do pré-processamento clássico, pode ser mapeada para um termo $\hat{C}_i$ em um Hamiltoniano de Ising por meio da associação expressa na Eq. (3.10). Dessa forma, a partir da equação que expressa o hamiltoniano, percebe-se que é formado a partir da combinação de produtos tensoriais de portas de Pauli Z_i, aplicadas ao respectivo qubit i.

Sendo assim, o objetivo é conseguir decompor o operador de fase $U_H(\gamma)$ de modo

que seja possível implementá-lo utilizando portas quânticas conhecidas. Para este fim, primeiro deve-se atentar à seguinte propriedade:

Sejam A e B matrizes n x n. Pode-se então afirmar que $e^{(A+B)t} = e^{At}e^{Bt}$, $\forall\ t$, sse $[A, B] = 0$ [72].

Ou seja, a exponencial de uma soma de matrizes pode ser decomposta no produto de exponenciais das respectivas matrizes, desde que estas comutem entre si. Esse fato é essencial, pois sabe-se ainda que todas as matrizes diagonais comutam entre si [72]. Logo, como o operador de Pauli Z é uma matriz diagonal, tem-se que produtos tensoriais entre eles, assim como entre eles e a matriz Identidade (também diagonal), também o são, e, como consequência, a combinação desses produtos tensoriais de termos também é uma matriz diagonal. Dessa forma, utilizando essas propriedades, consegue-se decompor o operador de fase, que é a exponencial de uma soma de matrizes, em um produto de exponenciais de matrizes. Pode-se ver essa decomposição para o caso da instância $m = 2363$, utilizada neste trabalho:

$$
\begin{aligned}
&U_H(\gamma) = e^{-i\gamma\hat{H}_c} = \\
&= e^{-i\gamma(4I_2I_1I_0+0.5I_2I_1Z_0+1.5I_2Z_1Z_0-0.5I_2Z_1I_0-0.5Z_2I_1Z_0-0.5Z_2Z_1I_0-1.5Z_2I_1I_0)} = \\
&= e^{-i\gamma(4I_2I_1I_0)}e^{-i\gamma(0.5I_2I_1Z_0)}e^{-i\gamma(1.5I_2Z_1Z_0)}e^{-i\gamma(-0.5I_2Z_1I_0)}e^{-i\gamma(-0.5Z_2I_1Z_0)}e^{-i\gamma(-0.5Z_2Z_1I_0)} \\
&e^{-i\gamma(-1.5Z_2I_1I_0)}
\end{aligned}
\tag{4.6}
$$

onde há produtos tensoriais entre os termos simples de Pauli e o índice $i \in \{0, 1, 2\}$ indica em qual qubit o operador está sendo aplicado.

A partir da Eq. (4.6), pode-se verificar que cada exponencial corresponderá a uma composição de portas quânticas que irão representar cada produto de termos Pauli. Para este fim, primeiramente deve-se notar que os operadores I podem ser ignorados nos produtos de termos Pauli nos expoentes [6, 73]. Logo, baseado no exemplo acima, pode-se observar essa afirmação, na prática, na Eq. (4.7).

$$
e^{-i\gamma(-0.5Z_2I_1Z_0)} = e^{-i\gamma(-0.5Z_2Z_0)} \tag{4.7}
$$

Whitfield et al. e Nielsen e Chuang [6, 73] demonstram ainda outras equivalências para os produtos de termos de Pauli que utilizam apenas portas single-qubit (R_z) e two-qubit ($CNOT$):

$$e^{iZ_2I_1Z_0t} = CNOT^{(0,2)}R_z^{(2)}(-2t)CNOT^{(0,2)} \tag{4.8}$$

$$e^{iZ_1Z_0t} = CNOT^{(0,1)}R_z^{(1)}(-2t)CNOT^{(0,1)} \tag{4.9}$$

$$e^{iZ_2Z_1Z_0t} = CNOT^{(0,1)}CNOT^{(1,2)}R_z^{(2)}(-2t)CNOT^{(1,2)}CNOT^{(0,1)} \tag{4.10}$$

onde os índices indicam os respectivos qubits em que as portas lógicas quânticas estão sendo aplicadas. Analogamente obtém-se as respectivas equivalências para produtos de termos de Pauli maiores.

Finalmente, de posse das equivalências expressas pelas Eqs. (4.8), (4.9), (4.10), pode-se construir os circuitos equivalentes, que estão representados na Fig. 4.19, onde $e^{-i\gamma Z_i} = R_z^i(2\gamma)$, ou seja, aplica uma rotação em torno do eixo Z no respectivo qubit i.

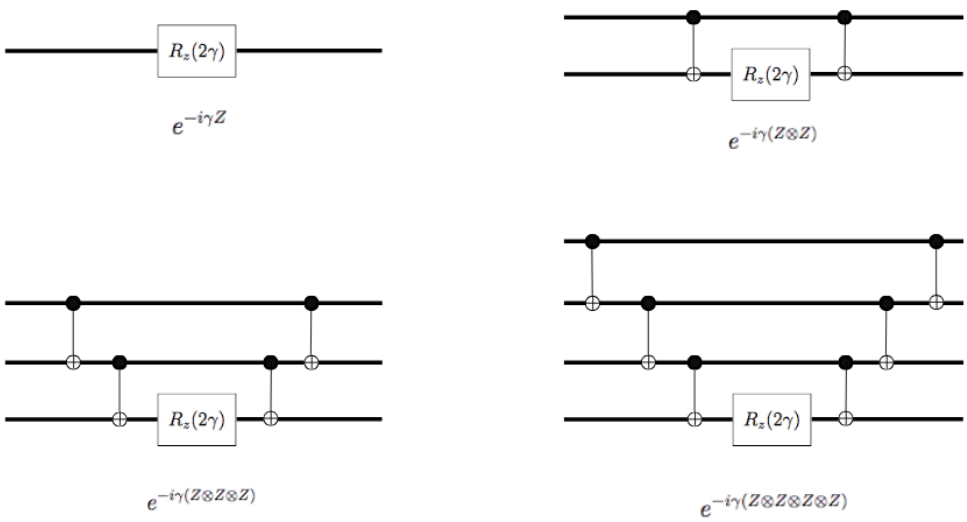


Figura 4.19: Mapeamento de equivalência em circuito dos componentes do operador decomposto, onde Z é o operador de Pauli Z.

Dessa forma, até o momento, é possível construir o estado inicial e o operador de fase, ambos comuns às duas versões testadas do algoritmo, conforme desejado inicialmente. Como próximo passo, faz-se necessário construir os circuitos equivalentes ao operador mixer do QAOA (transverse-field mixer) e ao operador mixer do QWOA (quantum-walk mixer).

No primeiro caso, pode-se observar, a partir da equação $U_a(\beta) = e^{-i\beta\sum_0^{n-1}X_i}$, que o operador mixer transverse-field original do QAOA consiste em uma exponenencial de soma de operadores de Pauli X aplicados a cada um dos qubits. Dessa forma, pode-se usar a propriedade já citada de exponenciais de matrizes para decompor o operador em um produto de exponenciais de matrizes, conforme descrito na Eq.

(4.11).

$$U_a(\beta) = e^{-i\beta \sum_0^{n-1} X_i} = \prod_0^{n-1} e^{-i\beta X_i} \tag{4.11}$$

onde $e^{-i\beta X_i} = R_x^i(2\beta)$, ou seja, aplica uma rotação em torno do eixo X no respectivo qubit i.

Tem-se, portanto, que o operador mixer transverse-field do QAOA original consiste na aplicação de uma porta $R_x(2\beta)$ em cada qubit do circuito.

Resta, portanto, construir o circuito correspondente ao segundo caso, ou seja que execute o definido pela Eq. (3.29) ($U_W(t)$), que, conforme já dito, expressa a rotação do sistema sobre um estado específico $|\psi\rangle$. Com esse objetivo, adaptou-se o circuito fornecido por Bennet et al. [5], para o caso em foco, em que o problema de otimização combinatória é irrestrito e, portanto, todas as soluções são viáveis. Tal circuito pode ser observado na Fig. 4.20.

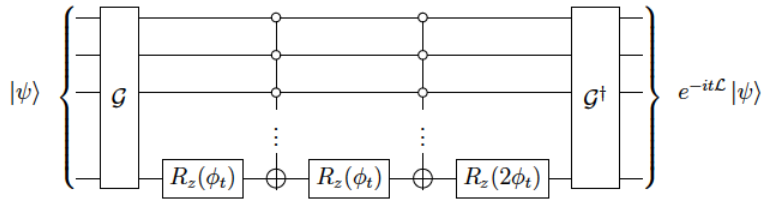


Figura 4.20: Circuito que executa a operação definida pela Eq. (3.29), que consiste na rotação do sistema sobre um estado específico $|\psi\rangle$. Onde $G = G^\dagger = H^{\otimes n}$, ou seja, prepara uma superposição uniforme entre todos os estados. Figura adaptada do artigo Bennett et al. [5].

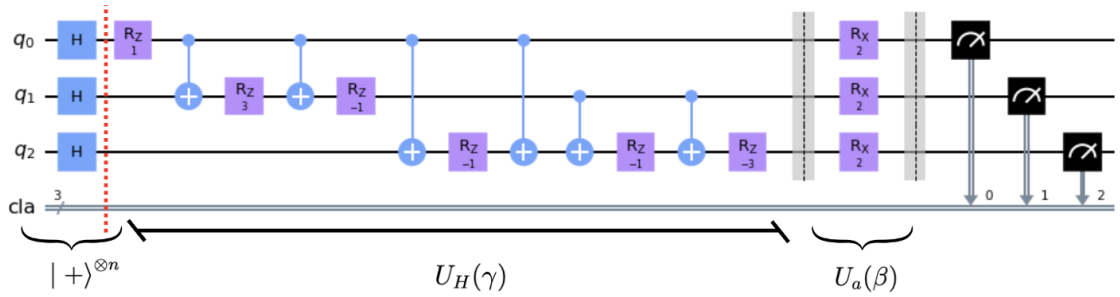
Finalmente, pode-se agora construir de forma completa os circuitos lógicos para as duas versões testadas, conforme desejado. Na Fig. 4.21 estão explícitos os dois circuitos para o caso da instância $m = 2363$.

Entretanto, para a execução dos circuitos lógicos no processador quântico, algumas observações ainda devem ser realizadas, conforme será discutido na seção 4.3.3.

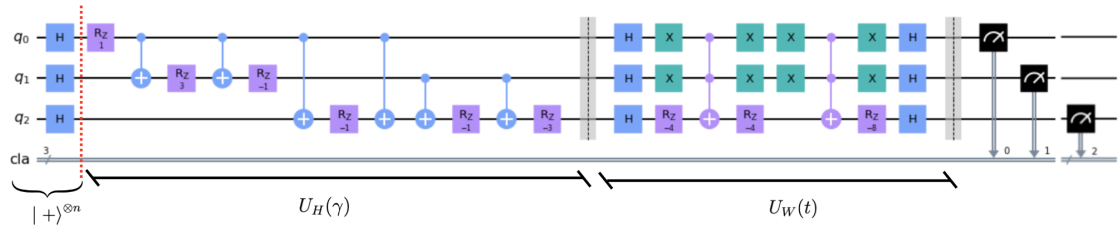
4.3.3 Transpilação de circuitos

Foi discutido na seção 4.3.2 o processo de construção dos circuitos lógicos utilizados neste trabalho para a execução do algoritmo VQF. Foram ainda reforçadas as limitações que a arquitetura dos processadores supercondutores disponibilizados pela IBM impõem aos circuitos que podem ser executados nos seus dispositivos.

