UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ISRAEL FERRAZ DE ARAUJO

Otimização de Circuitos para a Inicialização de Estados Quânticos

Recife

2022

ISRAEL FERRAZ DE ARAUJO

Otimização de Circuitos para a Inicialização de Estados Quânticos

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Inteligência Computacional.

Orientador: Adenilton José da Silva

Coorientadora: Teresa Bernarda Ludermir

Recife

2022

Catalogação na fonte
Bibliotecária Luiza de Oliveira, CRB4-1316

A658o Araujo, Israel Ferraz de
Otimização de circuitos para a inicialização de estados quânticos. /Israel Ferraz de Araújo. – 2022.
128 f.: il., tab.

Orientador: Adenilton José da Silva.
Coorientador: Teresa Bernarda Ludermir.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIN, Ciência da Computação, Recife, 2022.

Inclui referências. Inclui apêndice.

1. Computação quântica. 2. Preparação de estados. 3. Otimização de circuitos. 4.Estados aproximados. I. Silva, Adenilton jose de (orientador). II. Ludermir, Teresa Bernarda (coorientadora). III Título.

006.31 CDD (23. ed.) UFPE - CCEN 2022-143

Israel Ferraz de Araujo

“Otimização de Circuitos para a Inicialização de Estados Quânticos”

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Inteligência Computacional.

Aprovado em: 01/09/2022.

_____

Orientador: Prof. Dr. Adenilton José da Silva

BANCA EXAMINADORA

_____

Prof. Dr. Stefan Michael Blawid
Centro de Informática / UFPE

_____

Prof. Dr. Antônio Murilo Santos Macedo
Departamento de Física / UFPE

_____

Prof. Dr. Wilson Rosa de Oliveira Junior
Departamento de Estatística e Informática / UFRPE

_____

Prof. Dr. Leon Denis da Silva
Departamento de Matemática / UFRPE

_____

Profa. Dra. Samuraí Gomes de Aguiar Brito
Itau Unibanco, Grupo de Tecnologias Emergentes

Dedico esta tese a toda minha família, amigos e professores. Seu suporte e confiança são fundamentais.

AGRADECIMENTOS

Aos colegas, pelo companheirismo, amizade e ajuda.

Aos professores, pelo empenho, comprometimento, compreensão e paciência.

Ao amigo e compadre Rui Ribeiro, pelo permanente apoio e incentivo.

À família, por ser presente, sempre.

À minha esposa Bruna e à minha filha, Ingrid, pelo amor e confiança incondicionais.

Ao professor Wilson de Oliveira, que influenciou minha vida profissional e pessoal com sua criatividade inspiradora.

À professora e coorientadora Teresa Ludermir, que me recebeu de braços abertos e apoiou em todos os momentos.

Ao professor e orientador Adenilton da Silva, por quem eu adquiri uma admiração profunda e cuja atenção e crença são importantes. Obrigado.

Este trabalho foi financiado pela Fundação de Amparo à Ciência e Tecnologia de Pernambuco (FACEPE), pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Capes).

No man is an island entire of itself; every man is a piece of the continent, a part of the main; if a clod be washed away by the sea, Europe is the less, as well as if a promontory were, as well as any manner of thy friends or of thine own were; any man's death diminishes me, because I am involved in mankind. And therefore never send to know for whom the bell tolls; it tolls for thee. (DONNE, 1624, p.108).

RESUMO

A teoria dos algoritmos quânticos promete benefícios sem precedentes ao utilizar as leis da mecânica quântica para resolver certos problemas computacionais. Entretanto, alguns desafios impedem que tais vantagens se apliquem a todos os algoritmos quânticos. Entre esses desafios está o desenvolvimento de uma codificação eficiente de dados clássicos em um estado quântico. Em aplicações práticas, o custo para carregar a informação clássica em um dispositivo quântico pode dominar o custo computacional assintótico do algoritmo. Foram propostos diversos métodos baseados em circuitos para codificar dados clássicos nas amplitudes de probabilidade de um estado quântico. Entretanto, esses métodos produzem circuitos com complexidade que cresce linearmente com o tamanho do dado, anulando o benefício da aplicação quântica. O objetivo principal deste trabalho é reduzir a complexidade computacional dos circuitos para a inicialização de estados quânticos. As duas estratégias utilizadas para alcançar tal objetivo exploram a troca de complexidade temporal por espacial ou a transferência de complexidade computacional para um computador clássico. A primeira estratégia reinterpreta métodos algébricos para a decomposição de estados quânticos como caminhadas ao longo de árvores binárias. A segunda investiga o grau de emaranhamento de bipartições do estado quântico através da decomposição de Schmidt. Os métodos baseados em árvore alcançam uma redução exponencial na profundidade dos circuitos, em comparação às abordagens existentes. Os baseados na decomposição de Schmidt produzem circuitos de complexidade variável, proporcional ao emaranhamento do estado. A perspectiva é que o resultado desta tese auxilie os profissionais da computação quântica durante a era dos dispositivos ruidosos de escala intermediária.

Palavras-chave: computação quântica; preparação de estados; otimização de circuitos; estados aproximados.

ABSTRACT

The theory of quantum algorithms promises unprecedented benefits by using the laws of quantum mechanics to solve some computational problems. However, challenges prevent such advantages from applying to all quantum algorithms. Among these challenges is the development of an efficient encoding of classical data into a quantum state. In practical applications, the cost to load classical information into a quantum device can dominate the asymptotic computational cost of the algorithm. Several circuit-based methods have been proposed to encode classical data into the probability amplitudes of a quantum state. However, these methods produce circuits with complexity that grows linearly with data size, negating the quantum application benefit. The main goal of this work is to reduce the computational complexity of quantum state initialization circuits. The two strategies to achieve this goal exploit exchanging temporal complexity for spatial complexity or transferring computational complexity to a classical computer. The first strategy reinterprets algebraic methods for quantum state decomposition as a walk along binary trees. The second investigates the degree of entanglement of quantum state bipartitions via Schmidt decomposition. The tree-based methods achieve an exponential reduction in circuit depth compared to existing approaches. Those based on Schmidt decomposition produce circuits of varying complexity, proportional to the entanglement of the state. The perspective is that the result of this thesis will assist quantum computing practitioners during the noisy intermediate-scale quantum era.

Keywords: quantum computing; state preparation; circuit optimization; approximate states.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

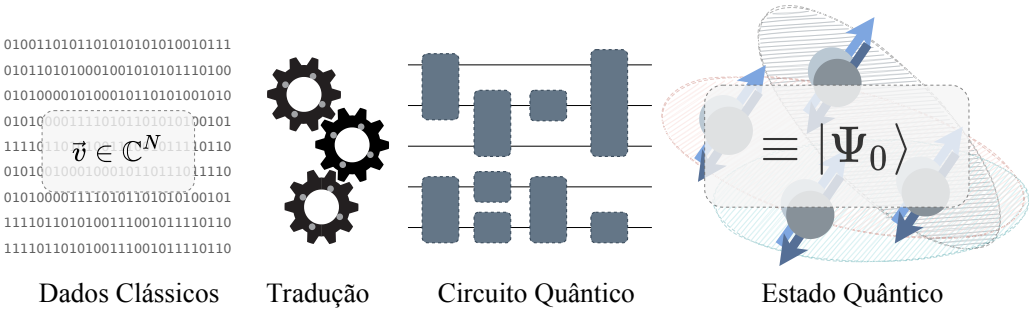
BAA	Bounded Approximation error state preparation Algorithm
BDSP	Bidirectional quantum State Preparation
CCD	Column–by–Column Decomposition
CSD	Cosine-Sine Decomposition
DCSP	Divide-and-Conquer quantum State Preparation
HHL	Aram Harrow, Avinatan Hassidim and Seth Lloyd
HOSVD	High Order Singular Value Decomposition
HTF	Hierarchical Tucker Format
LRSP	Low-Rank quantum State Preparation
MAE	Mean Absolute Error
MPS	Matrix Product State
NISQ	Noisy Intermediate-Scale Quantum
qGAN	Quantum Generative Adversarial Networks
QLA	Quantum Linear Algebra
QMem	Quantum Memory
QML	Quantum Machine Learning
QSD	Quantum Shannon Decomposition
SVD	Singular Value Decomposition
VQC	Variational Quantum Classifier
ZYZ	ZYZ decomposition

SUMÁRIO

1 INTRODUÇÃO

O desenvolvimento dos computadores quânticos pode reduzir drasticamente o tempo para resolver certas tarefas computacionais (ARUTE et al., 2019). O primeiro uso da computação quântica foi proposto por Feynman (1982), antes mesmo do primeiro modelo formalizado de computação quântica universal desenvolvido por Deutsch e Penrose (1985). Feynman sugeriu uma arquitetura computacional baseada em autômatos celulares com recursos quânticos e conjecturou que tal arquitetura poderia acelerar simulações de física e química quântica além da capacidade dos computadores clássicos. Além de Feynman, pioneiros como Benioff (1980) e Manin (1980) promoveram o desenvolvimento da computação quântica. Após pouco mais de uma década, Grover (1996) estabeleceu um algoritmo quântico de busca com aceleração quadrática em relação ao melhor algoritmo clássico conhecido. Em seguida, Shor (1999) demonstrou um algoritmo quântico de fatoração em tempo polinomial que pode ser utilizado para quebrar o sistema de criptografia RSA (GIDNEY; EKERÅ, 2021; RIVEST; SHAMIR; ADLEMAN, 1978).

Figura 1 – Processo de preparação de estados quânticos.

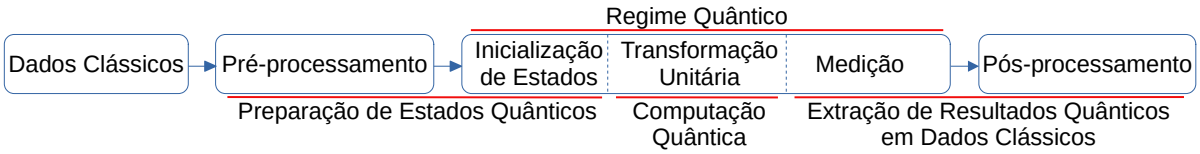


Fonte: Elaborada pelo autor (2022)

Recentemente, uma nova família de algoritmos quânticos promete ganho exponencial sobre os algoritmos clássicos equivalentes (AARONSON, 2015). O objetivo de tais algoritmos são problemas práticos como aprendizagem de máquina, clustering e solução de sistemas lineares (STOUDENMIRE; SCHWAB, 2016; SCHULD; FINGERHUTH; PETRUCCIONE, 2017; SCHULD; PETRUCCIONE, 2018; BENEDETTI et al., 2019; LEVINE et al., 2019; BLANK et al., 2020; TRUGENBERGER, 2001; TRUGENBERGER, 2002; GIOVANNETTI; LLOYD; MACCONE, 2008b; SILVA; OLIVEIRA; LUDERMIR, 2010; de Paula Neto et al., 2019; PARK; PETRUCCIONE; RHEE, 2019; HARROW; HASSIDIM; LLOYD, 2009; LLOYD; MOHSENI; REBENTROST, 2014; CHILDS; KOTHARI; SOMMA, 2017; WOSSNIG; ZHAO; PRAKASH, 2018; REBENTROST et al., 2018). Vantagens em vários cam-

pos de pesquisa e indústria são esperadas com o surgimento dessas aplicações quânticas. Mas algumas dificuldades impedem que tais aplicações sejam realizadas (AARONSON, 2015; BIAMONTE et al., 2017; LEYMANN; BARZEN, 2020).

Figura 2 – Computação quântica com entrada clássica.



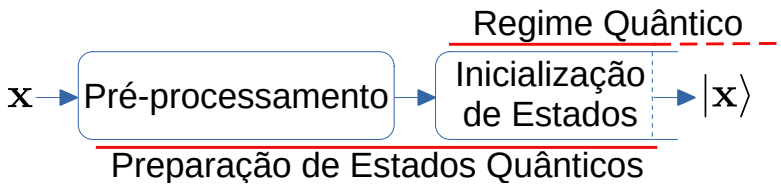
Fonte: Elaborada pelo autor (2022)

Um dos problemas em aberto que impedem aplicações práticas da computação quântica é o desenvolvimento de uma codificação eficiente de dados clássicos em um estado quântico (BIAMONTE et al., 2017; AARONSON, 2015; LEYMANN; BARZEN, 2020; TANG, 2021). Para efetuar essa codificação, o dado clássico é convertido para uma forma vetorial e um processo traduz a informação em operações realizadas em um computador quântico, comumente codificadas por um circuito quântico (Figura 1) (NIELSEN; CHUANG, 2010). A execução desse circuito cria o estado inicial pretendido. Vários métodos baseados em circuitos quânticos foram propostos para codificar esses dados como amplitudes de probabilidade de um estado quântico (VENTURA; MARTINEZ, 1999; GROVER, 2000; LONG; SUN, 2001; MOTTONEN et al., 2005; SHENDE; BULLOCK; MARKOV, 2006; PLESCH; BRUKNER, 2011; CORTESE; BRAJE, 2018). Entretanto, eles requerem profundidade do circuito quântico – caminho mais longo do circuito, representado pelo número de operadores nesse caminho – que cresce linearmente com o tamanho dos dados.

O HHL, criado por Harrow, Hassidim e Lloyd (2009), é um exemplo de algoritmo quântico que sofre de tal problema. Seu objetivo é solucionar o sistema linear $A\mathbf{x} = \mathbf{b}$. Para isso, o HHL expressa o vetor $\mathbf{b} \in \mathbb{C}^N$ como o estado quântico $|b\rangle$ usando $\log_2(N)$ qubits e o vetor $\mathbf{x}$ como o estado quântico $|x\rangle$. Dessa maneira, temos a equação $A|x\rangle = |b\rangle$, que pode ser solucionada multiplicando ambos os lados pela inversa A^{-1}. Para encontrar $|x\rangle$, o algoritmo HHL produz um circuito quântico com profundidade $O(\log_2^2(N))$, enquanto o algoritmo clássico precisa de $O(N \log_2(N))$ passos, caracterizando a vantagem quântica exponencial. Mas há ressalvas importantes quanto ao HHL (AARONSON, 2015). Uma delas é que o vetor $\mathbf{b}$ precisa ser codificado rapidamente na memória do computador quântico, preparando o estado $|b\rangle = \sum_{i=0}^{N-1} b_i |i\rangle$. Preparar esse estado utilizando qualquer método existente requer custo na ordem de N passos, anulando a vantagem exponencial do HHL em seu princípio.