Nesse contexto, é necessária a aplicação de um novo processo, a transpilação:



(a)



(b)

Figura 4.21: (a) Circuito lógico para execução do VQF com QAOA para a instância $m = 2363$. (b) Circuito lógico para execução do VQF com QWOA para a instância $m = 2363$. - Nos circuitos, os parâmetros β, γ e t foram inicializados com 1 para facilitar a visualização.

transpilação é o processo de reescrever um dado circuito de modo que corresponda à topologia de um dispositivo quântico específico, e/ou para otimizar o circuito para execução nos sistemas quânticos ruidosos atuais [74].

Para este fim, utilizou-se o módulo *Qiskit Transpiler* da IBM, com nível de otimização 1 (otimização leve) e com o layout de qubits físicos, para cada instância testada ($m = 1981, 2363, 56153$), de acordo com o definido na Fig. 4.22. Foi escolhido o modo de otimização 1 de modo a buscar otimização do circuito final (menor número de portas lógicas utilizadas, principalmente $CNOT$, que é fonte relevante de ruído; assim como menor profundidade de circuito), ao mesmo tempo que não gera um custo computacional tão alto. Já o layout de qubits físicos foi escolhido de modo a manter uma mesma estrutura topológica de conectividade entre os qubits físicos utilizados e por apresentarem fidelidades single-qubit e two-qubit mais altas (ou seja, precisão da operação quântica realizada em comparação com o esperado) e erros de leitura (na medição do qubit) mais baixos [4].

As portas lógicas quânticas presentes no circuito do VQF original (com QAOA) já podem ser implementadas diretamente, tendo traduções diretas para os seus equivalentes no backend (H, R_z, $CNOT$ e R_x). Já no caso da variação testada (com QWOA), as portas H, R_z, $CNOT$ e X possuem traduções diretas, não ocorrendo o mesmo com as portas NOT multi-controladas, que exigem transformações aplica-

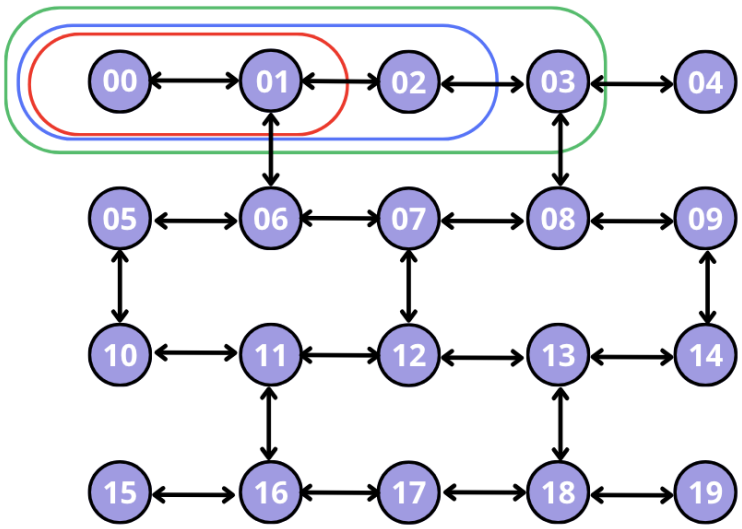
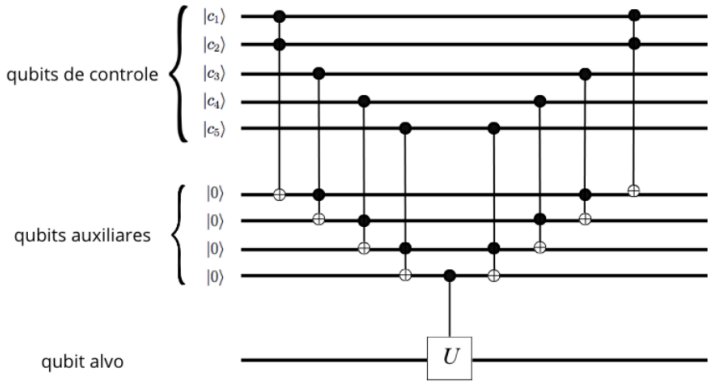


Figura 4.22: Mapa de conectividade dos qubits físicos do *FakeBoeblingenV2* (*ibm-boeblingen*) com os qubits escolhidos para cada instância utilizada. Para a instância $m = 1981$ são utilizados os qubits 0 e 1 (vermelho). Já para a instância $m = 2363$ são utilizados os qubits 0, 1 e 2 (azul). Finalmente, para a instância $m = 56153$, são utilizados os qubits 0, 1, 2 e 3 (verde).

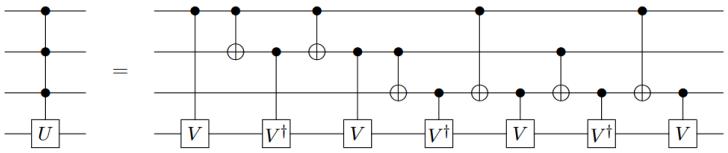
das pelo próprio módulo do transpiler (portas Toffoli e NOT 3,4-multi-controladas). Caso o número de qubtis de controle das portas NOT multi-controladas seja maior que 4, não há implementações diretas no Qiskit, ou seja, que são compreendidas diretamente pelo módulo transpiler. Sendo assim, faz-se necessária a sua decomposição em portas menores (com menor número de qubits).

Referente à decomposição das portas NOT multi-controladas que não possuem mapeamentos diretos realizados pelo módulo transpiler, Nielsen e Chuang [6] e Barenco et al. [7] propõem métodos para a decomposição de portas quânticas multi-controladas, conforme descrito na Fig. 4.23.

Como as instâncias utilizadas para a simulação com ruído foram de até 4 qubits no total, não foi necessária a decomposição mencionada anteriormente. Sendo assim, realizou-se a transpilação dos circuitos para a execução no backend. Na Fig. 4.24 pode-se observar o resultado para o caso da instância $m = 2363$. Percebe-se, portanto, que o operador mixer quantum-walk implica em uma profundidade de circuito bem maior que o operador mixer transverse-field, que requer profundidade 1 para implementação tanto no circuito lógico como no circuito transpilado. Uma análise mais profunda dos circuitos é realizada na seção 4.3.4. Os demais circuitos lógicos e transpilados para as instâncias testadas podem ser verificados no apêndice A.



(a) Método proposto por Nielsen e Chuang



(b) Método proposto por Barenco et al.

Figura 4.23: (a) Decomposição proposta por Nielsen and Chuang [6], exibindo rede que implementa a operação $C^n(U)$ para $n = 5$. Este método requer $n - 1$ qubits auxiliares adicionais, assim como $2(n - 1)$ portas Toffoli e uma única porta CU (unitário controlado), onde n é o número de qubits de controle. (b) Esquema retirado do artigo de Barenco et al. [7]. Este método utiliza um total de $2^n - 1$ portas V e $2^n - 2$ portas $CNOT$, onde novamente n é o número total de qubits e $V^4 = U$ no exemplo tratado.

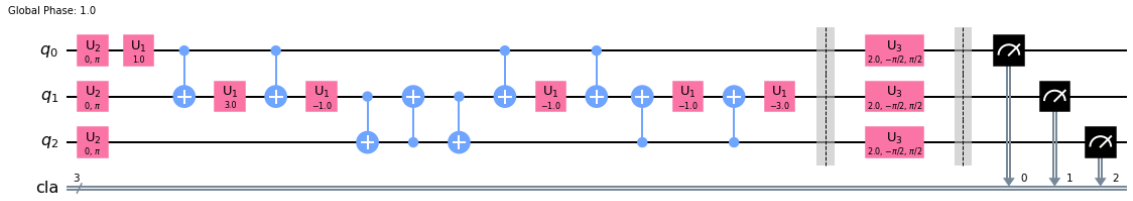
4.3.4 Análise de circuitos

Nas seções 4.3.2 e 4.3.3 foi discutida a construção dos circuitos lógicos e transpilados do VQF para execução no ambiente ideal e no ambiente ruidoso em foco.

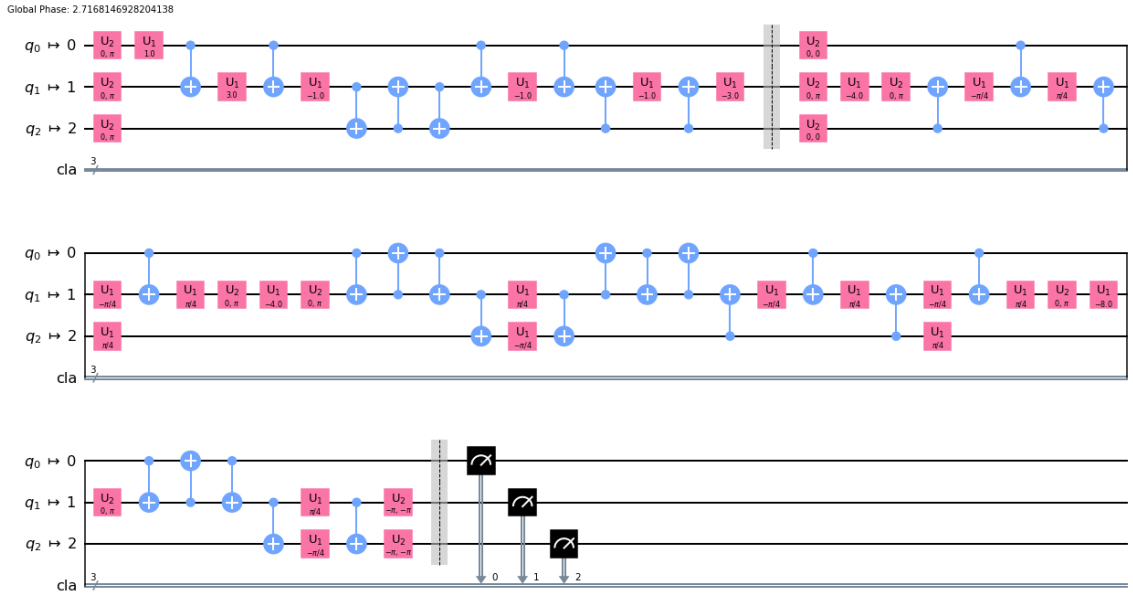
Nesse contexto, é importante analisar certas propriedades dos circuitos que influenciam diretamente na sua eficiência em executar as operações desejadas, como número de portas quânticas, seus tipos e a profundidade do circuito como um todo.

Em um computador quântico ideal, a qualidade da solução é elevada de forma monotônica com o aumento da profundidade do circuito. Entretanto, o aumento da profundidade em dispositivos NISQ atuais introduz ruído, deteriorando a qualidade da solução observada [75].

Cada porta lógica em um computador quântico possui uma taxa de erro determinada pelo qubit em que atuam (lembra-se ainda que cada qubit tem um erro de leitura característico). Somado a isso, portas que atuam em dois qubits são penalizadas por erros que dependem dos dois qubits físicos em que atua. Consequentemente, as taxas de erro para portas two-qubit são usualmente maiores que para as portas single-qubit, o que implica em um circuito com portas two-qubit apresentar mais



(a)



(b)

Figura 4.24: (a) Circuito transpilado para execução do VQF com QAOA, para a instância $m = 2363$, no backend *FakeBoeblingenV2*. (b) Circuito transpilado para execução do VQF com QWOA, para a instância $m = 2363$, no backend *FakeBoeblingenV2*. - Nos circuitos, os parâmetros β e γ e t foram inicializados com 1 para facilitar a visualização.

ruído do que um circuito com o mesmo número de portas, porém estas sendo portas single-qubit. Esse fato se provou essencial para compreender a diferença de resultados na taxa de sucesso para as duas variações implementadas, nos dois ambientes utilizados, conforme será discutido de modo mais profundo na seção 4.3.5.

Foi visto ainda nas seções anteriores que a topologia do processador quântico utilizado limita as conexões diretas que podem ser realizadas entre qubits, ou seja entre quais qubits podem ser aplicadas portas two-qubit (na Fig. 4.18, as arestas determinam entre quais qubits tais portas podem ser aplicadas e em quais sentidos). Dessa forma, faz-se necessária a realização de transformações no circuito de modo que portas two-qubit possam ser aplicadas entre qubits físicos que não possuem ligações diretas entre si. Essas transformações são realizadas via adição de portas SWAP que, por sua vez, são implementadas por três portas $CNOT$ cada no *IBM Qiskit Transpiler* [74]. Em consequência, cada porta $SWAP$ representa uma

operação bastante ruidosa no circuito transpilado, assim como custosa computacionalmente, uma vez que o módulo de transpilação executa um algoritmo heurístico estocástico (*SabreSwap* [76]) para realizar dito mapeamento.

Outra limitação consiste na necessidade de decomposição de portas 3+-qubit em portas two-qubit e single-qubit. Essa limitação tem implicação direta na transpilação do circuito do operador mixer quantum-walk, uma vez que possui portas NOT multi-controladas que, em consequência, devem ser decompostas. Como exemplo, uma porta Toffoli (NOT multi-controlada com dois qubits de controle) pode exigir até 6 portas $CNOT$ e outras portas single-qubit para ser executada no backend. Portanto, tal fato implica em um algoritmo de maior profundidade e mais afetado por ruído.

Realizadas essas considerações, efetuou-se uma análise das propriedades descritas anteriormente nos circuitos propriamente ditos, que, por sua vez, estão expressas nas tabelas do apêndice B.

Análise de circuito utilizando QAOA

Primeiramente, foram analisados os circuitos para o caso do algoritmo VQF com utilização do QAOA (original). Nesse contexto, as tabelas B.1, B.3 e B.5 explicitam a evolução dos requisitos para o circuito lógico, de acordo com o aumento do tamanho das instâncias. Pode-se perceber, a partir do exposto por elas, que há um aumento de profundidade do circuito, assim como do número de portas single-qubit e two-qubit ($CNOT$). Como a única diferença entre as instâncias é o circuito para execução do operador de fase $U_H(\gamma)$, visto que o operador mixer transverse-field ($U_a(\beta)$) permanece o mesmo, conclui-se que esse aumento se dá exclusivamente devido ao aumento de complexidade (sistema de cláusulas após o pré-processamento clássico com número crescente de clásusulas e váriaveis) do Hamiltoniano de Ising (H_I), conforme o esperado, devido ao aumento de tamanho, em ordem crescente, das instâncias $m = 1981, 2363, 56153$.

Por outro lado, pode-se perceber, a partir das tabelas B.1 e B.2, que o processo de transpilação do circuito, no caso da instância $m = 1981$, não altera a sua profundidade ou a quantidade e disposição das portas lógicas quânticas utilizadas. Isso se deve ao fato de serem utilizados apenas dois qubits físicos na simulação ruidosa, que apresentam ligação direta entre si, não exigindo, portanto, inserção de portas $SWAP$ (três portas $CNOT$ no circuito transpilado), e ao fato de o transverse-field mixer ser implementado em profundidade de circuito 1 (com apenas portas R_x, que, por sua vez, são traduzidas diretamente, assim como as demais portas single-qubit presentes no circuito lógico).

Esse comportamento, no entanto, não é observado no caso das instâncias $m = 2363, 56153$, que apresentam um aumento considerável de profundidade de

circuito e de portas $CNOT$, conforme pode ser observado nas tabelas B.3-B.6. Esse comportamento é explicado pela necessidade de inserção de portas SWAP pelo algoritmo de mapeamento, de modo a permitir a aplicação de portas two-qubit entre qubits físicos que não possuem ligação direta entre si. Finalizada essa etapa, pode-se agora analisar os circuitos utilizando QWOA.

Análise de circuito utilizando QWOA

Novamente, em um primeiro plano, foi analisada a evolução dos requisitos entre os circuitos lógicos correspondentes às instâncias utilizadas, conforme descrito nas tabelas B.7, B.9 e B.11.

Assim como no caso anterior, pode-se perceber, a partir do exposto por elas, que há um aumento de profundidade do circuito, assim como do número de portas single-qubit e two-qubit ($CNOT$). Entretanto, há uma nova variável presente nos circuitos: portas NOT multi-controladas. Com o aumento de tamanho das instâncias, aumenta-se o número de qubits de controle das portas NOT multi-controladas. Um detalhe que deve ser observado, no entanto, é que as portas NOT multi-controladas e as portas $CNOT$ do operador mixer quantum-walk $U_W(t)$ são implementadas com profundidade de circuito 1.

Sendo assim, o comportamento descrito anteriormente é explicado majoritariamente pelo aumento de complexidade do Hamiltoniano de Ising, assim como de algumas portas single-bit (H e X) introduzidas pelo operador mixer quantum-walk de acordo com o aumento de tamanho das instâncias.

Cabe agora analisar o efeito do processo de transpilação nos circuitos lógicos construídos. No caso da instância $m = 1981$, percebe-se, a partir das tabelas B.7 e B.8, que o processo de transpilação do circuito não ocasiona aumento de profundidade no circuito, assim como do número de portas single-qubit e two-qubit. Na realidade, é observada uma redução no número de portas single-qubit e da profundidade do circuito.

Em relação à manutenção do número de portas $CNOT$ durante o processo de transpilação, o comportamento é explicado da mesma forma que no caso anterior, ou seja devido ao fato das portas two-qubit serem aplicadas em qubits físicos que possuem ligações diretas entre si.

Já no caso da redução no número de portas single-qubit, esse fato é explicado pela aglutinação de portas lógicas realizadas pelo compilador sempre que possível. Por exemplo, se em uma mesma linha de circuito houver duas portas NOT (X) seguidas, o compilador irá convertê-las em uma única porta lógica equivalente I (identidade) e retirá-la do circuito. O mesmo ocorrendo com portas H, X, H em sequência, sendo convertidas em uma porta Z, equivalente. Outra possibilidade, efetuada pelo transpilador, é aglutinar portas em uma mesma porta no circuito transpilado. Como

exemplo, pode-se citar a aglutinação de uma porta X e uma porta H em uma porta 2-parâmetros $U2$, que irá executar as operações equivalentes.

Caso não seja desejável, por algum motivo, permitir essas aglutinações, deve-se colocar *barriers* (instruções presentes no Qiskit) de modo a separar tais portas e/ou até áreas do circuito como um todo. Dessa forma, esse recurso não é meramente estético, sendo um indicador visual de áreas do circuito, mas atuando também como uma diretiva para o compilador, de modo a delimitar áreas de tal forma que qualquer otimização ou reescrita seja restrita a elas [77].

Por último, em relação às instâncias $m = 2363, 56153$, nota-se que o processo de transpilação provoca um aumento expressivo na profundidade do circuito, assim como do número de portas single-qubit e two-qubit, conforme detalhado nas tabelas B.9-B.12. Esse comportamento é explicado por duas causas: a necessidade de inserção de portas $SWAP$ pelo algoritmo de mapeamento e a necessidade de decomposição das portas NOT multi-controladas.

Em relação à necessidade de inserção de portas $SWAP$, é o mesmo comportamento já explicado no caso com QAOA anterior. Já em relação à necessidade de decomposição das portas NOT multi-controladas, foi visto anteriormente nesta seção e na seção 4.3.3 que esse processo resulta em diversas portas $CNOT$ e outras portas single-qubit.

Finalmente, da análise caso a caso das propriedades dos circuitos lógicos e transpilados, tanto no caso com QAOA, como no caso com QWOA, pode-se perceber que a segunda versão resulta em circuitos de maior profundidade e maior número de portas lógicas, sejam elas single-qubit ou two-qubit. Essa diferença é explicada pela diferença entre os operadores mixer transverse-field e mixer quantum-walk. Na seção 4.3.5 é discutido como essas diferenças impactam os resultados obtidos para a taxa de sucesso do algoritmo VQF.

4.3.5 Ambiente ideal x ruidoso

As simulações realizadas nesta seção têm como objetivo comparar os resultados obtidos para a taxa de sucesso do algoritmo em ambiente ideal com os obtidos em ambiente ruidoso, de modo a verificar a sensibilidade do algoritmo, nos dois casos, ao ruído.

Dessa forma, realizou-se simulações para os dois casos (QAOA e QWOA), em ambiente ideal (backend *Aer*) e ambiente ruidoso (backend *FakeBoeblingenV2*), utilizando o método de inicialização de parâmetros por grid search e o algoritmo clássico de otimização L-BFGS-B[11]. Foram utilizadas ainda três instâncias, conforme

[11]Foi considerada uma tolerância de $tol = 10^{-5}$ para o algoritmo de otimização clássico e o número de amostras $v = 1000$. As barras verticais de erro nos gráficos indicam, cada uma, um desvio padrão em três instâncias do problema.

especificado nas Tabelas 4.4 e 4.5.