Na computação quântica, a etapa de codificação dos dados clássicos em um estado quân-

Figura 3 – Mapeamento do vetor de entrada clássico em um estado quântico.



Fonte: Elaborada pelo autor (2022)

tico é chamada de *Preparação de Estados Quânticos* (Figura 2 e Figura 3) (SCHULD; PETRUCCIONE, 2018). O foco desta tese é atuar na preparação de estados quânticos para produzir circuitos eficientes para a inicialização de estados. Essa etapa recebe como entrada um vetor de dados clássico e produz como saída um estado quântico (Figura 3) que será utilizado pela aplicação quântica na etapa seguinte (*Computação Quântica*, Figura 2). A preparação de estados é dividida em duas fases chamadas *Pré-processamento* e *Inicialização de Estados* (Figura 2 e Figura 3). A primeira fase ocorre no regime clássico e tem como objetivo construir um circuito quântico com a sequência de instruções, na forma de operadores, para a produção do estado. A segunda fase ocorre no regime quântico e executa as instruções do circuito criado no passo anterior para produzir efetivamente o estado pretendido. As complexidades de ambas as fases devem ser consideradas para estimar o esforço necessário para produzir o estado quântico.

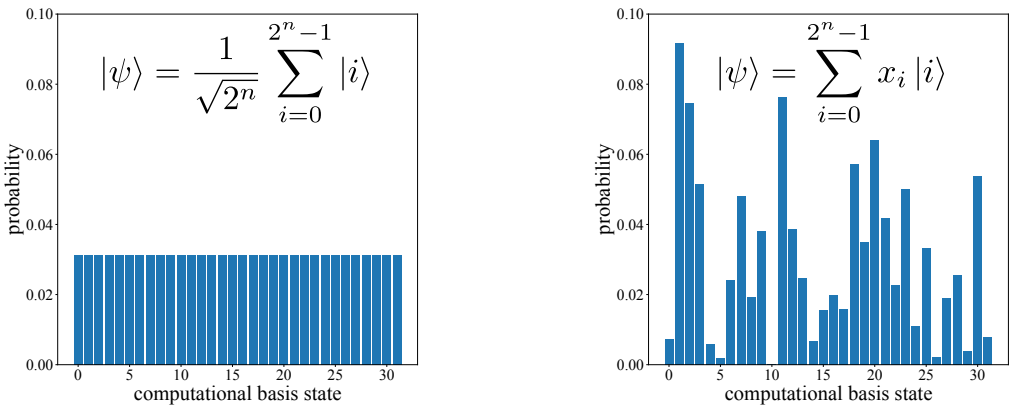
O restante deste capítulo está organizado da seguinte forma. A Seção 1.1 apresenta a motivação para a busca de uma preparação eficiente de estados quânticos. A Seção 1.2 descreve os objetivos desta tese. O Capítulo 2 revisa os principais fundamentos utilizados no desenvolvimento deste trabalho. O Capítulo 3 comenta algumas soluções existentes para a preparação de estados quânticos. O Capítulo 4 resume as contribuições desta tese, incluindo a biblioteca de código aberto qclib (ARAUJO et al., 2022) usada para produzir os gráficos e tabelas deste trabalho. Por fim, o Capítulo 5 apresenta a conclusão e os principais resultados. O detalhamento de todos os resultados e contribuições está nos apêndices A, B, C e D. O Apêndice E discute a relação entre emaranhamento e a preparação de estados quânticos.

1.1 MOTIVAÇÃO

1.1.1 Complexidade computacional da preparação de estados quânticos

O pior caso de complexidade na preparação de um estado quântico arbitrário (dados não estruturados) é exponencial no número de qubits (SHENDE; BULLOCK; MARKOV, 2006). Por este motivo, as vantagens quânticas (BIAMONTE et al., 2017) mais significativas ocorrem quando o algoritmo quântico opera em um estado de entrada que é fácil de preparar (DEUTSCH; JOZSA, 1992; HOGG; HUBERMAN; WILLIAMS, 1996; GROVER, 1996; SIMON, 1997; TERHAL; SMOLIN, 1998; SHOR, 1999), tais como a superposição uniforme de todos os estados da base computacional (MOZAFARI et al., 2020; MOZAFARI et al., 2021), os estados esparsos (MALVETTI; ITEN; COLBECK, 2021; VERAS; SILVA; SILVA, 2022; GLEINIG; HOEFLER, 2021) e as distribuições de probabilidade (ZOUFAL; LUCCHI; WOERNER, 2019; NAKAJI et al., 2021; MARIN-SANCHEZ; GONZALEZ-CONDE; SANZ, 2021).

Figura 4 – Representação de estados quânticos uniforme e arbitrário.



Fonte: Elaborada pelo autor (2022)

Para algoritmos que dependam do carregamento de dados em um estado quântico arbitrário, uma maneira eficiente de preparar os estados de entrada é um pré-requisito fundamental para a vantagem quântica (RøNNOW et al., 2014; AARONSON, 2015; BIAMONTE et al., 2017; HARROW; HASSIDIM; LLOYD, 2009; TANG, 2021). Várias pesquisas se concentram no desenvolvimento de algoritmos que supõem uma inicialização eficiente dos dados. Tal suposição é o que leva às declarações de vantagens exponenciais anunciadas por diversos trabalhos (WIEBE; BRAUN; LLOYD, 2012; LLOYD; MOHSENI; REBENTROST, 2014; REBENTROST; MOHSENI; LLOYD, 2014; WIEBE; KAPOOR; SVORE, 2014; AMIN et al., 2018; DUNJKO; TAYLOR; BRIEGEL,

2016; KIEFEROVÁ; WIEBE, 2017; LOW; YODER; CHUANG, 2014; WIEBE; GRANDADE, 2017; WIEBE; KAPOOR; SVORE, 2016).

1.1.2 Preparação de estados quânticos em dispositivos ruidosos

A solução para o problema da complexidade na preparação de estados torna-se ainda mais relevante sob as limitações do hardware quântico ruidoso de escala intermediária (NISQ, do inglês *Noisy Intermediate-Scale Quantum*) (MOLL et al., 2018; PRESKILL, 2018) como número de qubits limitado, conectividade restrita entre os qubits, erros de portas e decoerência (WILLSCH et al., 2017). Tais limitações impedem que técnicas de correção de erros sejam aplicadas e também restringem o uso de tais dispositivos a aplicações com circuitos de baixa profundidade. Uma alternativa é a produção de soluções adaptadas às limitações dos dispositivos NISQ.

Por causa do ruído e do número limitado de qubits, a profundidade de um algoritmo quântico – correspondente à profundidade do circuito quântico produzido pelo algoritmo – e também sua largura – número de qubits do circuito – precisam ser controladas. A Equação (1.1) pode ser usada para determinar os limites para a execução de um algoritmo num determinado dispositivo quântico (LEYMANN; BARZEN, 2020)

$$d \cdot w \ll \frac{1}{\epsilon} \tag{1.1}$$

onde ϵ é a taxa de erro do computador quântico, d a profundidade do algoritmo, e w a largura (WILLSCH et al., 2017). Portanto, se um algoritmo requer profundidade grande, deve usar apenas poucos qubits e assim pode ser simulado num computador clássico. Mas se um algoritmo requer muitos qubits, precisa ser raso. Isso implica que a vantagem quântica – soluções que não podem ser simuladas em computadores clássicos com espaço/tempo viável (PRESKILL, 2012; PRESKILL, 2018) – em dispositivos sem correção de erros tem que ser exibida por algoritmos rasos, que possuam poucas camadas de portas.

1.2 OBJETIVOS

O objetivo principal desta tese é reduzir a complexidade computacional de circuitos – profundidade e largura – para a inicialização de estados quânticos. Os objetivos específicos são:

1. Trocar profundidade do circuito quântico por largura para reduzir a complexidade temporal da inicialização de estados exatos (Divide-and-Conquer quantum State Preparation, Seção 4.1 e Bidirectional quantum State Preparation, Seção 4.2).

2. Utilizar o emaranhamento como recurso computacional para otimizar os circuitos quânticos produzidos para a inicialização de estados exatos, reduzindo a quantidade de operações necessárias (Low-Rank quantum State Preparation, Seção 4.3).

3. Manipular o emaranhamento para produzir estados aproximados de menor complexidade, otimizados para a inicialização em dispositivos NISQ (Bounded Approximation error state preparation Algorithm, Seção 4.3).

2 PRINCIPAIS FUNDAMENTOS E REFERÊNCIAS

2.1 DECOMPOSIÇÃO DE UNITÁRIOS

A decomposição de unitários permite que os algoritmos quânticos, baseados em circuitos, especifiquem operadores arbitrários que podem ser executados nos computadores quânticos. Sem essa decomposição, os algoritmos quânticos teriam que ser construídos utilizando apenas o conjunto universal de portas do dispositivo alvo (DIVINCENZO, 1995; BARENCO, 1995; LLOYD, 1995; DEUTSCH; BARENCO; EKERT, 1995).

2.1.1 Unitários de um qubit

Um operador arbitrário de um qubit pode ser decomposto como uma sequência de no máximo três portas R_y e R_z (BARENCO et al., 1995; NIELSEN; CHUANG, 2010) usando a decomposição ZYZ[1] (SHENDE; MARKOV; BULLOCK, 2004a). Dada qualquer matriz unitária $U_{2\times2}$, existem ângulos ϕ, α, β, e γ que satisfazem a seguinte equação:

$$U = e^{i\phi} R_z(\alpha) R_y(\beta) R_z(\gamma). \tag{2.1}$$

Tal decomposição é possível porque as linhas e colunas de matrizes unitárias são vetores ortonormais, portanto toda matriz unitária 2×2 pode ser escrita na forma (BARENCO et al., 1995; NIELSEN; CHUANG, 2010)

$$\begin{bmatrix} e^{i(\phi+\frac{\alpha}{2}+\frac{\gamma}{2})}\cos\frac{\beta}{2} & e^{i(\phi+\frac{\alpha}{2}-\frac{\gamma}{2})}\sin\frac{\beta}{2} \\ -e^{i(\phi-\frac{\alpha}{2}+\frac{\gamma}{2})}\sin\frac{\beta}{2} & e^{i(\phi-\frac{\alpha}{2}-\frac{\gamma}{2})}\cos\frac{\beta}{2} \end{bmatrix}. \tag{2.2}$$

A partir da matriz (2.2), a fatoração na forma da expressão (2.1) é imediata.

Para o caso em que $U_{2\times2}$ pertence ao grupo especial unitário $SU(2)$, a propriedade $\det(U) = 1$ implica que $e^{i\phi} = \pm 1$. Assim, a expressão (2.1) pode ser simplificada para:

$$U = R_z(\alpha) R_y(\beta) R_z(\gamma). \tag{2.3}$$

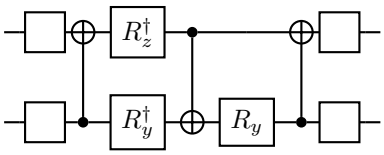
[1] A escolha dos eixos y e z é arbitrária. Qualquer par de vetores ortogonais pode ser escolhido.

2.1.2 Unitários de dois qubits

A síntese de operadores de dois qubits é relevante para a otimização de circuitos maiores. A tecnologia atual dos dispositivos quânticos não é capaz de executar operações que atuem em três ou mais qubits. Tais operações precisam ser decompostas utilizando portas menores. Portanto, implementar uma computação quântica como uma sequência de portas de dois qubits é de fundamental importância. Por sua vez, as portas arbitrárias de dois qubits precisam ser decompostas usando a porta padrão de dois qubits do conjunto de portas nativas do dispositivo alvo, além de portas de um qubit (DIVINCENZO, 2000). O CNOT é utilizado como o operador padrão de dois qubits na maioria dos trabalhos teóricos e práticos (WILLIAMS, 2011), mas sua implementação é ordens de magnitude ($\sim 10^2$) mais propensa a erros do que a implementação das portas de um qubit. Por esse motivo, uma forma de calcular o custo da execução dos circuitos em dispositivos quânticos é em termos do número de CNOTs. Sendo assim, o custo da decomposição de operadores de dois qubits precisa ser minimizado, pois tem impacto direto no custo das aplicações quânticas.

De maneira geral, todo operador de dois qubits pode ser representado pelo circuito da Figura 5 utilizando dezoito portas elementares – três CNOTs e quinze rotações de um qubit (SHENDE; MARKOV; BULLOCK, 2004a; SHENDE; MARKOV; BULLOCK, 2004b).

Figura 5 – Decomposição de unitários arbitrários de dois qubits.



Fonte: Elaborada pelo autor (2022)

Para demonstrar que essa decomposição é possível, começamos reescrevendo $U \in U(4)$ como

$$U' = e^{i\frac{\pi}{4}}\mathsf{SWAP}_1^2 U \in SU(4). \tag{2.4}$$

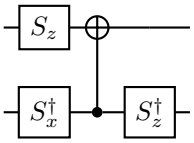
A *decomposição canônica* de $SU(4)$ (KHANEJA; BROCKETT; GLASER, 2001) afirma que existe a, b, c e $d \in SU(2)$ e a diagonal δ na *base mágica* (HILL; WOOTTERS, 1997; KRAUS; CIRAC, 2001) de maneira que

$$U' = (a \otimes b)\delta(c \otimes d). \tag{2.5}$$

Podemos reescrever a expressão (2.5) como $U' = (a \otimes b)E\Delta E^\dagger(c \otimes d)$, onde Δ é uma diagonal na base computacional, usando a matriz E de mudança de base (operador de dois

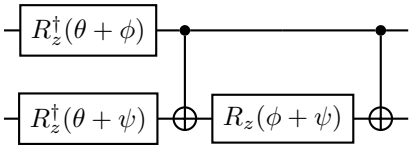
qubits que mapeia a base computacional para a base mágica) (KHANEJA; BROCKETT; GLASER, 2001) e definindo $S_m \coloneqq R_m(\pi/2)$ $(m \in \{x, y, z\})$

$$E = \frac{\sqrt{2}}{2}\begin{bmatrix} 1 & i & 0 & 0 \\ 0 & 0 & i & 1 \\ 0 & 0 & i & -1 \\ 1 & -i & 0 & 0 \end{bmatrix} = \tag{2.6}$$



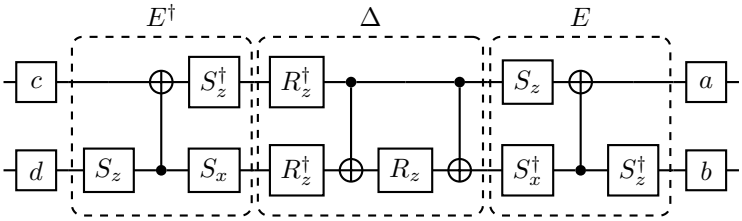
Sabendo que toda diagonal $SU(4)$ pode ser representada como (KHANEJA; BROCKETT; GLASER, 2001)

$$\Delta = \begin{bmatrix} e^{i\theta} & 0 & 0 & 0 \\ 0 & e^{i\phi} & 0 & 0 \\ 0 & 0 & e^{i\psi} & 0 \\ 0 & 0 & 0 & e^{-i(\theta+\phi+\psi)} \end{bmatrix} = \tag{2.7}$$



construímos o circuito da Figura 6 para U':

Figura 6 – Decomposição de unitários arbitrários de dois qubits sem simplificações.