Biprimo "m"	Número de qubits	Número de "carry-bits"	Simetria p ↔ q	Tamanho de grade QAOA
1981 = 7 x 283	2	0	Não	6 x 6
2363 = 17 x 139	3	1	Não	12 x 12
56153 = 233 x 241	4	0	Sim	18 x 18

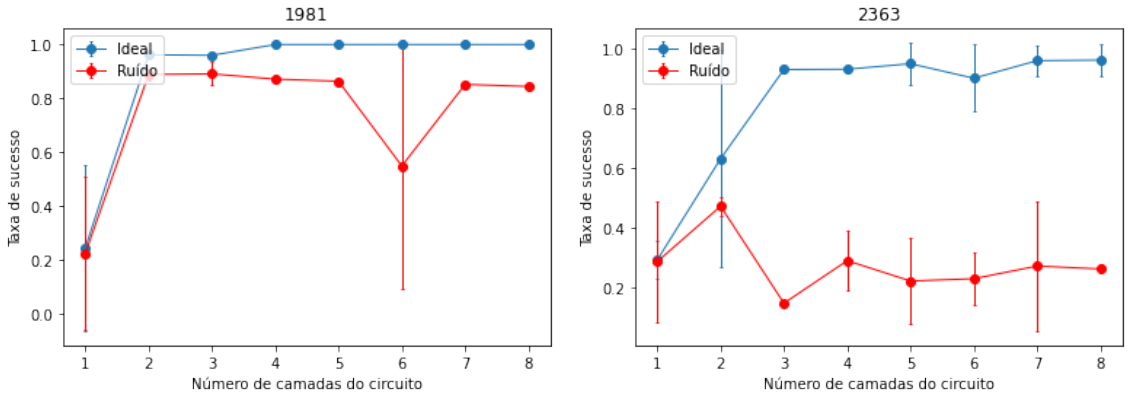
Tabela 4.4: Tabela descrevendo as instâncias utilizadas nas simulações em ambiente ruidoso para o caso utilizando QAOA. Primeira coluna: Indica os números biprimos utilizados. Segunda coluna: Expressa o número total de qubits necessário para executar o algoritmo VQF após a aplicação das regras de pré-processamento descritas pelas Eqs. (3.7) e (4.1). Terceira coluna: Representa, dentro do número total de qubits necessário, o número de carry-bits. Quarta coluna: Indica se existe simetria entre os fatores p e q após a aplicação das regras de pré-processamento clássico. Quinta coluna: Tamanho de grade utilizado na inicialização de parâmetros por grid search.

Biprimo "m"	Número de qubits	Número de "carry-bits"	Simetria p ↔ q	Tamanho de grade QWOA
1981 = 7 x 283	2	0	Não	12 x 12
2363 = 17 x 139	3	1	Não	12 x 12
56153 = 233 x 241	4	0	Sim	18 x 18

Tabela 4.5: Tabela descrevendo as instâncias utilizadas nas simulações em ambiente ruidoso para o caso utilizando QWOA. Primeira coluna: Indica os números biprimos utilizados. Segunda coluna: Expressa o número total de qubits necessário para executar o algoritmo VQF após a aplicação das regras de pré-processamento descritas pelas Eqs. (3.7) e (4.1). Terceira coluna: Representa, dentro do número total de qubits necessário, o número de carry-bits. Quarta coluna: Indica se existe simetria entre os fatores p e q após a aplicação das regras de pré-processamento clássico. Quinta coluna: Tamanho de grade utilizado na inicialização de parâmetros por grid search.

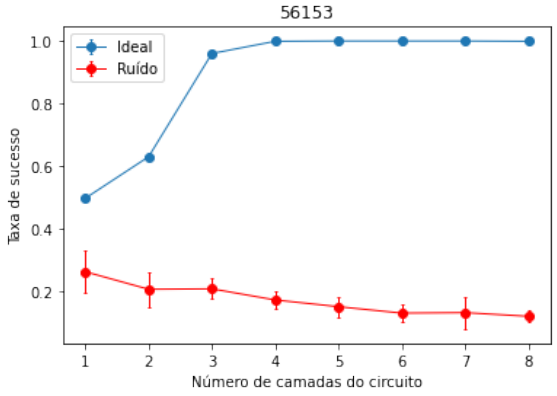
Iniciando pelo caso original (com QAOA), comparou-se os resultados obtidos para a taxa de sucesso no ambiente ideal e no ambiente com ruído (agora com implementação em circuito construído de acordo com o exposto nas seções anteriores), conforme pode observado na Fig. 4.25.

Em relação à instância $m = 1981$, pode-se perceber que, para uma camada, os dois casos (ideal e ruidoso), apresentam, aproximadamente, o mesmo valor para a taxa de sucesso. Entretanto, conforme o número de camadas do QAOA aumenta, percebe-se que a taxa de sucesso, para os dois casos, converge para valores distintos. Para o caso ideal, a taxa de sucesso converge para o valor máximo, enquanto para o caso ruidoso, é elevada até atingir um plateau, se mantendo, aproximadamente, na mesma faixa de valores. Dessa forma, observa-se que, para os dois casos, o valor da taxa de sucesso é elevada até atingir determinado valor, se mantendo no mesmo (no caso com ruído, de forma aproximada). Entretanto, cabe ressaltar que um comportamento de início marginal de declínio na taxa de sucesso, com o aumento do número de camadas, começa a ser percebido. Pode-se notar, portanto, o efeito



(a)

(b)



(c)

Figura 4.25: Variação com QAOA (original). (a) O valor da taxa de sucesso do VQF para a instância $m = 1981$. (b) O valor da taxa de sucesso do VQF para a instância $m = 2363$. (c) O valor da taxa de sucesso do VQF para a instância $m = 56153$.

do ruído característico do processador quântico supercondutor, de modo a reduzir o valor da taxa de sucesso obtida no ambiente com ruído. Conforme observado por Karamlou et al. [4] e mencionado anteriormente, a fonte dominante de ruído no processador quântico supercondutor é o acoplamento-ZZ residual nos qubits (ocasionado pela aplicação das portas $CNOT$). Isso fica ainda mais evidente quando analisamos os circuitos lógico e transpilado e as suas propriedades, dispostos na Fig. A.1 e tabelas B.1 e B.2, de modo que não há acréscimo de portas lógicas devido ao processo de transpilação.

Pode-se seguir então para a análise da instância $m = 2363$. Assim como no caso ideal da instância anterior, percebe-se que o valor da taxa de sucesso aumenta conforme o número de camadas do algoritmo também aumenta, até convergir para um valor, agora, bem próximo do máximo. Por outro lado, percebe-se um efeito mais acentuado do ruído para o caso da simulação em ambiente ruidoso. O valor da taxa de sucesso é severamente reduzido e mesmo o aumento do número de camadas do algoritmo que, idealmente, deveria aumentá-la, não consegue compensar o efeito

provocado pelo ruído. Isso se deve ao aumento da quantidade de portas lógicas quânticas no geral, principalmente das portas $CNOT$, e de profundidade do circuito no processo de transpilação, conforme pode ser percebido na Fig. A.2 e tabelas B.3 e B.4.

Por fim, pode-se analisar o caso da instância m = 56253. Observa-se um comportamento semelhante ao já mencionado anteriormente para a instância $m = 2363$. No ambiente ideal, auxiliado pela simetria $p \leftrightarrow q$ presente na instância, a taxa de sucesso converge rapidamente para o valor máximo. Já no ambiente ruidoso, nota-se novamente um efeito negativo acentuado no valor da taxa de sucesso do algoritmo, de modo que é reduzido a cada camada adicional. Mais uma vez, a adição de camadas não consegue compensar o efeito do ruído e, na prática, reduz ainda mais a taxa de sucesso, uma vez que o número de portas lógicas do circuito (principalmente $CNOT$) e profundidade do circuito, devido ao processo de transpilacão, aumenta em uma proporção maior do que a melhoria promovida pelo maior número de camadas. Tal fato pode ser observado na Fig. A.3 e tabelas B.5 e B.6.

Analisados os resultados para o caso com QAOA, pode-se seguir para o caso da variação utilizando QWOA. Novamente, comparou-se os resultados obtidos para a taxa de sucesso no ambiente ideal e no ambiente com ruído, conforme pode ser observado na Fig. 4.26.

Pode-se observar, a partir dos resultados nos gráficos, que o comportamento das instâncias no ambiente ideal e ruidoso foi o mesmo que o observado no caso da variação original anterior.[12] Logo, conclui-se que a variação proposta e testada também é sensível ao ruido característico do processador quântico supercondutor. Novamente, o principal agente causador de impacto negativo na taxa de sucesso do algoritmo é o ruído devido ao acoplamento-ZZ residual, provocado pela aplicação de portas $CNOT$ entre qubits. Outra vez, as restrições impostas pelo mapa de conectividade do *FakeBoeblingenV2* aos qubits físicos ocasionam um aumento expressivo de portas *CNOT*, no processo de transpilação, devido à necessidade de aplicação de portas SWAP para permitir a aplicação de portas *CNOT* entre qubits físicos que não possuem conexão direta entre si. Assim como no caso anterior, esse efeito é observado nas instâncias $m = 2363$ e $m = 56153$. Somado a isso, um efeito adicional para a multiplicação do número de portas lógicas no circuito transpilado, principalmente $CNOT$, é a necessidade de decomposição das portas NOT multi-controladas. Finalmente, os resultados piores obtidos para a taxa de sucesso para o ambiente ruidoso, em comparação com os obtidos, no mesmo cenário, com o QAOA, se devem à maior profundidade de circuito e número de portas lógicas quânticas ocasionados

[12] A única exceção consiste na instância $m = 56153$ no ambiente ideal, visto que ficou presa em um mínimo local, sendo necessários outros testes com maior número de camadas e tamanhos de grade para compreender completamente o seu comportamento. Os testes adicionais realizados com essas mudanças de parâmetros chegaram a culminar em taxas de sucesso de 0,7 a 0,8.

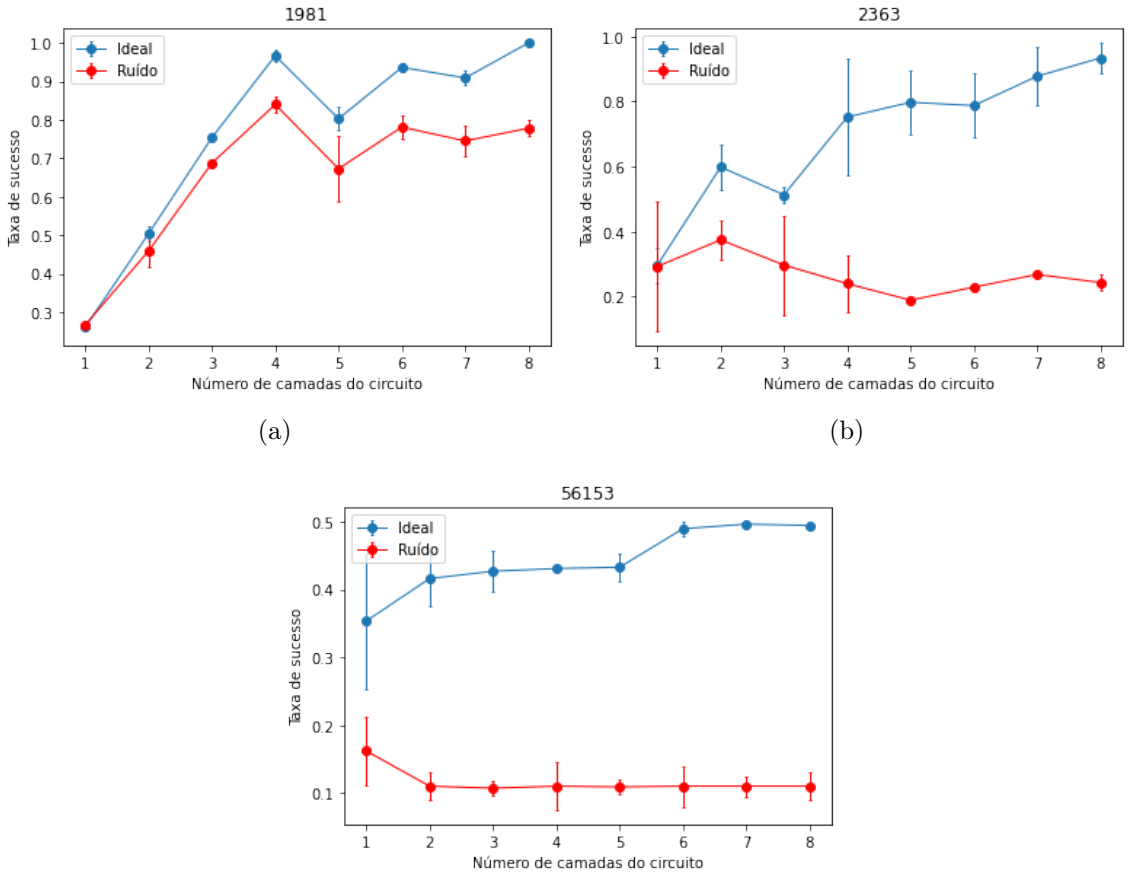


Figura 4.26: Variação com QWOA. (a) O valor da taxa de sucesso do VQF para a instância $m = 1981$. (b) O valor da taxa de sucesso do VQF para a instância $m = 2363$. (c) O valor da taxa de sucesso do VQF para a instância $m = 56153$.

pela transilaçao do operador quantum-walk mixer. As correlações descritas podem ainda ser analisadas nas Figs. A.4-A.6 e Tabelas B.7-B.12.

Por fim, cabe ressaltar que as diferenças entre os resultados obtidos, para o cenário ideal, entre a simulação realizada nesta seção e as simulações no modo caixa preta, descritas na seçao 4.19, se devem às diferenças de implementação, conforme já descrito no início da seção 4.3. Outro fato que deve ser notado é a diferença no tamanho de grade utilizado para certas instâncias. Isso foi necessário devido às diferenças de implementação mencionadas.

Capítulo 5

Conclusão

Neste trabalho, foi estudada a performance do algoritmo VQF em relação a uma variação utilizando o quantum-walk mixer do QWOA. Em ambas as variações, que consistem em abordagens híbridas, há duas partes principais, conforme descrito no capítulo 3: o pré-processamento clássico e o algoritmo quântico variacional de otimização. Enquanto a primeira parte tem como objetivo mapear o problema como uma otimização combinatória binária de minimização e codificá-lo no estado fundamental de um Hamiltoniano de Ising, reduzindo o número de qubits necessário sempre que possível, a segunda parte busca minimizar a função de custo e , com isso, encontrar o referido estado fundamental, onde os fatores originais estão codificados.

Em relação ao pré-processamento clássico, foi demonstrado que o esquema de pré-processamento é extremamente eficiente em reduzir o número de qubits necessário para representar o Hamiltoniano de Ising que codifica o problema. Foi visto ainda que o conhecimento prévio de propriedades dos fatores, mais especificamente o seu tamanho, reduz ainda mais significativamente o número de qubits necessário. Ressalta-se ainda que tal conhecimento prévio dificilmente está presente em situações reais, entretanto o seu uso é bastante interessante para a redução de requisitos de hardware necessários para a relização de simulações e experimentos, permitindo assim a averiguação de comportamentos e padrões.

Já em relação à parte de otimização, comparou-se o algoritmo de otimização clássico BFGS com a sua variação, o L-BFGS-B, verificando que de fato o segundo se comporta como uma boa aproximação para o primeiro, com uma menor utilização de recursos, conforme esperado de acordo com a sua proposta.

Por outro lado, quanto à questão de inicialização dos parâmetros a serem fornecidos como ponto inicial para o algoritmo de otimização clássico, constatou-se novamente a eficiência do método de grid search. Entretanto, a partir da análise de superfície de custo realizada por Karamlou et al. [4], e os testes realizados neste trabalho, observou-se que a estrutura da superfície de custo percorrida pelo otimizador clássico, para cada função objetivo, pode ser utilizada para o desenvolvimento de

heurísticas mais eficientes de inicialização de parâmetros, reduzindo assim o custo total de execução do algoritmo de forma relevante, uma vez que pode-se evitar um número elevado de avaliações do circuito quântico parametrizado.

No quesito de relações numéricas, concluiu-se que o número de qubits necessário para a construção do Hamiltoniano de Ising parece depender exclusivamente do esquema de pré-processamento clássico utilizado, a não ser que propriedades dos fatores originais sejam conhecidas previamente, o que, a priori, não será o caso. Entretanto, concluiu-se que a presença de simetria $p \leftrightarrow q$ entre os fatores e presença de carry-bits parece promover o aumento da taxa de sucesso, confirmando assim a hipótese deixada pelos autores no artigo original [2].

Focando-se agora na variação utilizando QWOA, percebeu-se que apresentou resultados melhores para instâncias do problema de tamanho até 6 qubits, tendo, no entanto, apresentado resultados piores para instâncias maiores (no caso testado de 8 qubits). Dessa forma, a princípio, não constitui uma boa substituição para a versão original no quesito de busca por escalabilidade, embora mais simulações numéricas sejam recomendadas.

Passando para as simulações realizadas em ambiente ruidoso, percebeu-se que as duas variações são sensíveis aos ruídos característicos dos processadores quânticos supercondutores. Ao analisar os respectivos circuitos lógicos e quânticos, conclui-se ainda que os requisitos são proibitivos, uma vez que são necessárias diversas portas $CNOT$ e, no caso da variação testada, diversas portas NOT multi-controladas. Uma hipótese deixada é a de que topologias de processadores que permitam o mapeamento físico de qubits com maior número de conexões entre eles, idealmente no modelo de grafo completo, promoveria uma redução altamente significativa no número de portas $CNOT$ utilizadas, principalmente no circuito transpilado, o que, por sua vez, reduziria o ruído introduzido.

O trabalho demonstrou também a relevância da proposição e teste de diferentes hamiltonianos mixers para o problema em foco. O preparo de mixers mais adequados ao problema, assim como de estados iniciais têm extrema importância na qualidade dos resultados obtidos.

Do demonstrado pelos trabalhos citados e pelos resultados obtidos neste estudo, conclui-se que o arcabouço dos algoritmos quânticos variacionais ainda está muito além da capacidade de se tornar uma real ameaça quântica aos sistemas criptográficos de chave pública existentes. Entretanto, o estudo de novas variações, para cada etapa dos algoritmos, torna-se essencial para a evolução das soluções e a possível chegada na vantagem quântica antes que dispositivos que sejam capazes de executar de forma eficiente o algoritmo de Shor sejam desenvolvidos.

Por fim, como trabalhos futuros, há o teste de diferentes mixers e circuitos para implementá-los, e possível utilização de aprendizado de máquina e análise das su-

perfícies de custo das funções objetivo do problema de modo a otimizar a etapa de inicialização de parâmetros.

Referências Bibliográficas

[1] CEREZO, M., ARRASMITH, A., BABBUSH, R., et al. "Variational quantum algorithms", *Nature Reviews Physics*, v. 3, n. 9, pp. 625–644, 2021.

[2] ANSCHUETZ, E., OLSON, J., ASPURU-GUZIK, A., et al. "Variational Quantum Factoring". In: Feld, S., Linnhoff-Popien, C. (Eds.), *Quantum Technology and Optimization Problems*, pp. 74–85, Cham, 2019. Springer International Publishing. ISBN: 978-3-030-14082-3.

[3] BURGES, C. J. "Factoring as optimization", *Microsoft Research MSR-TR-200*, 2002.

[4] KARAMLOU, A., SIMON, W., KATABARWA, A., et al. "Analyzing the performance of variational quantum factoring on a superconducting quantum processor", *Quantum Inf*, p. 7:156, 2021.

[5] BENNETT T, MATWIEJEW E, M. S., JB, W. "Quantum Walk-Based Vehicle Routing Optimisation", *Front. Phys.*, v. 9, n. 730856, Dec 2021. doi: 10.3389/fphy.2021.730856.

[6] NIELSEN, M. A., CHUANG, I. L. *Quantum Computation and Quantum Information*. Cambridge, Cambridge University Press, 2000.

[7] BARENCO, A., BENNETT, C. H., CLEVE, R., et al. "Elementary gates for quantum computation", *Phys. Rev. A*, v. 52, pp. 3457–3467, Nov 1995. doi: 10.1103/PhysRevA.52.3457. Disponível em: <`https://link.aps.org/doi/10.1103/PhysRevA.52.3457`>.

[8] RIVEST, R. L., SHAMIR, A., ADLEMAN, L. "A method for obtaining digital signatures and public-key cryptosystems", *Commun. ACM*, v. 21, n. 2, pp. 120–126, feb 1978. ISSN: 0001-0782. doi: 10.1145/359340.359342.

[9] CONSULTING, E. "What is RSA? How does an RSA work?" `https://www.encryptionconsulting.com/education-center/what-is-rsa/`, 2024/2024. Accessed: 2024-01-22.

[10] DICKENS, R. E. "What are Encryption Protocols and How do They Work?" https://www.encryptionconsulting.com/what-are-encryption-protocols-and-how-do-they-work/, 2021/2024. Accessed: 2024-01-22.

[11] SHOR, P. W. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer", *SIAM Review*, v. 41, n. 2, pp. 303–332, 1999. doi: 10.1137/S0036144598347011.

[12] GAGLIARDONI, T. K. S. R. "Quantum attack resource estimate: using Shor's Algorithm to break RSA vs DH/DSA vc ECC". https://research.kudelskisecurity.com/2021/08/24/quantum-attack-resource-estimate-using-shors-algorithm-to-break-rsa-vs-dh- 2021/2024. Accessed: 2024-01-22.

[13] PORTUGAL, R. "Basic Quantum Algorithms", *arXiv:2201.10574v6*, 2023.

[14] PRESKILL, J. "Quantum Computing in the NISQ era and beyond", *Quantum*, v. 2, pp. 79, aug 2018. ISSN: 2521-327X. doi: 10.22331/q-2018-08-06-79.

[15] CHENG, B., DENG, X.-H., GU, X., et al. "Noisy intermediate-scale quantum computers", *Frontiers of Physics*, v. 18, n. 2, mar. 2023. ISSN: 2095-0470. doi: 10.1007/s11467-022-1249-z. Disponível em: <http://dx.doi.org/10.1007/s11467-022-1249-z>.

[16] GIDNEY, C., EKERÅ, M. "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits", *Quantum*, v. 5, pp. 433, abr. 2021. ISSN: 2521-327X. doi: 10.22331/q-2021-04-15-433. Disponível em: <http://dx.doi.org/10.22331/q-2021-04-15-433>.

[17] GOUZIEN, E., SANGOUARD, N. "Factoring 2048-bit RSA Integers in 177 Days with 13436 Qubits and a Multimode Memory", *Physical Review Letters*, v. 127, n. 14, set. 2021. ISSN: 1079-7114. doi: 10.1103/physrevlett.127.140503. Disponível em: <http://dx.doi.org/10.1103/PhysRevLett.127.140503>.

[18] YAMAGUCHI, J., YAMAZAKI, M., TABUCHI, A., et al. "Estimation of Shor's Circuit for 2048-bit Integers based on Quantum Simulator". Cryptology ePrint Archive, Paper 2023/092, 2023. Disponível em: <https://eprint.iacr.org/2023/092>. https://eprint.iacr.org/2023/092.