Fonte: Elaborada pelo autor (2022)

O circuito acima pode ser simplificado fazendo $dS_z = d'$, $S_z^\dagger b = b'$, $S_z^\dagger R_z^\dagger S_z = R_z^\dagger$, $S_x R_z^\dagger = R_y^\dagger S_x$ e $R_z S_x^\dagger = S_x^\dagger R_y$ $(\vec{n} \perp \vec{m} \rightarrow S_n R_m = R_{n\times m} S_n)$, e cancelando S_x e $S_x^\dagger$ nas duas últimas expressões, como na Figura 7.

Figura 7 – Decomposição de unitários arbitrários de dois qubits com simplificações.

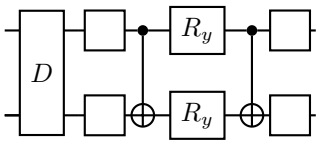
Fonte: Elaborada pelo autor (2022)

Ainda na Figura 7, os dois CNOTs adjacentes são substituídos por um CNOT e um SWAP. O SWAP foi movido para o fim do circuito e será cancelado pelo SWAP da expressão (2.4).

Portanto, o circuito da Figura 5 representa o unitário U, com exceção apenas da fase global $e^{i\frac{\pi}{4}}$ que precisa ser corrigida.

Ainda é possível alcançar um circuito mais otimizado que o da Figura 5 para os casos em que vários operadores adjacentes atuam no mesmo conjunto de qubits. Um operador $U'' \in SU(4)$ pode ser representado usando apenas dois CNOTs e portas de um qubit $SU(2)$ arbitrárias se $\mathrm{tr}\left[\gamma(U'')\right]$ é real (SHENDE; BULLOCK; MARKOV, 2004c), onde $\gamma(U'') = (E^\dagger U'' E)(E^\dagger U'' E)^T$. Sabendo que o operador arbitrário U' pode ser decomposto como $U' = DU''$, onde D é um operador diagonal, temos o circuito da Figura 8 (SHENDE; BULLOCK; MARKOV, 2006).

Figura 8 – Decomposição de unitários arbitrários de dois qubits até uma diagonal.



Fonte: Elaborada pelo autor (2022)

Dessa maneira, a diagonal pode migrar para o próximo operador de dois qubits adjacente. Esse outro operador é decomposto da mesma forma, continuando o processo de migrar a diagonal até restar apenas um operador. Como um CNOT é salvo para cada operador de dois qubits menos o último, temos uma redução total de $M - 1$ CNOTs em relação à decomposição da Figura 5, onde M é o número total de operadores adjacentes.

2.1.3 Decomposição Cosseno-Seno

A decomposição cosseno-seno de unitários (*Cosine-Sine Decomposition* ou CSD) é baseada na técnica matemática de mesmo nome *decomposição cosseno-seno* (STEWART; SUN, 1990; PAIGE; WEI, 1994). Supondo um unitário $U \in \mathbb{C}^{N\times N}$, onde N é par, o teorema CSD afirma que U sempre pode ser decomposto na seguinte forma

$$U = RD_{cs}L = \begin{bmatrix} R_0 & 0 \\ 0 & R_1 \end{bmatrix} \begin{bmatrix} C & -S \\ S & C \end{bmatrix} \begin{bmatrix} L_0 & 0 \\ 0 & L_1 \end{bmatrix}, \tag{2.8}$$

composta por unitários $L_i, R_i \in \mathbb{C}^{N/2\times N/2}$ e matrizes diagonais reais C e S tais que $C^2 + S^2 = \mathbb{I}_{N/2}$, onde $C = \mathrm{diag}(\cos(\theta_0), \ldots, \cos(\theta_{N/2}))$ e $S = \mathrm{diag}(\sin(\theta_0), \ldots, \sin(\theta_{N/2}))$.

O algoritmo CSD (TUCCI, 1999; SHENDE; BULLOCK; MARKOV, 2006) aplica a decomposição cosseno-seno recursivamente. Inicia com o operador U de dimensão $N \times N$, onde $N = 2^n$ e

$n \geq 1$ é o número de qubits. U é decomposto na matriz D_{cs}, dois unitários L_i e dois unitários R_i (Equação 2.8 e Figura 9). Em seguida, os quatro unitários produzidos no passo anterior são decompostos em quatro matrizes centrais e dezesseis unitários, como na Equação (2.9). Os unitários formam quatro conjuntos $R_{R_0} + R_{R_1}$, $L_{R_0} + L_{R_1}$, $R_{L_0} + R_{L_1}$ e $L_{L_0} + L_{L_1}$ (Figura 10), cada conjunto com quatro operadores.

$$
\begin{aligned}
U = &\begin{bmatrix}
\begin{bmatrix} R_{R_0,0} & 0 \\ 0 & R_{R_0,1} \end{bmatrix} \begin{bmatrix} C_{R_0} & -S_{R_0} \\ S_{R_0} & C_{R_0} \end{bmatrix} \begin{bmatrix} L_{R_0,0} & 0 \\ 0 & L_{R_0,1} \end{bmatrix} & 0 \\
0 & \begin{bmatrix} R_{R_1,0} & 0 \\ 0 & R_{R_1,1} \end{bmatrix} \begin{bmatrix} C_{R_1} & -S_{R_1} \\ S_{R_1} & C_{R_1} \end{bmatrix} \begin{bmatrix} L_{R_1,0} & 0 \\ 0 & L_{R_1,1} \end{bmatrix}
\end{bmatrix} \\
&\begin{bmatrix} C & -S \\ S & C \end{bmatrix} \\
&\begin{bmatrix}
\begin{bmatrix} R_{L_0,0} & 0 \\ 0 & R_{L_0,1} \end{bmatrix} \begin{bmatrix} C_{L_0} & -S_{L_0} \\ S_{L_0} & C_{L_0} \end{bmatrix} \begin{bmatrix} L_{L_0,0} & 0 \\ 0 & L_{L_0,1} \end{bmatrix} & 0 \\
0 & \begin{bmatrix} R_{L_1,0} & 0 \\ 0 & R_{L_1,1} \end{bmatrix} \begin{bmatrix} C_{L_1} & -S_{L_1} \\ S_{L_1} & C_{L_1} \end{bmatrix} \begin{bmatrix} L_{L_1,0} & 0 \\ 0 & L_{L_1,1} \end{bmatrix}
\end{bmatrix}.
\end{aligned}
\tag{2.9}
$$

A recursão continua até que os unitários possuam dimensão 2×2 (operadores de um qubit). No fim, após $n-1$ iterações, haverá 2^{n-1} conjuntos de unitários, cada um com 2^{n-1} operadores de um qubit. Cada conjunto corresponde a um operador uniformemente controlado pelos qubits mais significativos e aplicado ao qubit menos significativo (Figura 9) (SHENDE; BULLOCK; MARKOV, 2006). Esses conjuntos são separados pelos operadores centrais (Figura 10), que

Figura 9 – Identidades utilizadas para a decomposição cosseno-seno (CSD) de operadores quânticos.

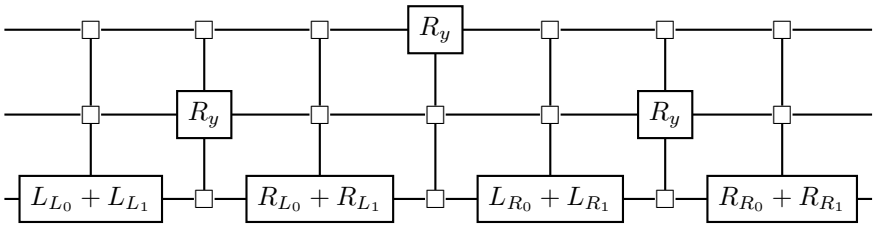
Fonte: Elaborada pelo autor (2022)

possuem a mesma estrutura das rotações R_y uniformemente controladas aplicadas ao qubit correspondente ao passo da recursão (SHENDE; BULLOCK; MARKOV, 2006). Essa decomposição requer um total de $4^n - 2^{n+1}$ CNOTs.

2.1.4 Decomposição de Shannon Quântica

O algoritmo para a decomposição de Shannon Quântica (Quantum Shannon Decomposition ou QSD) é uma variação do algoritmo CSD apresentado na Seção 2.1.3. A mudança é

Figura 10 – Circuito para a decomposição cosseno-seno (CSD) de um operador quântico de três qubits.



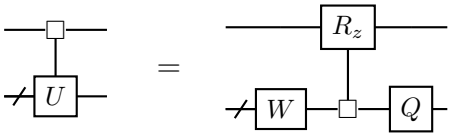
Fonte: Elaborada pelo autor (2022)

feita na decomposição dos operadores L e R uniformemente controlados na primeira identidade da Figura 9. Sendo $U = U_0 \oplus U_1$ (assim como as matrizes L e R do CSD), podemos encontrar os unitários Q e W e a diagonal D que satisfazem (SHENDE; BULLOCK; MARKOV, 2006)

$$U = (I \otimes Q)(D \oplus D^\dagger)(I \otimes W). \tag{2.10}$$

A partir da Equação (2.10), temos que $U_0 = QDW$ e $U_1 = QD^\dagger W$, o que implica $U_0 U_1^\dagger = QD^2Q^\dagger$. Temos então que D e Q podem ser calculados através de diagonalização. O unitário W pode ser encontrado usando as relações $W = D^\dagger Q^\dagger U_0$ ou $W = DQ^\dagger U_1$. Sabendo que a diagonal D é composta pelos autovalores da diagonalização, a matriz $D \oplus D^\dagger$ pode ser representada por uma rotação R_z uniformemente controlada aplicada ao qubit mais significativo (Figura 11) (SHENDE; BULLOCK; MARKOV, 2006).

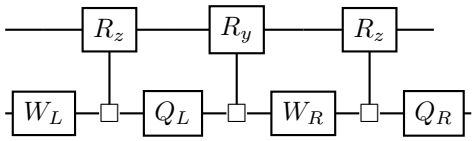
Figura 11 – Identidade utilizada para a decomposição de Shannon (QSD) de operadores quânticos.



Fonte: Elaborada pelo autor (2022)

De maneira semelhante ao algoritmo CSD, o QSD aplica as identidades definidas na Figura 9 e na Figura 11 recursivamente. Inicia aplicando a primeira identidade da Figura 9 para decompor o operador $U \in \mathbb{C}^{N \times N}$ em uma matriz central D_{cs} e unitários L_i e R_i. Em seguida, a identidade da Figura 11 é aplicada aos operadores L e R, decompondo-os na diagonal $D \oplus D^\dagger$ e nos unitários W e Q de dimensão $N/2 \times N/2$ (Figura 12). O procedimento é reiniciado a partir da primeira identidade, atuando sobre os operadores W e Q, e continua até que esses operadores possuam dimensão 2×2 representando portas de um qubit. A decomposição ZYZ utilizada para representar os operadores de um qubit não possuem CNOTs, portanto apenas as rotações uniformemente controladas contribuem para o número total de CNOTs no circuito $\frac{3}{4}4^n - \frac{3}{2}2^n$ (SHENDE; BULLOCK; MARKOV, 2006).

Figura 12 – Circuito para a decomposição de Shannon (QSD) de um operador quântico de dois qubits.



Fonte: Elaborada pelo autor (2022)

Se a recursão encerrar quando os operadores Q e W atuarem sobre dois qubits, haverá 4^{n-2} operadores desse tipo e menos $3 \times 4^{n-2}$ rotações uniformemente controladas por um qubit. Utilizando a decomposição da Seção 2.1.2, cada operador de dois qubits contribui com 3 CNOTs. Cada rotação uniformemente controlada atuando sobre dois qubits contribui com 2 CNOTs (ITEN et al., 2016). Portanto, a contagem de CNOTs diminui $3 \times 4^{n-2}$ e o total é reduzido para $\frac{9}{16}4^n - \frac{3}{2}2^n$. Essa variante do algoritmo foi chamada de QSD ($l = 2$) por Shende, Bullock e Markov (2006), onde l indica o número de qubits em que a recursão encerra. O caso anterior, que encerra em operadores de um qubit, chama-se QSD ($l = 1$).

O QSD ($l = 2$) ainda pode ser otimizado de duas maneiras (SHENDE; BULLOCK; MARKOV, 2006). A primeira, quando a recursão encerra em operadores V e W atuando sobre dois qubits, aplica a otimização indicada pela Figura 8 da Seção 2.1.2, reduzindo $4^{n-2} - 1$ CNOTs (um CNOT por operador de dois qubits, com exceção do último). A segunda otimização utiliza portas CZ (*Controlled-Z*) no lugar de CNOTs para decompor as matrizes centrais, pois o último CZ pode ser absorvido pelo multiplexador vizinho, salvando $(4^{n-2} - 1)/3$ CNOTs. Com ambas as otimizações, o número total de CNOTs é reduzido para $\frac{23}{48}4^n - \frac{3}{2}2^n + \frac{4}{3}$. Até o momento, essa é a decomposição mais eficiente para unitários gerais. Shende, Bullock e Markov (2006) chamou essa decomposição otimizada de QSD ($l = 2$, otimizada).