[19] PHAN, V., PÖNNI, A., RAASAKKA, M., et al. "On quantum factoring using noisy intermediate scale quantum computers". 2022.

[20] DRIDI, R., ALGHASSI, H. "Prime factorization using quantum annealing and computational algebraic geometry", *Scientific Reports*, v. 7, 04 2016. doi: 10.1038/srep43048.

[21] JUN, K., LEE, H. "HUBO and QUBO models for prime factorization", *Scientific Reports*, v. 13, n. 1, jun. 2023. ISSN: 2045-2322. doi: 10.1038/s41598-023-36813-x. Disponível em: <http://dx.doi.org/10.1038/s41598-023-36813-x>.

[22] DHAULAKHANDI, R., BEHERA, B. K., SEO, F. J. "Factorization of large tetra and penta prime numbers on IBM quantum processor". 2023.

[23] FARHI, E., GOLDSTONE, J., GUTMANN, S. "A quantum approximate optimization algorithm", *arXiv preprint arXiv:1411.4028*, 2014.

[24] HADFIELD, S., WANG, Z., O'GORMAN, B., et al. "From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz", *Algorithms*, v. 12, n. 2, 2019. ISSN: 1999-4893. doi: 10.3390/a12020034. Disponível em: <https://www.mdpi.com/1999-4893/12/2/34>.

[25] YAN, B., TAN, Z., WEI, S., et al. "Factoring integers with sublinear resources on a superconducting quantum processor". 2022.

[26] QIU, L., ALAM, M., ASH-SAKI, A., et al. "Resiliency analysis and improvement of variational quantum factoring in superconducting qubit". In: *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, ISLPED '20, p. 229–234, New York, NY, USA, 2020. Association for Computing Machinery. ISBN: 9781450370530. doi: 10.1145/3370748.3406586. Disponível em: <https://doi.org/10.1145/3370748.3406586>.

[27] MARSH, S., WANG, J. B. "Combinatorial optimization via highly efficient quantum walks", *Phys. Rev. Res.*, v. 2, pp. 023302, Jun 2020. doi: 10.1103/PhysRevResearch.2.023302. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevResearch.2.023302>.

[28] DE LIMA MARQUEZINO, F., PORTUGAL, R., LAVOR, C. "Quantum Walks". In: *A Primer on Quantum Computing*, pp. 79–101, Cham, Springer International Publishing, 2019. ISBN: 978-3-030-19066-8. doi: 10.1007/978-3-030-19066-8_5. Disponível em: <https://doi.org/10.1007/978-3-030-19066-8_5>.

[29] BENNETT, T. K., MATWIEJEW, E., MARSH, S., et al. "Quantum Walk-Based Vehicle Routing Optimisation". In: *Frontiers of Physics*, 2021. Disponível em: <https://api.semanticscholar.org/CorpusID:238226813>.

[30] SLATE, N., MATWIEJEW, E., MARSH, S., et al. "Quantum walk-based portfolio optimisation", *Quantum*, v. 5, pp. 513, jul 2021. ISSN: 2521-327X. doi: 10.22331/q-2021-07-28-513. Disponível em: <http://dx.doi.org/10.22331/q-2021-07-28-513>.

[31] SIMMONS, G. J. "Cryptology". https://www.britannica.com/topic/cryptology, 2023/2024. Accessed: 2024-01-17.

[32] LTD, C. R. L. P. "A Brief History of Cryptography". https://www.cypher.com.au/crypto_history.htm, 2013/2024. Accessed: 2024-01-17.

[33] KAHN, D. *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet*. Rev sub ed. USA, Scribner, December 1996.

[34] OF STANDARDS, N. I., TECHNOLOGY. "Cryptography". https://www.nist.gov/cryptography, 2023/2024. Accessed: 2024-01-20.

[35] (CRYPTOMATHIC), D. M. T. "Summary of Cryptographic Algorithms - According to NIST". https://www.cryptomathic.com/news-events/blog/summary-of-cryptographic-algorithms-according-to-nist, 2019/2024. Accessed: 2024-01-20.

[36] SHOR, P. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134, 1994. doi: 10.1109/SFCS.1994.365700.

[37] N.S. DATTANI, N. B. "Quantum factorization of 56153 with only 4 qubits", *arXiv preprint arXiv:1411.6758*, 2014.

[38] CAO, D., DINH, K. "Leveraging Quantum Computing for the Ising Model to Simulate Two Real Systems: Magnetic Materials and Biological Neural Networks (BNNs)", *Journal of Quantum Information Science*, v. 13, pp. 138–155, 09 2023. doi: 10.4236/jqis.2023.133008.

[39] SINGH, S. "The Ising Model: Brief Introduction and Its Application". In: *Solid State Physics-Metastable, Spintronics Materials and Mechanics of Deformable Bodies-Recent Progress*, p. 19, London, IntechOpen, 02 2020. ISBN: 978-1-83881-165-5. doi: 10.5772/intechopen.90875.

[40] BRUSH, S. G. "History of the Lenz-Ising Model", *Rev. Mod. Phys.*, v. 39, pp. 883–893, Oct 1967. doi: 10.1103/RevModPhys.39.883. Disponível em: <https://link.aps.org/doi/10.1103/RevModPhys.39.883>.

[41] OBERMEYER, J. "The Ising Model in One and Two Dimensions". In: *Seminar on Statistical Physics at the University of Heidelberg*, 2020.

[42] BARKOUTSOS, P. K., NANNICINI, G., ROBERT, A., et al. "Improving Variational Quantum Optimization using CVaR", *Quantum*, v. 4, pp. 256, abr. 2020. ISSN: 2521-327X. doi: 10.22331/q-2020-04-20-256. Disponível em: <https://doi.org/10.22331/q-2020-04-20-256>.

[43] SCHRÖDINGER, E. "An undulatory theory of the mechanics of atoms and molecules", *Physical Review*, v. 28, pp. 1049–1070, 12 1926.

[44] GRIFFITHS, D. J., SCHROETER, D. F. *Introduction to Quantum Mechanics*. 3 ed. Cambridge, Cambridge University Press, 2018.

[45] BORGHI, R. "The variational method in quantum mechanics: an elementary introduction", *European Journal of Physics*, v. 39, n. 3, pp. 035410, apr 2018. doi: 10.1088/1361-6404/aaafd9. Disponível em: <https://dx.doi.org/10.1088/1361-6404/aaafd9>.

[46] FITZPATRICK, R. "Introductory Quantum Mechanics Variational Principle". https://phys.libretexts.org/Bookshelves/Quantum_Mechanics/Introductory_Quantum_Mechanics_(Fitzpatrick)/13%3A_Variational_Methods/13.01%3A_Variational_Principle, 2023/2023. Accessed: 2023-09-30.

[47] FARHI, E., GOLDSTONE, J., GUTMANN, S., et al. "Quantum Computation by Adiabatic Evolution", *arXiv preprint quant-ph/0001106*, 02 2000.

[48] PERRON, O. "Zur Theorie der Matrices", *Mathematische Annalen*, v. 64, n. 2, pp. 248–263, 06 1907. doi: 10.1007/BF01449896.

[49] FROBENIUS, G. "Ueber Matrizen aus nicht negativen Elementen", *Sitzungsberichte der Königlich Preussischen Akademie der Wissenschaften*, p. 456–477, 05 1912.

[50] MARSH, S., WANG, J. "A quantum walk assisted approximate algorithm for bounded NP optimisation problems", *Quantum Information Processing*, v. 18, 01 2019. doi: 10.1007/s11128-019-2171-3.

[51] KLUBER, G. "Trotterization in Quantum Theory", *arXiv:2310.13296v2 [quant-ph]*, 2023.

[52] BRIDI, G. A., DE LIMA MARQUEZINO, F. "Analytical results for the Quantum Alternating Operator Ansatz with Grover Mixer", *arXiv:2401.11056v1 [quant-ph]*, 2024.

[53] FLETCHER, R. *Practical Methods of Optimization.* Second ed. New York, NY, USA, John Wiley & Sons, 1987.

[54] MANOUCHEHRI, K., WANG, J. *Physical Implementation of Quantum Walks.* Quantum Science and Technology. Netherlands, Springer, 2014. ISBN: 9783642360138. doi: 10.1007/978-3-642-36014-5.

[55] CHILDS, A. M., GOSSET, D., WEBB, Z. "Universal Computation by Multiparticle Quantum Walk", *Science*, v. 339, n. 6121, pp. 791–794, feb 2013. ISSN: 1095-9203. doi: 10.1126/science.1229957. Disponível em: <http://dx.doi.org/10.1126/science.1229957>.

[56] VILFRED, V. "On circulant graphs", *Balakrishnan, R.; Sethuraman, G.; Wilson, Robin J., Graph Theory and its Applications (Anna University, Chennai*, pp. 34–36, 2001.

[57] CAMPS, D., VAN BEEUMEN, R., YANG, C. "Quantum Fourier transform revisited", *Numerical Linear Algebra with Applications*, v. 28, n. 1, pp. e2331, 2021. doi: https://doi.org/10.1002/nla.2331. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/nla.2331>.

[58] YODER, T. J., LOW, G. H., CHUANG, I. L. "Fixed-Point Quantum Search with an Optimal Number of Queries", *Physical Review Letters*, v. 113, n. 21, nov 2014. ISSN: 1079-7114. doi: 10.1103/physrevlett.113.210501. Disponível em: <http://dx.doi.org/10.1103/PhysRevLett.113.210501>.

[59] LIU, D., NOCEDAL, J. "On the limited memory BFGS method for large scale optimization", *Mathematical Programming*, v. 45, pp. 503:528, 1989.

[60] BYRD, R. H., LU, P., NOCEDAL, J., et al. "A Limited Memory Algorithm for Bound Constrained Optimization", *SIAM Journal on Scientific Computing*, v. 16, n. 5, pp. 1190–1208, 1995. doi: 10.1137/0916069.

[61] ZHOU, L., WANG, S.-T., CHOI, S., et al. "Quantum Approximate Optimization Algorithm: Performance, Mechanism, and Implementation on Near-Term Devices", *Phys. Rev. X*, v. 10, pp. 021067, Jun 2020. doi: 10.1103/PhysRevX.10.021067. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevX.10.021067>.

[62] BÄRTSCHI, A., EIDENBENZ, S. "Grover mixers for QAOA: Shifting complexity from mixer design to state preparation". In: *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pp. 72–82. IEEE, 10 2020. doi: 10.1109/QCE49297.2020.00020.

[63] BENNETT, T., WANG, J. B. "Quantum optimisation via maximally amplified states", *arXiv preprint arXiv:2111.00796*, 2021. Disponível em: <https://api.semanticscholar.org/CorpusID:240353991>.

[64] AKSHAY, V., PHILATHONG, H., MORALES, M. E. S., et al. "Reachability Deficits in Quantum Approximate Optimization", *Phys. Rev. Lett.*, v. 124, pp. 090504, Mar 2020. doi: 10.1103/PhysRevLett.124.090504. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevLett.124.090504>.