2.2 TIPO DE PREPARAÇÃO DE ESTADOS QUÂNTICOS

Para codificar um vetor de entrada clássico em um estado quântico é necessário construir um circuito quântico que prepare tal estado (CORTESE; BRAJE, 2018). Esse circuito é gerado numa fase de pré-processamento clássico aplicando instruções quânticas do dispositivo alvo. As fases de pré-processamento clássico e de inicialização do estado quântico combinadas – chamadas de *preparação de estados quânticos* (Figura 2 e Figura 3) – formam o principal objeto de estudo desta pesquisa.

2.2.1 Codificação na base

Essa codificação transforma o número binário $x \in \{0,1\}^n$ em um estado quântico na base computacional. O circuito quântico correspondente é gerado pela aplicação de portas X no qubit i quando $x^i = 1$. A execução desse circuito produzirá o estado quântico representando o número binário x.

O método é utilizado de diversas formas. Por exemplo, um vetor $\mathbf{x} = (x_1, \ldots, x_M)$, onde $1 \leq M \leq 2^n$, pode ser codificado através da transformação de cada componente x_j em estados da base e da concatenação das codificações resultantes. De maneira mais sofisticada, podemos representar o vetor $\mathbf{x}$ numa superposição uniforme dos estados dos seus componentes codificados binariamente (VENTURA; MARTINEZ, 2000; SCHULD; PETRUCCIONE, 2018)

$$|\mathbf{x}\rangle = \frac{1}{\sqrt{M}} \sum_{j=1}^{M} |x_j\rangle \,. \tag{2.11}$$

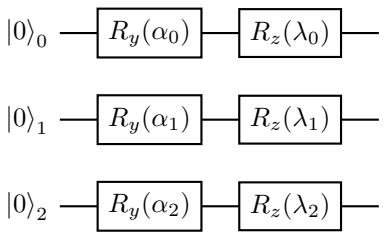
2.2.2 Codificação nos ângulos

Essa codificação mapeia cada componente x_j do vetor $\mathbf{x} \in \mathbb{C}^N$ como rotações R_y e R_z de qubits individuais (Figura 13). O procedimento consiste em redimensionar o módulo de cada elemento vetorial de dados x_j dentro da faixa $[0, \pi]$ para depois codificá-lo em um qubit q_j. Dados $\alpha_j = \text{abs}(x_j)$ e $\lambda_j = \arg(x_j)$, temos para cada qubit

$$|q_j\rangle = \cos\left(\frac{\alpha_j}{2}\right)|0\rangle + e^{i\lambda_j}\sin\left(\frac{\alpha_j}{2}\right)|1\rangle \,.$$

É menos eficiente em termos de espaço comparado à codificação nas amplitudes, exigindo N qubits para codificar um vetor de dados N-dimensional (STOUDENMIRE; SCHWAB, 2016). Mas a preparação do estado é simples, pois requer apenas rotações de um qubit.

Figura 13 – Codificação em ângulos de qubits utilizando rotações.



Fonte: Elaborada pelo autor (2022)

2.2.3 Codificação nas amplitudes

A codificação do vetor $\mathbf{x} \in \mathbb{C}^N$ ($|\mathbf{x}| = 1$) nas amplitudes de um estado quântico é representada por $|\mathbf{x}\rangle = \sum_j x_j |j\rangle$. Essa codificação requer $n = \lceil \log_2(N) \rceil$ qubits para representar um dado N-dimensional.

Nossa pesquisa e as preparações de estados baseadas em rotações (Seção 2.3.1) e na decomposição de Schmidt (Seção 2.3.2) fazem parte desta abordagem. Codificações deste tipo conseguem preparar $|\mathbf{x}\rangle$ a partir do estado $|0\rangle$ usando $O(2^n)$ portas (MOTTONEN et al., 2005; BERGHOLM et al., 2005; SHENDE; BULLOCK; MARKOV, 2006). Portanto, ela é vantajosa em relação às outras preparações de estados quando o dispositivo quântico consegue executar circuitos profundos e quando a dimensionalidade do vetor de entrada é muito alta, tornando impraticável utilizar outro tipo de preparação de estados devido ao número de qubits.

2.3 EXEMPLO DE PREPARAÇÃO DE ESTADOS QUÂNTICOS

2.3.1 Baseada em rotações

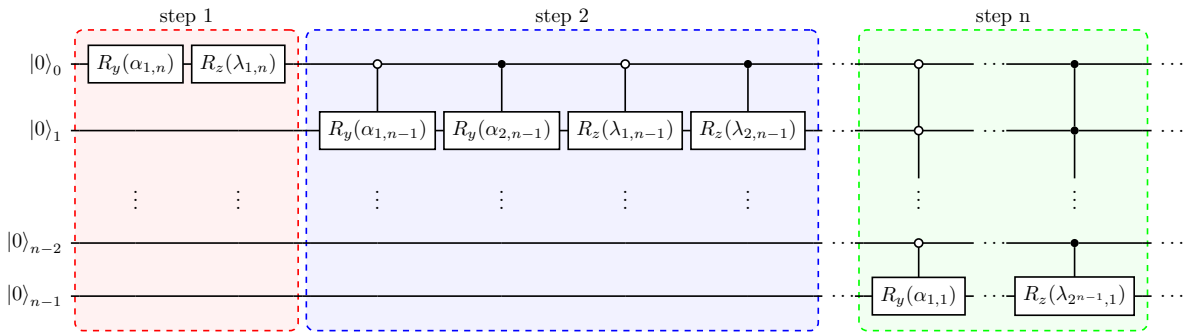
A preparação de estados proposta por Mottonen et al. (2005) descreve o problema como a busca por um operador U que transforma um vetor de estado arbitrário $|x\rangle = \sum_p |x_p| e^{iw_p} |p\rangle$ em um vetor fixo $|r\rangle$. Por conveniência, o vetor fixo é escolhido como o primeiro vetor da base computacional $|r\rangle = |0\rangle$. Conhecendo U de maneira que $U|x\rangle = |0\rangle$, sempre podemos efetuar a operação inversa $U^\dagger |0\rangle = |x\rangle$, que é a preparação de estados. De fato, a transformação proposta é composta apenas por rotações uniformemente controladas, então sua inversa é simples e podemos construí-la diretamente, não sendo necessário efetuar a inversão. Portanto, a preparação de estados baseada em rotações pode ser definida como uma transformação linear consistindo de uma sequência de rotações uniformemente controladas que transforma o estado inicial $|0\rangle^{\otimes n}$ no vetor arbitrário $|x\rangle$.

Para explicar o procedimento, são introduzidas as expressões (2.12) e (2.13), definidas pelo vetor $\mathbf{x} = (|x_0|e^{iw_0}, \ldots, |x_{N-1}|e^{iw_{N-1}})$ (MOTTONEN et al., 2005)

$$\alpha_{j,k} = 2\text{asin}\left(\sqrt{\sum_{l=1}^{2^{k-1}} |x_{(2j-1)2^{k-1}+l}|^2} \Big/ \sqrt{\sum_{l=1}^{2^k} |x_{(j-1)2^k+l}|^2}\right) \tag{2.12}$$

$$\lambda_{j,k} = \sum_{l=1}^{2^{k-1}} (\omega_{(2j-1)2^{k-1}+l} - \omega_{(2j-2)2^{k-1}+l})/2^{k-1} \tag{2.13}$$

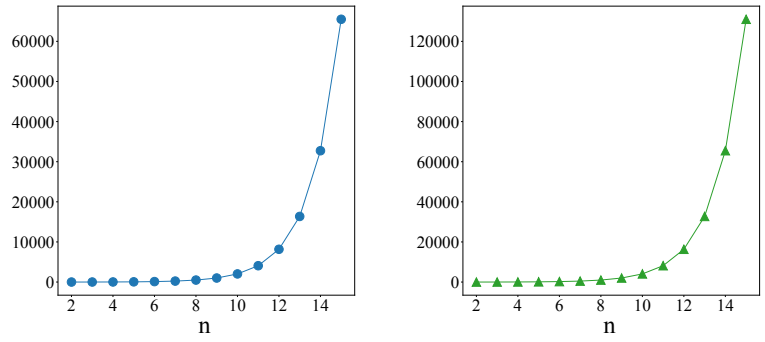
Figura 14 – Inicialização de um estado quântico de n qubits usando rotações para codificar um vetor de entrada complexo de dimensão $N = 2^n$ nas amplitudes do estado.



Fonte: Elaborada pelo autor (2022)

onde $j = 1, 2, \ldots, 2^{n-k}$, $k = 1, 2, \ldots, n$ e $n = \log_2(N)$. Essas expressões determinam ângulos de rotação para a codificação do vetor $\mathbf{x}$. Os argumentos das fases do vetor $|x\rangle$ são codificados através de rotações R_z de ângulos $\lambda_{j,k}$ e as magnitudes através de rotações R_y de ângulos $\alpha_{j,k}$ (Figura 14).

Figura 15 – Número de CNOTs e a profundidade de circuitos produzidos pela biblioteca qclib usando a preparação de estados baseada em rotações, em função do logaritmo do comprimento da entrada $n = \log_2(N)$.



(a) Número de CNOTs (b) Profundidade

Fonte: Elaborada pelo autor (2022)

O procedimento inicia preparando o seguinte estado para $k = n$ (estamos preparando a inversa da transformação, o que implica que a ordem de k é invertida), onde $\beta_{j,k} = \sin(\alpha_{j,k})/2$

$$|\psi_n\rangle = e^{-i\frac{\lambda_{1,n}}{2}}\sqrt{1-|\beta_{1,n}|^2}\,|0\rangle + e^{i\frac{\lambda_{1,n}}{2}}\beta_{1,n}\,|1\rangle\,. \tag{2.14}$$

Esse estado pode ser construído com apenas duas rotações no primeiro qubit (primeiro passo da Figura 14). Em seguida, o estado da expressão (2.14) é combinado com os valores do

segundo passo

$$|\psi_{n-1}\rangle = |0\rangle\langle 0|\psi_n\rangle \left(e^{-i\frac{\lambda_{1,n-1}}{2}}\sqrt{1-|\beta_{1,n-1}|^2}\,|0\rangle + e^{i\frac{\lambda_{1,n-1}}{2}}\beta_{1,n-1}\,|1\rangle\right) + |1\rangle\langle 1|\psi_n\rangle \left(e^{-i\frac{\lambda_{2,n-1}}{2}}\sqrt{1-|\beta_{2,n-1}|^2}\,|0\rangle + e^{i\frac{\lambda_{2,n-1}}{2}}\beta_{2,n-1}\,|1\rangle\right)$$

$$|\psi_{n-1}\rangle = e^{-i\frac{\lambda_{1,n}}{2}}\sqrt{1-|\beta_{1,n}|^2}\Big(e^{-i\frac{\lambda_{1,n-1}}{2}}\sqrt{1-|\beta_{1,n-1}|^2}\,|00\rangle + e^{i\frac{\lambda_{1,n-1}}{2}}\beta_{1,n-1}\,|01\rangle\Big) + e^{i\frac{\lambda_{1,n}}{2}}\beta_{1,n}\Big(e^{-i\frac{\lambda_{2,n-1}}{2}}\sqrt{1-|\beta_{2,n-1}|^2}\,|10\rangle + e^{i\frac{\lambda_{2,n-1}}{2}}\beta_{2,n-1}\,|11\rangle\Big)$$

$$|\psi_{n-1}\rangle = e^{-i\frac{\lambda_{1,n}}{2}}\sqrt{1-|\beta_{1,n}|^2}\Big(e^{-i\frac{\lambda_{1,n-1}}{2}}\sqrt{1-|\beta_{1,n-1}|^2}\,|0\rangle + e^{i\frac{\lambda_{1,n-1}}{2}}\beta_{1,n-1}\,|1\rangle\Big) + e^{i\frac{\lambda_{1,n}}{2}}\beta_{1,n}\Big(e^{-i\frac{\lambda_{2,n-1}}{2}}\sqrt{1-|\beta_{2,n-1}|^2}\,|2\rangle + e^{i\frac{\lambda_{2,n-1}}{2}}\beta_{2,n-1}\,|3\rangle\Big)$$

Então, os estados continuam a ser combinados com os valores dos passos seguintes

$$|\psi_k\rangle = \sum_{j=1}^{2^{n-k}} |j-1\rangle\langle j-1|\psi_{k+1}\rangle \left(e^{-i\frac{\lambda_{j,k}}{2}}\sqrt{1-|\beta_{j,k}|^2}\,|0\rangle + e^{i\frac{\lambda_{j,k}}{2}}\beta_{j,k}\,|1\rangle\right) \tag{2.15}$$

A atualização do estado $|\psi_k\rangle$ é repetida para $k = (n-1), \ldots, 1$, obtendo o estado desejado

$$|\psi_1\rangle = |x_0|e^{i\omega_0}\,|0\rangle + \ldots + |x_{N-1}|e^{i\omega_{N-1}}\,|N-1\rangle\,.$$

A combinação dos estados é realizada com rotações multicontroladas. A cada passo, é montada uma sequência de rotações sobre um qubit e controladas pelos qubits dos passos anteriores. Primeiro, rotações R_y são aplicadas para determinar as magnitudes das amplitudes, seguidas por rotações R_z para determinar as fases. Esses passos estão representados no Algoritmo 1 e na Figura 14.

Algoritmo 1: Preparação de estados baseada em rotações

input : Um vetor de estado com 2^n amplitudes.
output: O circuito para a inicialização do estado quântico.

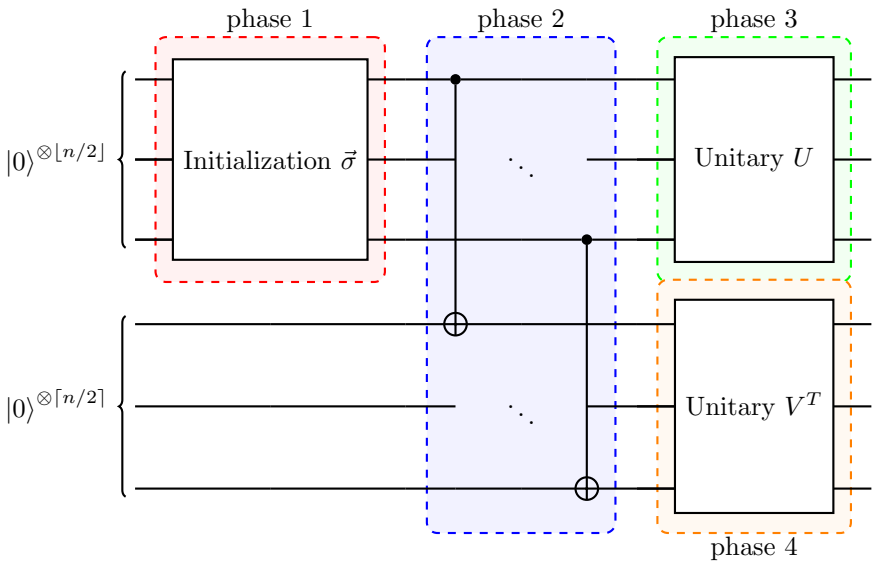
1 Cria um circuito quântico com n qubits (um qubit para cada passo)
2 Aplica uma rotação R_y e uma rotação R_z no primeiro qubit usando os ângulos $\alpha_{1,n}$ e $\lambda_{1,v}$ (Eq. 2.14)
3 Faz $k = n-1$ (segundo passo)
4 Realiza uma *Rotação Uniformemente Controlada* controlada pelos qubits $0, 1, \ldots, n-k-1$ (correspondendo aos passos anteriores) com o qubit atual $n-k$ como alvo, usando $\alpha_{j,k}$ e $\lambda_{j,k}$ $(1 \leq j \leq 2^{n-k})$ do passo corrente como ângulos das rotações (Eq. 2.15)
5 Se $k > 1$, faz $k = k-1$ e retorna para a linha 3, caso contrário retorna o circuito quântico para a inicialização do estado.

Fonte: Elaborado pelo autor (2022)

2.3.2 Baseada na decomposição de Schmidt

A preparação de estados proposta por Plesch e Brukner (2011) é baseada na decomposição de Schmidt. Dado um estado quântico $|\psi\rangle = \sum_j \lambda_j |j\rangle$, o primeiro passo do algoritmo é executar a decomposição de Schmidt $|\psi\rangle = \sum_i \sigma_i |\alpha_i\rangle \otimes |\beta_i\rangle \in \mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$ onde o estado é fatorado em dois sistemas quânticos $\mathcal{H}_A$ e $\mathcal{H}_B$, σ_i são os coeficientes de Schmidt, $\{|\alpha_i\rangle\} \in \mathcal{H}_A$ e $\{|\beta_i\rangle\} \in \mathcal{H}_B$ são bases ortonormais, e $1 \leq i \leq \min(\dim(\mathcal{H}_A), \dim(\mathcal{H}_B))$. O segundo passo inicializa o estado quântico $\sum_i \sigma_i |i\rangle |0\rangle$ no primeiro registrador quântico. O objetivo deste passo é inicializar um estado com os coeficientes de Schmidt codificados nas amplitudes. O terceiro passo aplica $\lfloor n/2 \rfloor$ CNOTs para criar o estado $\sum_i \sigma_i |i\rangle |i\rangle$. Sendo U e V unitários, onde $U|i\rangle = |\alpha_i\rangle$ e $V^T|i\rangle = |\beta_i\rangle$, o último passo do algoritmo aplica U no primeiro registrador e V^T no segundo. A Figura 16 e o Algoritmo 2 ilustram essa preparação de estados.

Figura 16 – Inicialização de um estado quântico de n qubits usando a decomposição de Schmidt.



Fonte: Elaborada pelo autor (2022)

Quando atuando em k qubits, uma preparação de estados requer $2^k - k - 1$ CNOTs (BERGHOLM et al., 2005) e um operador unitário $\frac{23}{48}2^{2k} - \frac{3}{2}2^k + \frac{4}{3}$ (SHENDE; BULLOCK; MARKOV, 2006). Sendo $\mathcal{H}_A$ um subsistema com n_A qubits ($n_A = n/2$ para n par e $n_A = (n-1)/2$ para n ímpar) e $n_B = n - n_A$ o número de qubits do subsistema $\mathcal{H}_B$ (n_B é maior ou igual a n_A) e considerando o circuito completo para a inicialização de estados, o número de CNOTs somando as contribuições de todas as fases (Figura 16) é representado por

n par:

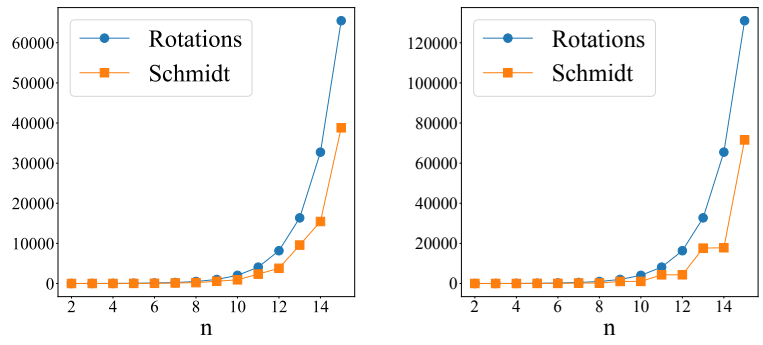
$$\underbrace{2^{n_A} - n_A - 1}_{\text{fase 1}} + \underbrace{n_A}_{\text{fase 2}} + \underbrace{2\left(\frac{23}{48}2^n - \frac{3}{2}2^{n_A} + \frac{4}{3}\right)}_{\text{fases 3 e 4 (unitários)}} \tag{2.16}$$

n ímpar:

$$\underbrace{2^{n_A} - n_A - 1}_{\text{fase 1}} + \underbrace{n_A}_{\text{fase 2}} + \underbrace{\frac{23}{48}2^{2n_A} - \frac{3}{2}2^{n_A} + \frac{4}{3}}_{\text{fase 3 (unitário)}} + \underbrace{\frac{23}{48}2^{2n_B} - \frac{3}{2}2^{n_B} + \frac{4}{3}}_{\text{fase 4 (unitário)}} \tag{2.17}$$

As chaves nas equações (2.16) e (2.17) indicam a contribuição de cada fase do procedimento de Plesch e Brukner (2011) para o número de CNOTs. A fase 1 é uma preparação de estado, a fase 2 uma sequência de portas CNOT e as fases 3 e 4 são decomposições de unitários. O número total de CNOTs é limitado por $\frac{23}{24}2^n$ quando n é par e por $\frac{115}{96}2^n$ quando n é ímpar. A profundidade do circuito é determinada pelo unitário da fase 4, pois $n_B = \lceil n/2 \rceil \geq n_A = \lfloor n/2 \rfloor$. A Figura 17 ilustra como a profundidade é afetada pela paridade do número de qubits $n = \log_2(N)$.

Figura 17 – Comparação entre o número de CNOTs e a profundidade de circuitos produzidos pela biblioteca qclib utilizando as preparações de estados baseadas em rotações e na decomposição de Schmidt, em função do logaritmo do comprimento da entrada $n = \log_2(N)$.



(a) Número de CNOTs (b) Profundidade

Fonte: Elaborada pelo autor (2022)

A decomposição de Schmidt reduz a profundidade do circuito quântico utilizado para a inicialização de estados (comparado com Mottonen et al. (2005)), mas há um aumento no custo computacional clássico – complexidade do algoritmo na etapa de preprocessamento clássico (Figura 2). Para calcular a decomposição de Schmidt é necessário realizar a decomposição SVD (Linha 1 do Algoritmo 2), que possui custo cúbico na raiz do comprimento da entrada $O(N^{3/2})$ (VASUDEVAN; RAMAKRISHNA, 2017).

Algoritmo 2: Preparação de estados baseada na decomposição de Schmidt

input : Um vetor de estado com 2^n amplitudes.
input : Uma partição com os primeiros $\lfloor n/2 \rfloor$ qubits do registrador.
output: O circuito para a inicialização do estado quântico.

1 Decompor o estado usando *SVD* e a partição (armazena os unitários u e vh, e o vetor s)
2 Cria um circuito quântico com n qubits ($n = \log_2(\text{len(vetor de estado)})$)
3 Codifica o vetor normalizado s nos qubits $q \in \{\text{partição}\}$ usando um algoritmo para a preparação de estados nas amplitudes. Este passo é chamado de *Phase 1*
4 Realiza $\lfloor n/2 \rfloor$ *CNOTs* entre os qubits de controle $q_c \in \{\text{partição}\}$ e os alvos $q_t \notin \{\text{partição}\}$. Este passo é chamado de *Phase 2*
5 Codifica o unitário u nos qubits $q \in \{\text{partição}\}$. Este passo é chamado de *Phase 3*
6 Codifica o unitário vh^T nos qubits $q \notin \{\text{partição}\}$. Este passo é chamado de *Phase 4*
7 Retorna o circuito quântico para a inicialização do estado.

Fonte: Elaborado pelo autor (2022)

2.4 DECOMPOSIÇÃO DE ISOMETRIAS

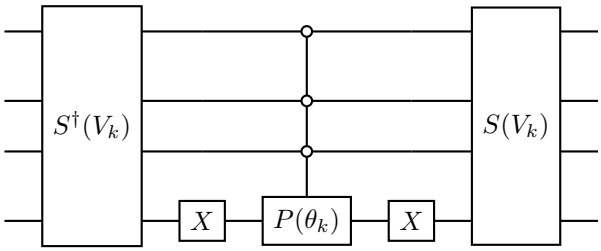
Isometrias são generalizações de operadores unitários e de preparações de estados. Uma isometria V é representada por uma matriz $2^n \times 2^m$ satisfazendo $V^\dagger V = \mathbb{I}$, onde $0 \leq m \leq n$. Os casos especiais em que $m = 0$ e $m = n$ são equivalentes à preparação de estados e à decomposição de unitários, respectivamente.

2.4.1 Decomposição de Knill

Essa decomposição foi idealizada por Knill (1995). Sendo V uma isometria de dimensão $2^n \times 2^m$, podemos estender V combinando-a com o complexo conjugado do seu espaço nulo $Z = \text{null}(V^T)^\dagger$ de dimensão $(2^n - 2^m) \times 2^n$ para produzir o unitário $U = \left[VZ^T\right]_{2^n \times 2^n}$, onde $\text{null}(V^T) = \{\vec{v} \in \mathbb{C}^{2^n} | V^T\vec{v} = \vec{0}\}$ e $V = U\mathbb{I}_{2^n \times 2^m}$. O transposto conjugado do espaço nulo é necessário para a condição de unitariedade $UU^\dagger = \mathbb{I}$. Esse procedimento estende V para uma base ortonormal em que as colunas de U são os estados da base e que possui no máximo 2^m autovalores com argumento complexo θ_k diferente de zero, minimizando o número de instâncias do circuito da Figura 18.

A Figura 18 mostra o esquema do circuito quântico para a decomposição de uma das 2^m colunas da isometria V. O circuito completo tem 2^m instâncias desse diagrama ($0 \leq k < 2^m$). O operador S é uma preparação de estado do tipo codificação nas amplitudes (Seção 2.2.3), e θ_k é o argumento do autovalor k do unitário U. Se o argumento for zero, a instância do

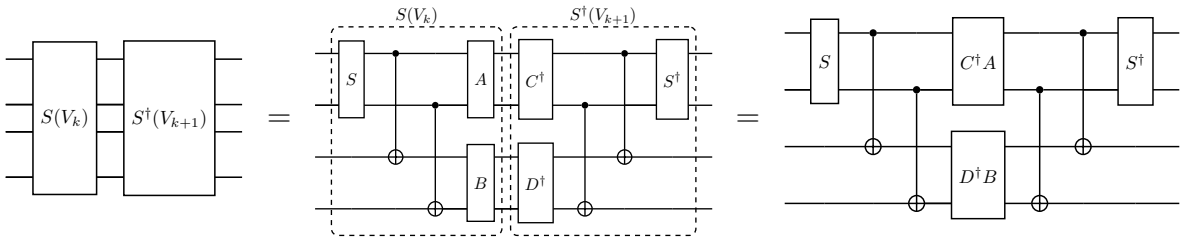
Figura 18 – Esquema de Knill para a decomposição de isometrias.



Fonte: Elaborada pelo autor (2022)

circuito pode ser ignorada.

Figura 19 – Otimização para a decomposição cosseno-seno (CSD) de isometrias.



Fonte: Elaborada pelo autor (2022)

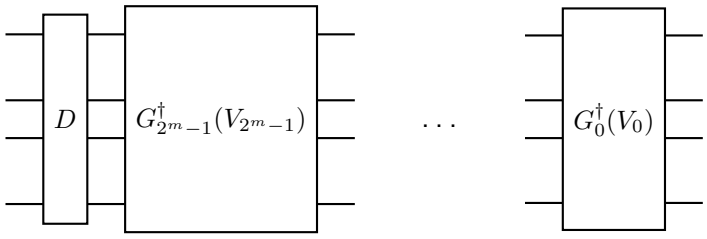
Iten et al. (2016) sugere utilizar a preparação de estados baseada na decomposição de Schmidt (Seção 2.3.2). Com essa inicialização, é possível otimizar a decomposição de isometrias baseadas no esquema de Knill. A Figura 19 mostra como combinar os operadores das fases 3 e 4 entre instâncias consecutivas do diagrama 18. Com essa otimização, a decomposição de Knill requer $\frac{23}{24}(2^{m+n} + 2^n) + O(n^2)2^m$ CNOTs para n par e $\frac{115}{96}(2^{m+n} + 2^n) + O(n^2)2^m$ CNOTs para n ímpar (ITEN et al., 2016).

2.4.2 Decomposição Coluna-por-Coluna

O método de decomposição coluna-por-coluna (*Column–by–Column Decomposition* ou CCD) para a decomposição de isometrias foi introduzido por Iten et al. (2016). Cada coluna k da isometria V é decomposta em um operador G_k (Figura 20 e Algoritmo 3). O operador G_k é uma variação da codificação nas amplitudes de Mottonen et al. (2005), sendo apenas $G_0^\dagger$ exatamente a preparação de estados baseada em rotações (Seção 2.3.1).