[65] GOLDEN, J., BÄRTSCHI, A., O'MALLEY, D., et al. "The Quantum Alternating Operator Ansatz for Satisfiability Problems". In: *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, v. 01, pp. 307–312, 2023. doi: 10.1109/QCE57702.2023.00042.

[66] GOLDEN, J., BÄRTSCHI, A., O'MALLEY, D., et al. "Numerical Evidence for Exponential Speed-Up of QAOA over Unstructured Search for Approximate Constrained Optimization". In: *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, v. 01, pp. 496–505, 09 2023. doi: 10.1109/QCE57702.2023.00063.

[67] KRANTZ, P., KJAERGAARD, M., YAN, F., et al. "A quantum engineer's guide to superconducting qubits", *Applied Physics Reviews*, v. 6, pp. 021318, 06 2019. doi: 10.1063/1.5089550.

[68] DEVORET, M., MARTINIS, J. "Implementing Qubits with Superconducting Integrated Circuits". In: *Quantum Information Processing - QUANTUM INF PROCESS*, v. 3, pp. 163–203, Berlin, Springer, 01 2005. ISBN: 978-0-387-23045-0. doi: 10.1007/0-387-27732-3_12.

[69] GIRVIN, S. M. "Basic Concepts in Quantum Information", *arXiv: Quantum Physics*, 2013. Disponível em: <https://api.semanticscholar.org/CorpusID:117437367>.

[70] KOCH, J., YU, T. M., GAMBETTA, J., et al. "Charge-insensitive qubit design derived from the Cooper pair box", *Phys. Rev. A*, v. 76, pp. 042319, Oct 2007. doi: 10.1103/PhysRevA.76.042319. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevA.76.042319>.

[71] DE ALMEIDA, A. A. A., DUECK, G. W., DA SILVA, A. C. R. "CNOT Gate Mappings to Clifford+T Circuits in IBM Architectures". In: *2019 IEEE 49th International Symposium on Multiple-Valued Logic (ISMVL)*, pp. 7–12, 2019. doi: 10.1109/ISMVL.2019.00010.

[72] SALMAN, M. A. S., BORKAR, V. "Exponential Matrix and Their Properties", *International Journal of Scientific and Innovative Mathematical Research (IJSIMR*, v. 4, pp. 53–63, Jan 2016. doi: http://dx.doi.org/10.20431/2347-3142.0401008.

[73] JAMES D. WHITFIELD, J. B., ASPURU-GUZIK, A. "Simulation of electronic structure Hamiltonians using quantum computers", *Molecular Physics*, v. 109, n. 5, pp. 735–750, 2011. doi: 10.1080/00268976.2011.552441. Disponível em: <`https://doi.org/10.1080/00268976.2011.552441`>.

[74] IBM. "Qiskit Transpiler". `https://docs.quantum.ibm.com/api/qiskit/transpiler`, 2023/ 2023. Accessed: 2023-09-30.

[75] JOHNSTUN, S., VAN HUELE, J.-F. "Understanding and compensating for noise on IBM quantum computers", *American Journal of Physics*, v. 89, n. 10, pp. 935–942, 10 2021. ISSN: 0002-9505. doi: 10.1119/10.0006204. Disponível em: <`https://doi.org/10.1119/10.0006204`>.

[76] LI, G., DING, Y., XIE, Y. "Tackling the Qubit Mapping Problem for NISQ-Era Quantum Devices". In: *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 1001–1014, 04 2019. doi: 10.1145/3297858.3304023.

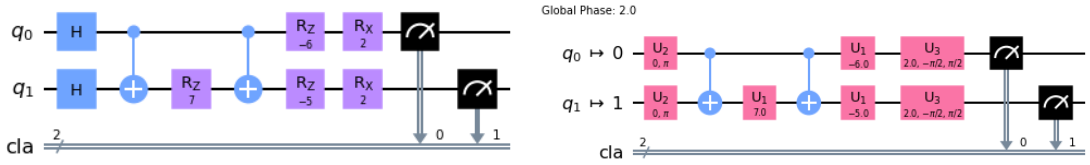
[77] IBM. "Qiskit Barrier". `https://docs.quantum.ibm.com/api/qiskit/qiskit.circuit.library.Barrier`, 2023/ 2023. Accessed: 2023-09-30.

Apêndice A

Circuitos lógicos e transpilados para o VQF

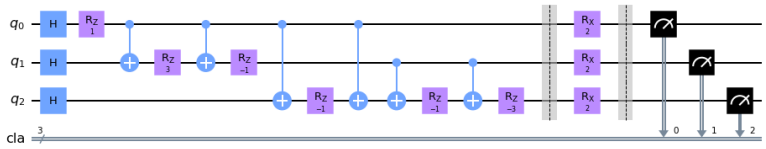
Nas Figs. A.1-A.6, são apresentados os circuitos lógicos e transpilados para o algoritmo VQF, para cada caso e instância testados, discutidas nas seções 4.3.2 e 4.3.3.

Foi utilizado o módulo *Qiskit transpiler* da IBM [74] para transpilar os circuitos lógicos (utilizados nas simulações em ambiente ideal - backend *Aer*) para os circuitos adequados ao backend *FakeBoeblingenV2*, tanto em termos de mapa de acoplamento (conectividade) de qubits físicos, como de portas lógicas quânticas aceitas.

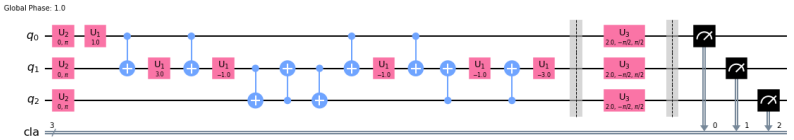


(a) (b)

Figura A.1: (a) Circuito lógico para execução do VQF com QAOA para a instância $m = 1981$. (b) Circuito transpilado para execução do VQF com QAOA, para a instância $m = 1981$, no backend *FakeBoeblingenV2*. - Nos circuitos, os parâmetros β, γ e t foram inicializados com 1 para facilitar a visualização.

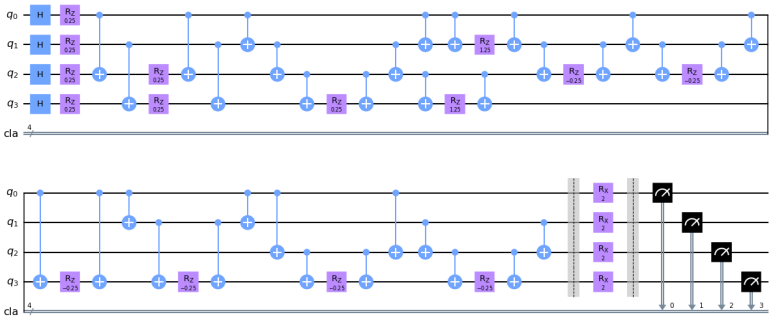


(a)

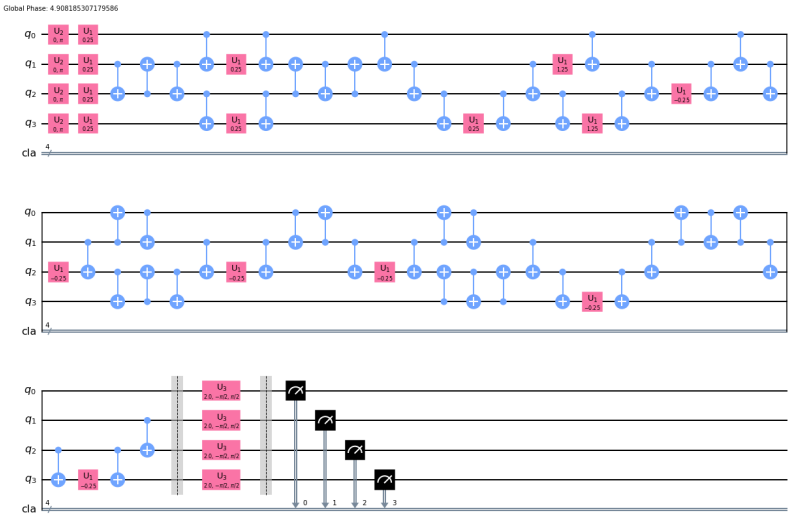


(b)

Figura A.2: (a) Circuito lógico para execução do VQF com QAOA para a instância $m = 2363$. (b) Circuito transpilado para execução do VQF com QAOA, para a instância $m = 2363$, no backend *FakeBoeblingenV2*. - Nos circuitos, os parâmetros β, γ e t foram inicializados com 1 para facilitar a visualização.

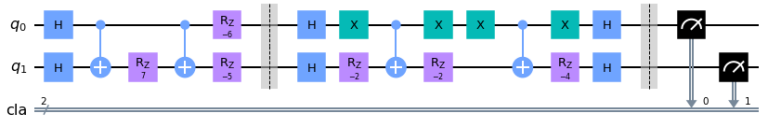


(a)

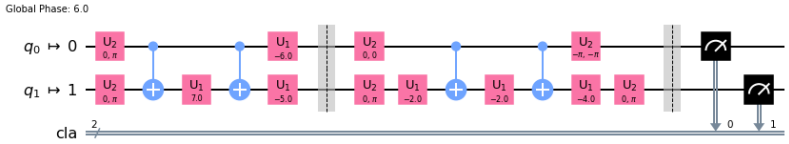


(b)

Figura A.3: (a) Circuito lógico para execução do VQF com QAOA para a instância $m = 56153$. (b) Circuito transpilado para execução do VQF com QAOA, para a instância $m = 56153$, no backend *FakeBoeblingenV2*. - Nos circuitos, os parâmetros β, γ e t foram inicializados com 1 para facilitar a visualização.

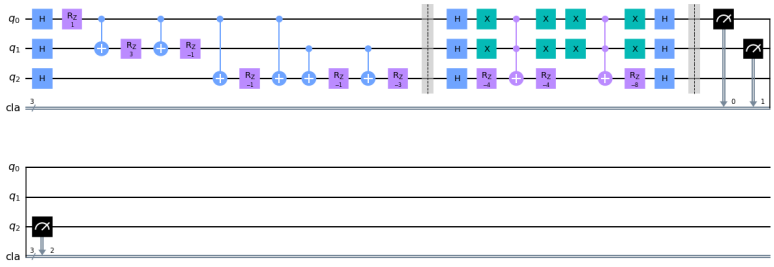


(a)

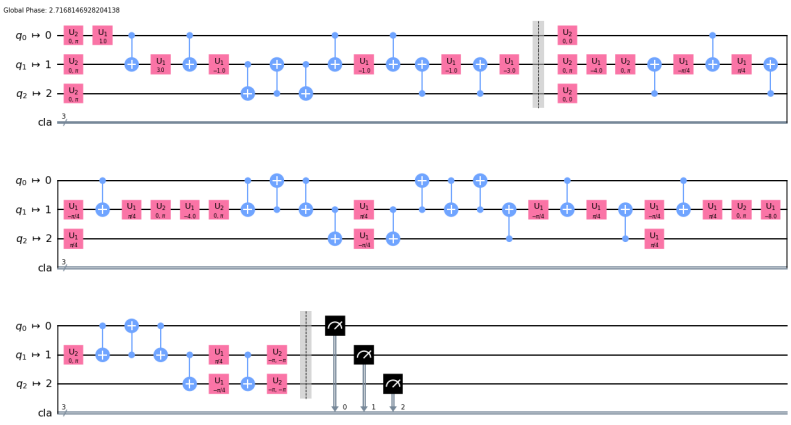


(b)

Figura A.4: (a) Circuito lógico para execução do VQF com QWOA para a instância $m = 1981$. (b) Circuito transpilado para execução do VQF com QWOA, para a instância $m = 1981$, no backend *FakeBoeblingenV2*. - Nos circuitos, os parâmetros β, γ e t foram inicializados com 1 para facilitar a visualização.