A variação (linhas 4–6 do Algoritmo 3) tem como objetivo garantir que a decomposição de cada nova coluna não afete as anteriores. Quando necessário, a variação remove alguns dos unitários U_l (Equação 2.20) dos operadores uniformemente controlados do algoritmo de Mottonen et al. (2005) e os substitui por um operador U multicontrolado (Linha 5 do Algoritmo 3).

Figura 20 – Esquema coluna-por-coluna para a decomposição de isometrias.



Fonte: Elaborada pelo autor (2022)

A Figura 20 e o Algoritmo 3 sumarizam o método. A Figura 20 apresenta o circuito quântico para a decomposição de 2^m colunas da isometria V. No Algoritmo 3, o unitário U e o conjunto de unitários U_l das linhas 5 e 7 são construídos conforme a Equação (2.19) e a Equação (2.20) Iten et al. (2016). A função a^k_{s+1} representa o inteiro composto pelos $n-(s+1)$ bits mais significativos de k. A função b^k_{s+1}, o inteiro composto pelos $s+1$ bits menos significativos de k (ver Apêndice A3 de (ITEN et al., 2016)). Na Equação (2.18), o estado $|\psi_l\rangle$ é composto pelos dois elementos de índices $2^s(2l)+b^k_{s+1}$ e $2^s(2l+1)+b^k_{s+1}$, normalizados, localizados na coluna k da isometria V, e $|\phi_l\rangle = -\langle\psi_l|1\rangle|0\rangle + \langle\psi_l|0\rangle|1\rangle$ (ver *Lemma 2* de (ITEN et al., 2016)). Deve ser percebido que as colunas são preparadas até um operador diagonal, o que significa que uma correção de fase é necessária. Isso é feito pelo operador D da Figura 20.

$$u_l = \begin{cases} |0\rangle\langle\psi_l| + |1\rangle\langle\phi_l| \text{ se } s=0 \\ |1\rangle\langle\psi_l| + |0\rangle\langle\phi_l| \text{ se } s\neq 0 \end{cases} \tag{2.18}$$

$$U = \begin{cases} u_l \quad \text{para } l = a^k_{s+1} \end{cases} \tag{2.19}$$

$$U_l = \begin{cases} \mathbb{I}_{2\times 2} \quad \text{para } l \in \{0,1,\ldots,a^k_{s+1}\} \text{ se } b^k_{s+1}\neq 0 \\ \mathbb{I}_{2\times 2} \quad \text{para } l \in \{0,1,\ldots,a^k_{s+1}-1\} \text{ se } b^k_{s+1}= 0 \\ u_l \quad \text{para } l \in \{a^k_{s+1}+1, a^k_{s+1}+2,\ldots,2^{n-1-s}-1\} \text{ se } b^k_{s+1}\neq 0 \\ u_l \quad \text{para } l \in \{a^k_{s+1}, a^k_{s+1}+1,\ldots,2^{n-1-s}-1\} \text{ se } b^k_{s+1}= 0 \end{cases} \tag{2.20}$$

Sabendo que até a conclusão do Algoritmo 3 a matriz quadrada $V_{2^m\times 2^m}$ (2^m é o número de colunas da isometria V) precisa tornar-se igual à identidade, qualquer fase presente na diagonal da matriz indica uma fase que deve ser corrigida usando o operador D. Portanto, ele é construído como um operador diagonal onde as fases são aquelas presentes em $V_{2^m\times 2^m}$.

Esse é o método mais eficiente para a decomposição de isometrias, com exceção do caso 2^{n-1}–para–2^n em que o método CSD é melhor (Seção 2.4.3). O número de CNOTs requerido

pelo CCD é dado por $2^{m+n} - \frac{1}{24}2^n + O(n^2)2^m$ (ITEN et al., 2016).

Algoritmo 3: Monta um operador G_k que realiza a decomposição da coluna k da isometria V.

input : Uma isometria V, número de qubits n, e o índice k da coluna da isometria.
output: Operador G_k.

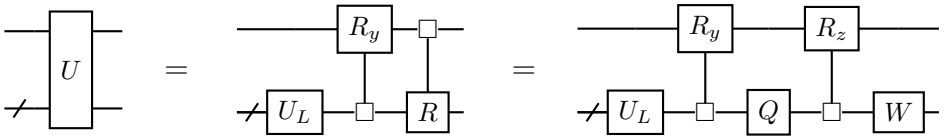
1 Cria um circuito quântico com n qubits para construir o operador G_k
2 Faz $s = 0$ e k_s como o bit de índice s da representação binária de k
3 Faz $target = n - s - 1$, $control = \{0, \ldots, target - 1\}$, e $ancilla = \{target + 1, \ldots, n - 1\}$
4 Se $k_s = 0$ e $b^k_{s+1} \neq 0$ continue, caso contrário vá para 7
5 Aplica uma porta multicontrolada MCG com unitário U, controlada por $control + ancilla$ e atuando em $target$
6 Atualiza a isometria $V = MCG \times V$
7 Aplica uma porta uniformemente controlada UCG com unitários U_l, controlada por $control$ e atuando em $target$
8 Atualiza a isometria $V = UCG \times V$
9 Se $s < n - 1$, faz $s = s + 1$ e retorna para 3, caso contrário retorna o operador G_k.

Fonte: Elaborado pelo autor (2022)

2.4.3 Decomposição Cosseno-Seno

Essa decomposição é uma adaptação da decomposição de unitários CSD (Seção 2.1.3) ou de sua otimização QSD (Seção 2.1.4) (ITEN et al., 2016). Assim como na decomposição de Knill (Seção 2.4.1), a isometria é estendida e representada em termos de um unitário $V = U\mathbb{I}_{2^n \times 2^m}$. Sendo $n > m$ e aplicando a primeira identidade da decomposição de unitários (Figura 9), o primeiro qubit do operador L da decomposição do unitário U não tem efeito sobre as colunas associadas à isometria (Figura 21), portanto não é necessário aplicar o controle sobre ele. Sendo assim, o operador L atua sobre $n-1$ qubits e é uma isometria 2^m–para–2^{n-1} estendida para unitário, identificada na Figura 21 como U_L. Em seguida, a identidade da decomposição de unitários QSD (Figura 11) é aplicada apenas ao operador uniformemente controlado R, produzindo os unitários Q e W separados pela rotação uniformemente controlada R_z (Figura 21).

Figura 21 – Esquema para a decomposição cosseno-seno (CSD) de isometrias.



Fonte: Elaborada pelo autor (2022)

O procedimento é repetido recursivamente para decompor U_L. Os unitários Q e W podem ser decompostos utilizando a identidade da Figura 11 até produzir operadores de dois qubits, como explicado na Seção 2.1.4.

Esse método produz circuitos com $\frac{23}{144}(4^m + 2 \times 4^n) + O(m)$ CNOTs, apropriado apenas quando $m = (n-1)$ (ITEN et al., 2016). Para $m < (n-1)$, o método CCD produz circuitos com número menor de CNOTs.

3 SOLUÇÕES EXISTENTES PARA A PREPARAÇÃO DE ESTADOS QUÂNTICOS

Várias soluções para o problema da preparação de estados quânticos foram propostas (VENTURA; MARTINEZ, 1999; GROVER, 2000; LONG; SUN, 2001; MOTTONEN et al., 2005; SHENDE; BULLOCK; MARKOV, 2006; PLESCH; BRUKNER, 2011; CORTESE; BRAJE, 2018), mas todas produzem circuitos com profundidade crescendo pelo menos exponencialmente no número de qubits. Por exemplo, o método proposto por Ventura e Martinez (1999) atinge a compressão exponencial $O(\log_2(N))$ na largura do circuito quântico enquanto exige profundidade linear $O(N)$ para dados N-dimensionais. Como há uma sobrecarga adicional de recursos em muitos algoritmos quânticos devido ao postulado da medição quântica (PARK; PETRUCCIONE; RHEE, 2019; PARK et al., 2019), tal custo linear pode impor restrições a possíveis vantagens, dominando o custo computacional da aplicação quântica pretendida.

As tentativas de preparar estados quânticos de forma mais eficiente incluem abordagens probabilísticas (ZHANG; YUNG; YUAN, 2021; PARK; PETRUCCIONE; RHEE, 2019) e estratégias para inicializar estados quânticos aproximados (ZOUFAL; LUCCHI; WOERNER, 2019; NAKAJI et al., 2021; MARIN-SANCHEZ; GONZALEZ-CONDE; SANZ, 2021). Mais recentemente, existem trabalhos que se concentram em classes específicas de estados quânticos. Por exemplo, como preparar estados uniformes (MOZAFARI et al., 2021), esparsos (MALVETTI; ITEN; COLBECK, 2021) ou de distribuição de probabilidade (ZOUFAL; LUCCHI; WOERNER, 2019). Entretanto, não há um entendimento claro de quais classes de estados quânticos podem ser criadas de forma eficiente.

O emaranhamento é um dos recursos quânticos que também permite o desenvolvimento de algoritmos e protocolos mais eficientes (ver Apêndice E). Sua relação com a complexidade algorítmica de um estado quântico foi demonstrada em trabalhos anteriores (MORA; BRIEGEL, 2006; MORA; BRIEGEL, 2005) e desempenha um papel importante em muitas aplicações quânticas, tais como comunicação quântica, correção de erros quânticos e criptografia quântica (RAUSSENDORF; BRIEGEL, 2001; BARREIRO et al., 2010; CALDERBANK; SHOR, 1996; STEANE, 1996; CLEVE; GOTTESMAN; LO, 1999; DüR; CIRAC, 2000).

Embora esteja claro que o emaranhamento é um recurso importante na computação quântica (JOZSA; LINDEN, 2003), ser capaz de codificar estados de baixo emaranhamento, tais como funções de densidade de probabilidade, ainda é útil, pois os algoritmos podem usar blocos de construção de baixo emaranhamento para alcançar estruturas de alto emaranhamento. Um exemplo é a simulação de processos estocásticos (BLANK; PARK; PETRUCCIONE, 2021;

Tabela 1 – Artigos sobre preparação de estados.

ano	artigo	flip-flop	caixa-preta	variacional	probabilístico	árvore	decomposição
1995	(KNILL, 1995)(†)(∗)						
1999	(VENTURA; MARTINEZ, 1999)(†)(∗)					x	
2000	(GROVER, 2000)		x				
2001	(LONG; SUN, 2001)					x	
2005	(BERGHOLM et al., 2005)					x	
	(MOTTONEN et al., 2005)(†)(∗)					x	
2006	(SHENDE; BULLOCK; MARKOV, 2006)(†)(∗)					x	
2008	(GIOVANNETTI; LLOYD; MACCONE, 2008b)	x					
	(GIOVANNETTI; LLOYD; MACCONE, 2008a)	x					
2011	(PLESCH; BRUKNER, 2011)(†)(∗)						x
2015	(ARUNACHALAM et al., 2015)						
2016	(ITEN et al., 2016)(†)(∗)					x	
2018	(CORTESE; BRAJE, 2018)					x	
	(LOW; KLIUCHNIKOV; SCHAEFFER, 2018)					x	
2019	(SANDERS et al., 2019)		x				
	(ZOUFAL; LUCCHI; WOERNER, 2019)(†)			x			
	(PARK; PETRUCCIONE; RHEE, 2019)				x		
2020	(KUZMIN; SILVI, 2020)			x			
	(MOZAFARI et al., 2020)				x		
	(CASARES, 2020)	x					
	(MATTEO; GHEORGHIU; MOSCA, 2020)	x					
	(PALER; OUMAROU; BASMADJIAN, 2020)	x					
2021	(VERAS et al., 2021)(†)	x					
	(VERAS; SILVA; SILVA, 2022)(†)	x					
	(ARAUJO et al., 2021b)(†)(∗)					x	
	(ARAUJO et al., 2021a)(†)(∗)					x	
	(WANG et al., 2021)		x				
	(GUO et al., 2021)			x			
	(ZHANG; YUNG; YUAN, 2021)					x	
	(ZHAO et al., 2021)			x			
	(GLEINIG; HOEFLER, 2021)(†)					x	
	(NAKAJI et al., 2021)			x			
	(MOZAFARI et al., 2021)					x	
	(MARIN-SANCHEZ; GONZALEZ-CONDE; SANZ, 2021)					x	
	(MALVETTI; ITEN; COLBECK, 2021)					x	

Fonte: Elaborada pelo autor (2022)

STAMATOPOULOS et al., 2020; WOERNER; EGGER, 2019), usando a estimativa de amplitude.

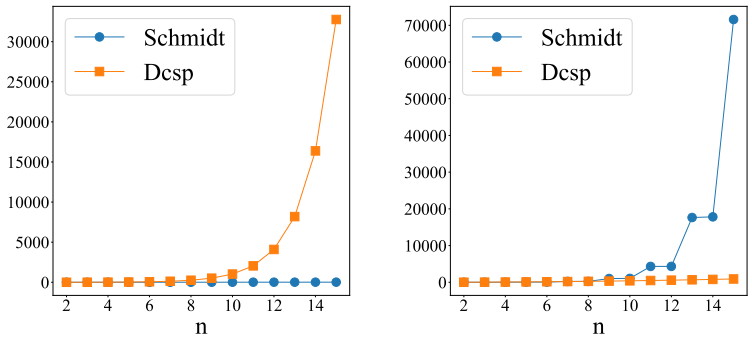
A Tabela 1 mostra uma lista do levantamento bibliográfico feito até o momento, categorizando o método proposto em cada artigo para a preparação de estados. Há 12 métodos implementados no repositório qclib, indicados por uma marca $(\dagger)$. Os oito itens em que trabalhei diretamente, estão indicados por $(*)$.

4 PRINCIPAIS CONTRIBUIÇÕES

4.1 A DIVIDE-AND-CONQUER ALGORITHM FOR QUANTUM STATE PREPARATION

O trabalho Araujo et al. (2021b) (Apêndice A) produziu um algoritmo capaz de carregar um vetor N-dimensional com vantagem de tempo exponencial usando um circuito quântico com profundidade polilogarítmica e informação emaranhada com qubits auxiliares (*Divide-and-Conquer quantum State Preparation* ou DCSP). Os resultados mostram que podemos carregar dados com eficiência em dispositivos quânticos usando uma estratégia de divisão e conquista para trocar tempo computacional por espaço.