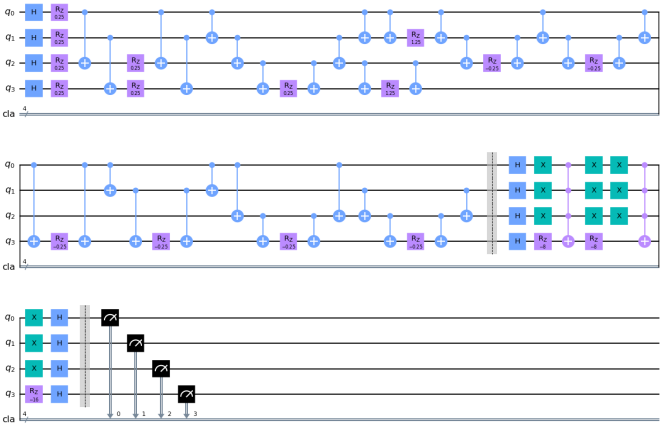


(a)

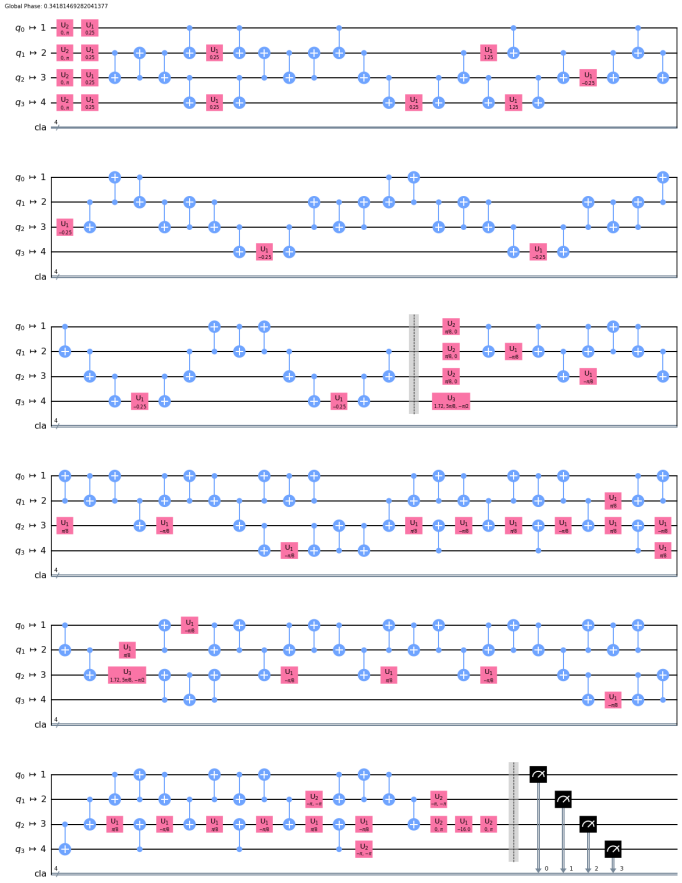


(b)

Figura A.5: (a) Circuito lógico para execução do VQF com QWOA para a instância $m = 2363$. (b) Circuito transpilado para execução do VQF com QWOA, para a instância $m = 2363$, no backend *FakeBoeblingenV2*. - Nos circuitos, os parâmetros β, γ e t foram inicializados com 1 para facilitar a visualização.



(a)



(b)

Figura A.6: (a) Circuito lógico para execução do VQF com QWOA para a instância $m = 56153$. (b) Circuito transpilado para execução do VQF com QWOA, para a instância $m = 56153$, no backend *FakeBoeblingenV2*. - Nos circuitos, os parâmetros β, γ e t foram inicializados com 1 para facilitar a visualização.

Apêndice B

Propriedades dos circuitos lógicos e transpilados para o VQF

Nas tabelas B.1-B.12, são apresentadas as propriedades dos circuitos lógicos e transpilados para o algoritmo VQF, para cada caso e instância testados, discutidas na seção 4.3.4.

Foi utilizado o módulo *Qiskit transpiler* da IBM [74] para transpilar os circuitos lógicos (utilizados nas simulações em ambiente ideal - backend *Aer*) para os circuitos adequados ao backend *FakeBoeblingenV2*, tanto em termos de mapa de acoplamento (conectividade) de qubits físicos, como de portas lógicas quânticas aceitas.

Por fim, ressalta-se, conforme também discutido na seção 4.3.4, que o *IBM Qiskit Transpiler* utiliza um algoritmo heurístico estocástico (*SabreSwap*) para realizar o mapeamento no circuito transpilado. Sendo assim, devido a essa natureza estocástica, resultados diferentes são obtidos a cada execução do módulo. Dessa forma, realizou-se o processo de transpilação 10 vezes, para cada caso e instância, e foi escolhido o circuito que apresentou o menor número de portas $CNOT$, buscando-se assim reduzir o número de portas two-qubit no circuito e, consequentemente o ruído por elas gerado.

Camadas	**Portas CNOT**	**Portas single-qubit**	**Profundidade do Circuito**
1	2	9	7
2	4	14	12
3	6	19	17
4	8	24	22
5	10	29	27
6	12	34	32
7	14	39	37
8	16	44	42

Tabela B.1: Propriedades do circuito lógico, utilizado na simulação em ambiente ideal, para a instância $m = 1981$, no caso com QAOA.

Camadas	Portas CNOT	Portas single-qubit	Profundidade do Circuito
1	2	9	7
2	4	14	12
3	6	19	17
4	8	24	22
5	10	29	27
6	12	34	32
7	14	39	37
8	16	44	42

Tabela B.2: Propriedades do circuito transpilado, utilizado na simulação ruidosa, para a instância $m = 1981$, no caso com QAOA. Foram utilizados os qubits físicos [0,1] para essa instância.

Camadas	Portas CNOT	Portas single-qubit	Profundidade do Circuito
1	6	15	14
2	12	24	26
3	18	33	38
4	24	42	50
5	30	51	62
6	36	60	74
7	42	69	86
8	48	78	98

Tabela B.3: Propriedades do circuito lógico, utilizado na simulação em ambiente ideal, para a instância $m = 2363$, no caso com QAOA.

Camadas	Portas CNOT	Portas single-qubit	Profundidade do Circuito
1	9	15	18
2	21	24	36
3	30	33	52
4	42	42	70
5	51	51	86
6	63	60	104
7	73	69	120
8	84	78	138

Tabela B.4: Propriedades do circuito transpilado, utilizado na simulação ruidosa, para a instância $m = 2363$, no caso com QAOA. Foram utilizados os qubits físicos [0,1,2] para essa instância.

Camadas	Portas CNOT	Portas single-qubit	Profundidade do Circuito
1	34	27	43
2	68	46	84
3	102	65	125
4	136	84	166
5	170	103	207
6	204	122	248
7	238	141	289
8	272	160	330

Tabela B.5: Propriedades do circuito lógico, utilizado na simulação em ambiente ideal, para a instância $m = 56153$, no caso com QAOA.

Camadas	Portas CNOT	Portas single-qubit	Profundidade do Circuito
1	50	27	56
2	100	46	110
3	150	65	164
4	200	84	218
5	250	103	272
6	300	122	326
7	350	141	380
8	400	160	434

Tabela B.6: Propriedades do circuito transpilado, utilizado na simulação ruidosa, para a instância $m = 56153$, no caso com QAOA. Foram utilizados os qubits físicos [0,1,2,3] para essa instância.

Camadas	Portas CNOT	Portas single-qubit	Portas MCX	Profundidade do Circuito
1	4	20	0	14
2	8	35	0	26
3	12	50	0	38
4	16	65	0	50
5	20	80	0	62
6	24	95	0	74
7	28	110	0	86
8	32	125	0	98

Tabela B.7: Propriedades do circuito lógico, utilizado na simulação em ambiente ideal, para a instância $m = 1981$, no caso com QWOA. - MCX indicam as portas NOT multi-controladas.

Camadas	Portas CNOT	Portas single-qubit	Portas MCX	Profundidade do Circuito
1	4	16	0	13
2	8	27	0	24
3	12	38	0	35
4	16	49	0	46
5	20	60	0	57
6	24	71	0	68
7	28	82	0	79
8	32	93	0	90

Tabela B.8: Propriedades do circuito transpilado, utilizado na simulação ruidosa, para a instância $m = 1981$, no caso com QWOA. Foram utilizados os qubits físicos [0,1] para essa instância. - MCX indicam as portas NOT multi-controladas.

Camadas	Portas CNOT	Portas single-qubit	Portas C2X	Profundidade do Circuito
1	6	31	2	21
2	12	55	4	40
3	18	79	6	59
4	24	103	8	78
5	30	127	10	97
6	36	151	12	116
7	42	175	14	135
8	48	199	16	154

Tabela B.9: Propriedades do circuito lógico, utilizado na simulação em ambiente ideal, para a instância $m = 2363$, no caso com QWOA. - C2X indicam as portas NOT 2-controladas.

Camadas	Portas CNOT	Portas single-qubit	Portas C2X	Profundidade do Circuito
1	33	41	0	55
2	63	74	0	110
3	90	107	0	168
4	124	140	0	221
5	150	173	0	278
6	186	206	0	332
7	213	239	0	382
8	239	272	0	434

Tabela B.10: Propriedades do circuito transpilado, utilizado na simulação ruidosa, para a instância $m = 2363$, no caso com QWOA. Foram utilizados os qubits físicos [0,1,2] para essa instância. - C2X indicam as portas NOT 2-controladas.

Camadas	Portas CNOT	Portas single-qubit	Portas C3X	Profundidade do Circuito
1	34	48	2	50
2	68	87	4	98
3	102	126	6	146
4	136	165	8	194
5	170	204	10	242
6	204	243	12	290
7	238	282	14	338
8	272	321	16	386

Tabela B.11: Propriedades do circuito lógico, utilizado na simulação em ambiente ideal, para a instância $m = 56153$, no caso com QWOA. - C3X indicam as portas NOT 3-controladas.

Camadas	**Portas CNOT**	**Portas single-qubit**	**Portas C3X**	**Profundidade do Circuito**
1	136	61	0	141
2	282	110	0	300
3	427	159	0	468
4	567	208	0	633
5	720	257	0	788
6	866	306	0	953
7	1012	355	0	1110
8	1157	404	0	1273

Tabela B.12: Propriedades do circuito transpilado, utilizado na simulação ruidosa, para a instância $m = 56153$, no caso com QWOA. Foram utilizados os qubits físicos [0,1,2,3] para essa instância. - C3X indicam as portas NOT 3-controladas.