Figura 22 – Comparação entre o número de qubits e a profundidade de circuitos produzidos pela biblioteca qclib utilizando a preparação de estados baseada na decomposição de Schmidt e o algoritmo DCSP, em função do logaritmo do comprimento da entrada $n = \log_2(N)$.



(a) Número de qubits (b) Profundidade

Fonte: Elaborada pelo autor (2022)

O método proposto estabelece uma abordagem para a preparação do estado quântico que pode complementar ou melhorar os métodos conhecidos, tais como a codificação nos ângulos (Seção 2.2.2) e a codificação nas amplitudes (Seção 2.2.3). A estratégia foi baseada no algoritmo de Mottonen et al. (2005) e em uma abordagem de divisão e conquista usando portas swap controladas e qubits auxiliares. Com esta modificação, obtemos uma redução exponencial no tempo para carregar um vetor N-dimensional nas amplitudes de um estado quântico com um circuito de profundidade $O(\log_2^2(N))$ e espaço $O(N)$ (Figura 4.1), onde $N = 2^n$ e n é o número de qubits do registrador de dados. Os qubits de dados estão associados aos nós da vista esquerda da árvore de estado (ver Apêndice A e Apêndice B). O registrador auxiliar possui $N - n - 1$ qubits e está associado ao restante dos nós da árvore de estado.

A redução na profundidade é atingida ao custo do uso de qubits auxiliares emaranhados com os qubits do registrador de dados. A informação emaranhada significa que, embora os

dados sejam codificados nas amplitudes de uma superposição de qubits, os resultados após a observação dos qubits auxiliares deixará o registrador de dados em um estado misto, o que pode ser uma desvantagem no caso de algoritmos para a solução de sistemas lineares (HARROW; HASSIDIM; LLOYD, 2009) ou equações diferenciais (XIN et al., 2020). Entretanto, mostramos que alguns problemas como a aprendizagem de máquina quântica supervisionada e a análise estatística podem ser realizados com o estado quântico de entrada dado pelo método.

Foi demonstrada uma prova de conceito utilizando um dispositivo quântico real e apresentadas duas aplicações para a aprendizagem de máquina quântica (ver Apêndice A). Espera-se que esta nova estratégia de carregamento permita a vantagem quântica para tarefas que requerem carregar um volume significativo de informações para os dispositivos quânticos. A troca entre as complexidades de tempo e espaço é favorável quando aumentar a largura do circuito é mais barato do que aumentar a profundidade, o que é um cenário provável de ocorrer durante o desenvolvimento dos dispositivos NISQ (IBM, 2020).

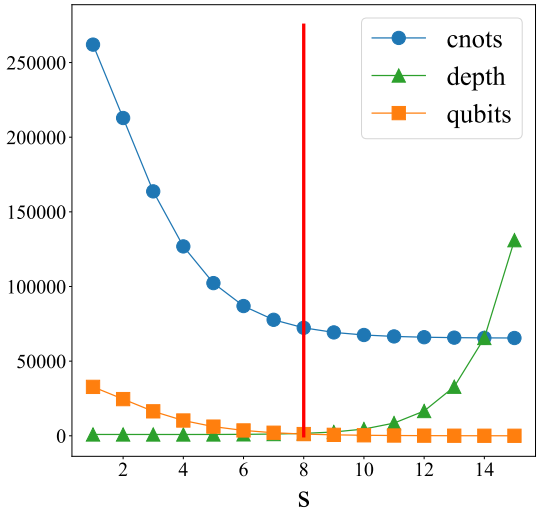
4.2 CONFIGURABLE SUBLINEAR CIRCUITS FOR QUANTUM STATE PREPARATION

A contribuição do trabalho Araujo et al. (2021a) (Apêndice B) é um procedimento bidirecional configurável que aborda o problema do custo de carregamento de dados clássicos em um estado quântico (*Bidirectional quantum State Preparation* ou BDSP), regulando a troca inicialmente apresentada no DCSP entre a largura e a profundidade do circuito quântico. Em particular, é apresentada uma configuração que codifica um estado N-dimensional usando um circuito quântico com largura e profundidade sublineares $O(\sqrt{N})$ e informação emaranhada com qubits auxiliares (Figura 23).

Os métodos existentes para a preparação de estados requerem pelo menos um recurso do circuito quântico – profundidade ou largura – crescendo linearmente com o tamanho do problema. O algoritmo BDSP fornece uma estrutura geral para configurar a troca entre estes recursos que pode ser útil para gerenciá-los em dispositivos NISQ. Olhando para os algoritmos de preparação de estados como uma caminhada na árvore de estado, o BDSP constitui uma forma sistemática de caminhar em duas direções opostas. Os métodos anteriores são baseados em caminhadas em apenas uma direção.

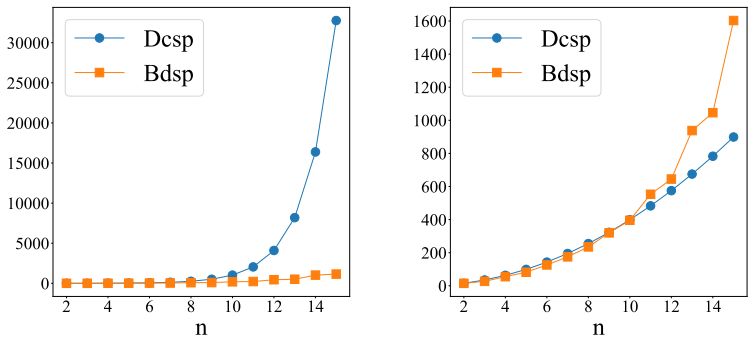
Tanto a complexidade temporal quanto a espacial dependem do número de qubits do registrador de dados n e também do parâmetro $s \in [1..n]$ que ajusta a troca entre tempo e espaço computacional. Dado um vetor de entrada N-dimensional ($N = 2^n$), a complexidade

Figura 23 – Ponto de sublinearidade do algoritmo BDSP para $\log_2(N) = 15$, indicado pela linha vermelha em $s = 8$.



Fonte: Elaborada pelo autor (2022)

Figura 24 – Comparação entre o número de qubits e a profundidade de circuitos produzidos pela biblioteca qclib utilizando o algoritmo DCSP e o BDSP com configuração sublinear $O(\sqrt{N})$, em função do logaritmo do comprimento da entrada $n = \log_2(N)$.



(a) Número de qubits (b) Profundidade

Fonte: Elaborada pelo autor (2022)

total do tempo do algoritmo bidirecional é $O_c(N) + O_d(2^s + \log_2^2(N) - s^2)$, onde $O_c(N)$ é o tempo de pré-processamento clássico para criar o circuito quântico e $O_d(2^s + \log_2^2(N) - s^2)$ é a profundidade do circuito. Tipicamente o mesmo vetor de entrada é carregado $l \gg N$ vezes, portanto o tempo computacional amortizado é $O_d(2^s + \log_2^2(N) - s^2)$. Note que o pré-processamento clássico é comum na computação clássica e também é necessário em outros métodos para a preparação de estados quânticos. A complexidade espacial (ou seja, a largura) do circuito é $O_w((s+1)^{N/2^s})$ (Figura 24).

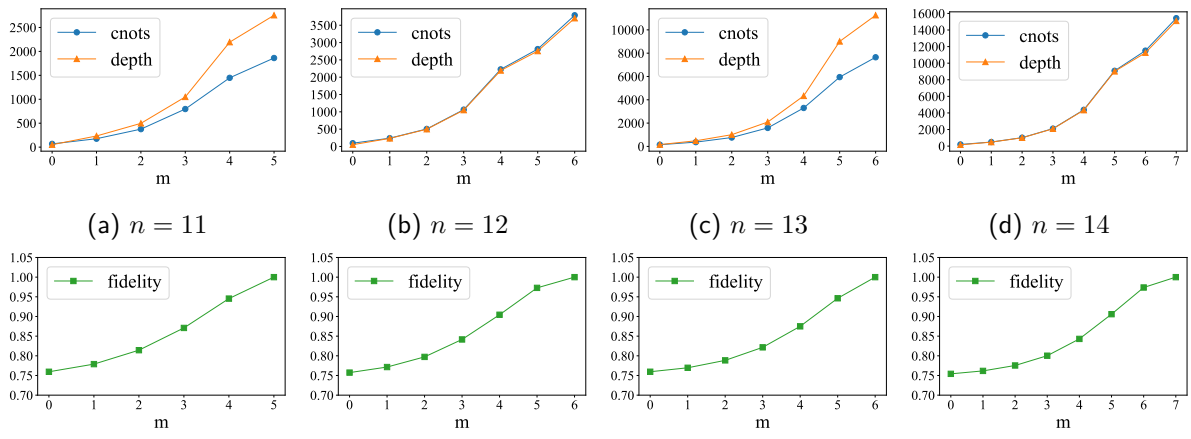
Além do custo do circuito sublinear, a capacidade de configurar a troca entre os recursos

quânticos é vantajosa quando especificações realistas de hardware quântico são consideradas, pois um recurso pode ser mais escalável do que o outro. Por exemplo, é útil para os futuros dispositivos NISQ, com a promessa de dispositivos com um grande número de qubits físicos (IBM, 2020), embora o ruído ainda limite a profundidade dos circuitos (PRESKILL, 2018). Como prova de conceito, o algoritmo foi aplicado em cinco computadores quânticos acessados através dos serviços de nuvem quântica da IBM e da IonQ.

4.3 APPROXIMATED QUANTUM-STATE PREPARATION WITH ENTANGLEMENT DEPENDENT COMPLEXITY

O artigo Araujo, Blank e Silva (2022) (apêndices C e D) resultou em um conjunto de algoritmos capaz de reduzir a profundidade do circuito de preparação de estados ao transferir parte da complexidade computacional para um computador clássico, mantendo a compressão exponencial na largura do circuito $O(\log_2(N))$. O estado quântico inicializado pode ser exato ou aproximado. Foi demonstrado que a aproximação é melhor nos processadores quânticos atuais do que a inicialização do estado original. Avaliou-se através de experimentos que o método proposto permite a inicialização de distribuições de probabilidade em estados quânticos e o carregamento aproximado de imagens para tarefas de aprendizagem de máquinas quânticas. De fato, os experimentos com dispositivos IBMQ (IBM, 2021b) mostram que o algoritmo provavelmente auxiliará os profissionais da computação quântica durante toda a era NISQ.

Figura 25 – Profundidade e número de CNOTs do circuito produzido pelo LRSP para carregar um vetor complexo de dimensão 2^n em um estado quântico aproximado com fidelidade ajustada pelo parâmetro m.



(a) $n = 11$ (b) $n = 12$ (c) $n = 13$ (d) $n = 14$

(e) $n = 11$ (f) $n = 12$ (g) $n = 13$ (h) $n = 14$

Fonte: Elaborada pelo autor (2022)

Algoritmos anteriores para a preparação de estados requerem $O(2^n)$ CNOTs para inicializar um estado quântico arbitrário exato. Entretanto, esses algoritmos não consideram a quantidade de emaranhamento. Este trabalho mostra que a complexidade do circuito para a preparação do estado está relacionada ao emaranhamento através de medidas numéricas de emaranhamento de estados puros baseadas no quadrado dos coeficientes de Schmidt. Essas medidas estão diretamente relacionadas ao número de CNOTs necessários para a inicialização exata do estado (Apêndice C). A redução da complexidade no regime quântico não é gratuita, pois a análise do emaranhamento na etapa de preprocessamento clássico, que permite que tal redução seja alcançada, tem um custo cúbico na raiz do comprimento da entrada, que não pode ser desprezado. Portanto, há uma troca entre as complexidades das etapas nos regimes clássico e quântico.

Tabela 2 – Erro Absoluto Médio entre probabilidades de medições de estados aproximados ($0 \leq m \leq 2$) de 7 qubits gerados pelo LRSP e as probabilidades esperadas para o estado original exato ($m = 3$).

m	CNOTs	profund.	fidelidade	ibmq_qasm_simulator	ibmq_casablanca	ibmq_jakarta	ibm_perth
0	13	11	0.78969	0.003730(0.000080)	**0.004384**(0.000280)	**0.004394**(0.000186)	0.004894(0.000467)
1	36	51	0.85460	0.003226(0.000067)	0.004557(0.000160)	0.004848(0.000286)	0.004666(0.000170)
2	84	115	0.95016	0.002068(0.000070)	0.004564(0.000173)	0.004523(0.000192)	**0.004581**(0.000274)
3	150	243	1.00000	**0.000696**(0.000049)	0.006293(0.001588)	0.005272(0.001090)	0.005353(0.000780)

Fonte: Elaborada pelo autor (2022)

Tabela 3 – Erro Absoluto Médio entre probabilidades de medições de estados aproximados ($l > 0.00$) de 7 qubits gerados pelo BAA e as probabilidades esperadas para o estado original exato ($l = 0.00$).

l	CNOTs	profund.	fidelidade	ibmq_qasm_simulator	ibmq_casablanca	ibmq_jakarta	ibm_perth
0.00	150	243	1.00000	**0.000706**(0.000064)	0.005557(0.001640)	0.005175(0.000841)	0.0054878(0.000848)
0.12	54	57	0.88742	0.003046(0.000073)	0.005015(0.000309)	0.004834(0.000212)	0.0052397(0.000347)
0.18	31	53	0.82141	0.003527(0.000101)	0.004653(0.000166)	0.004541(0.000159)	0.0051652(0.000507)
0.19	30	53	0.81630	0.003639(0.000045)	0.004630(0.000132)	0.004801(0.000208)	0.0051723(0.000338)
0.22	10	11	0.78536	0.003748(0.000069)	0.004332(0.000174)	0.004016(0.000219)	0.0048093(0.000333)
0.23	5	9	0.77064	0.003822(0.000065)	**0.003976**(0.000084)	**0.003939**(0.000070)	0.0042927(0.000290)
0.24	4	9	0.76403	0.004060(0.000074)	0.004054(0.000106)	0.004126(0.000081)	0.0043133(0.000174)
0.25	1	3	0.75097	0.004043(0.000090)	0.004214(0.000105)	0.004056(0.000079)	**0.0040059**(0.000068)
0.26	0	1	0.74502	0.004168(0.000059)	0.004369(0.000086)	0.004223(0.000055)	0.0042804(0.000085)

Fonte: Elaborada pelo autor (2022)

A complexidade dos circuitos produzidos pelo Low-Rank quantum State Preparation (LRSP) pode ser determinada de maneira semelhante à feita para a preparação de estados baseada na decomposição de Schmidt (Seção 2.3.2). Quando atuando em k qubits, uma preparação de estados nas amplitudes requer $2^k - k - 1$ CNOTs (BERGHOLM et al., 2005), um operador unitário $\frac{23}{48}2^{2k} - \frac{3}{2}2^k + \frac{4}{3}$ (SHENDE; BULLOCK; MARKOV, 2006) e uma isometria $2^{m+k} - \frac{1}{24}2^k + O(k^2)2^m$ (ITEN et al., 2016). Sendo $|\psi\rangle$ um estado com decomposição de Schmidt $|\psi\rangle = \sum_{i=1}^{k} \sigma_i |i_A\rangle |i_B\rangle$, onde o subsistema $\mathcal{H}_A$ tem n_A qubits ($1 \leq n_A \leq \lfloor n/2 \rfloor$), o subsistema

$\mathcal{H}_B$ tem $n_B = n - n_A$ qubits e k é o rank de Schmidt. Considerando $m = \lceil \log_2(k) \rceil < n_A$ a medida de Schmidt, o número de CNOTs (Figura 25 e Tabela 2) é dado por

$\underline{m < n_A}$:

$$\underbrace{2^m - m - 1}_{\text{fase 1}} + \underbrace{m}_{\text{fase 2}} + \underbrace{2^{m+n_A} - \frac{1}{24}2^{n_A}}_{\text{fase 3 (isometria)}} + \underbrace{2^{m+n_B} - \frac{1}{24}2^{n_B}}_{\text{fase 4 (isometria)}} \tag{4.1}$$

$\underline{m = n_A \text{ e } n_A < n_B}$:

$$\underbrace{2^{n_A} - n_A - 1}_{\text{fase 1}} + \underbrace{n_A}_{\text{fase 2}} + \underbrace{\frac{23}{48}2^{2n_A} - \frac{3}{2}2^{n_A} + \frac{4}{3}}_{\text{fase 3 (unitário)}} + \underbrace{2^n - \frac{1}{24}2^{n_B}}_{\text{fase 4 (isometria)}} \tag{4.2}$$

$\underline{m = n_A \text{ e } n_A = n_B}$:

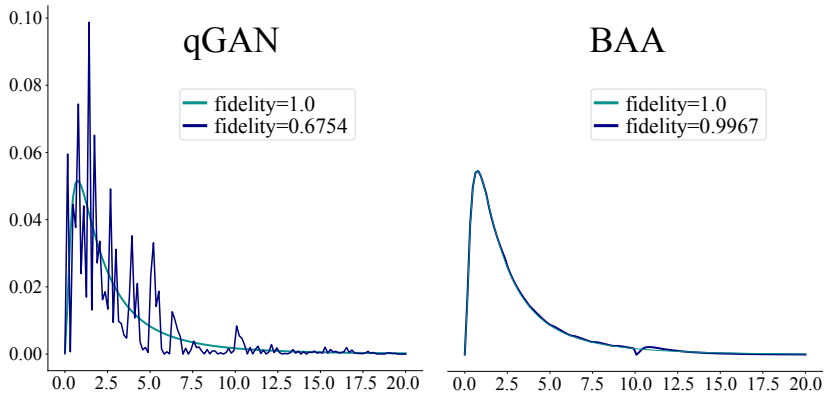
$$\underbrace{2^{n_A} - n_A - 1}_{\text{fase 1}} + \underbrace{n_A}_{\text{fase 2}} + \underbrace{2\left(\frac{23}{48}2^n - \frac{3}{2}2^{n_A} + \frac{4}{3}\right)}_{\text{fases 3 e 4 (unitários)}} \tag{4.3}$$

As contagens acima dependem da escolha das decomposições para as isometrias e os unitários. Os métodos escolhidos são o QSD (Seção 2.1.4) para os unitários (SHENDE; BULLOCK; MARKOV, 2006) e o CCD (Seção 2.4.2) para as isometrias (ITEN et al., 2016). É possível reduzir o número de CNOTs das isometrias para $\frac{23}{64}2^{2k}$ quando $m = k - 1$ usando o CSD (Seção 2.4.3) (ITEN et al., 2016).

Quando a bipartição é completamente separável ($m = 0$), as fases 1 e 2 não possuem operações e as fases 3 e 4 são isometrias de uma coluna. Sabendo que tais isometrias são equivalentes a uma preparação de estados (Seção 2.4), elas podem ser substituídas pelo próprio LRSP. Essa operação é executada recursivamente enquanto as bipartições forem separáveis. Assim, o custo assintótico do circuito $O(2^{m+n_e})$ será determinado pela maior partição não separável na recursão com n_e qubits.

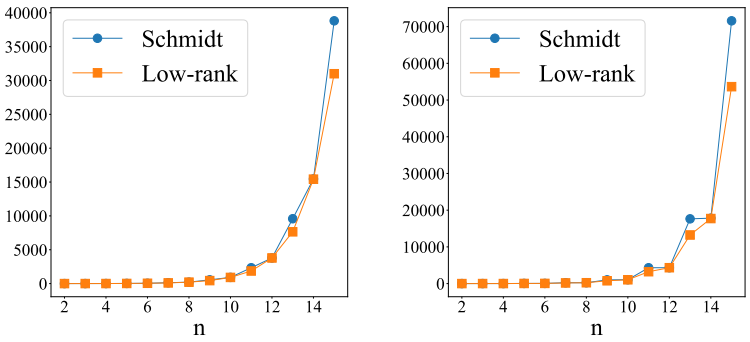
Outro resultado deste trabalho é a aproximação low-rank (ECKART; YOUNG, 1936) para a inicialização de estados quânticos usando o LRSP. Fixando a medida de Schmidt m e descartando os valores singulares menos significativos, o algoritmo encontra um estado com emaranhamento reduzido mais próximo possível do original. Isto leva a uma redução na complexidade dos circuitos, o que diminui o número de operações ruidosas. Mas quão complexo o estado aproximado deve ser para alcançar o melhor resultado depende das características do próprio dispositivo quântico. A partir desta perspectiva, a abordagem tem o potencial de caracterizar dispositivos quânticos quanto à sua capacidade de lidar com a complexidade da preparação de estados.

Figura 26 – Comparativo entre estados de 7 qubits inicializados por circuitos com 21 CNOTs construídos pelos algoritmos qGAN e BAA.



Fonte: Elaborada pelo autor (2022)

Figura 27 – Comparação entre o número de CNOTs e a profundidade de circuitos produzidos pela biblioteca qclib utilizando o algoritmo de Schmidt e o LRSP para inicializar estados emaranhados, em função do logaritmo do comprimento da entrada $n = \log_2(N)$.



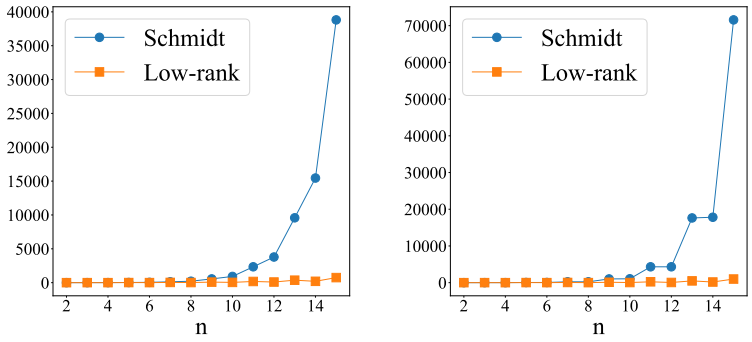
(a) Número de CNOTs (b) Profundidade

Fonte: Elaborada pelo autor (2022)

Uma variante da abordagem LRSP permite um algoritmo de preparação de estados com erro de aproximação limitada (*Bounded Approximation error state preparation Algorithm* ou BAA) sendo útil se um certo erro de aproximação l tiver que ser observado. Os estados de baixo emaranhamento podem ser geralmente criados como estados separáveis com um erro de aproximação baixo e custo constante, enquanto os estados de alto emaranhamento continuam sendo difíceis de criar. Mas mesmo para emaranhamento médio há melhorias consideráveis em portas CNOT salvas, ao custo de um aumento exponencial na complexidade computacional da etapa de preprocessamento clássico. A Figura 26 mostra um comparativo entre o BAA e o *Quantum Generative Adversarial Networks* (qGAN), onde ambos os métodos codificam uma distribuição log-normal como um estado de sete qubits usando apenas 21 CNOTs em um simulador ideal.

As tabelas 2 e 3 mostram a avaliação do Erro Absoluto Médio (*Mean Absolute Error* ou

Figura 28 – Comparação entre o número de CNOTs e a profundidade de circuitos produzidos pela biblioteca qclib utilizando o algoritmo de Schmidt e o LRSP para inicializar estados parcialmente emaranhados, em função do logaritmo do comprimento da entrada $n = \log_2(N)$.



(a) Número de CNOTs (b) Profundidade

Fonte: Elaborada pelo autor (2022)

MAE) entre probabilidades ideais da medição de estados aleatórios e as probabilidades da medição dos mesmo estados codificados em três dispositivos quânticos ruidosos. Para cada configuração, um estado aleatório de 7 qubits foi inicializado dez vezes. Para dispositivos ruidosos, o menor MAE ocorre quando o emaranhamento é menor que o do estado original. Isso indica que, em dispositivos ruidosos, é vantajoso usar estados aproximados com emaranhamento reduzido porque a perda de fidelidade introduzida pela aproximação é menor que a perda causada pelo ruído de circuitos mais complexos.

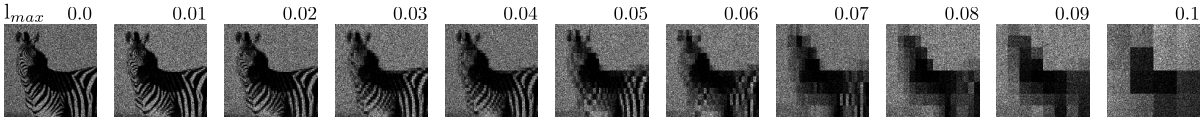
4.4 QCLIB: A QUANTUM COMPUTING LIBRARY

Todos os métodos listados acima estão implementados em um repositório de código aberto disponível publicamente no github.com. A biblioteca é chamada qclib (ARAUJO et al., 2022) e abrange todos os códigos usados nesta pesquisa, mas também possui outras rotinas de preparação de estados, unitários e isometrias que estão disponíveis de forma independente. É baseada no Qiskit (ALEKSANDROWICZ; AL., 2021) portanto, amplamente aplicável a muitos simuladores e dispositivos quânticos.

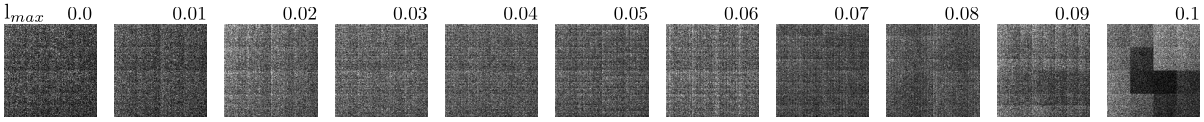
Um dos experimentos feitos para avaliar os limites da biblioteca, e que não está presente nos artigos, é o chamado *Experimento Zebra*. Testamos a viabilidade de codificar a imagem de uma zebra em dispositivos quânticos atuais. O experimento usou uma imagem preto e branco 8ppp com resolução de 128x128 píxeis (imagem original de Brand (2019)). Tal imagem possui 16384 píxeis que podem ser codificados nas amplitudes de um estado quântico de 14 qubits. A Figura 29a mostra os vetores de entrada aproximados pelo BAA. Os números acima de

Figura 29 – Experimento para avaliar os limites das ferramentas atuais da biblioteca qclib.

(a) Aproximações da imagem original produzidas pelo algoritmo BAA.



(b) Reconstrução das imagens feitas a partir de medições usando um simulador quântico ideal.



(c) Reconstrução das imagens a partir de medições usando um simulador quântico ruidoso.

Fonte: Elaborada pelo autor (2022)

cada imagem (as quais representam os vetores) é a perda de fidelidade máxima l_{max} usada nas aproximações. As imagens na Figura 29b foram construídas a partir de medições usando o simulador ideal ibmq_qasm_simulator. Um total de 20 execuções com 8192 *shots* foram aplicadas para cada vetor de entrada para alcançar esse resultado.

Tabela 4 – Número de CNOTs para inicializar um estado quântico representando diferentes aproximações da imagem usando o BAA.

l_{max}	0.0	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1
CNOTs	18175	11292	4473	2767	2767	1087	658	251	148	54	8

Fonte: Elaborada pelo autor (2022)

Finalmente, na Figura 29c, temos um experimento similar ao da figura anterior, mas usando os modelos de ruído e conectividade do dispositivo ibmq_mumbai. Pode ser visto que o ruído eliminou a habilidade de recuperar as imagens através de medições. A exceção é a imagem com a aproximação $l_{\text{max}} = 0.1$. Dada essa configuração, o circuito para a inicialização do estado possui apenas 8 CNOTs (Tabela 4), mas tal redução faz a imagem reconhecível apenas por suas características mais significativas. Em algumas aplicações, como aprendizagem de máquina, isso pode de fato ser suficiente, mas é evidente que o baixo desempenho exibido aqui limitará a aplicação.

O experimento mostra que os dispositivos atuais conseguem lidar apenas com imagens simples. E, principalmente, também mostra que o BAA consegue ajustar as imagens conforme a capacidade do dispositivo utilizado. Nossa expectativa é que os futuros dispositivos NISQ logo conseguirão lidar com aplicações de imagens do mundo real.

5 CONCLUSÃO

Embora os modelos de preparação de estados quânticos baseados em oráculos quânticos (GROVER, 2000; SANDERS et al., 2019; WANG et al., 2021) ou memória quântica de acesso aleatório (TRUGENBERGER, 2001; GIOVANNETTI; LLOYD; MACCONE, 2008b; GIOVANNETTI; LLOYD; MACCONE, 2008a; VERAS et al., 2021; CASARES, 2020; MATTEO; GHEORGHIU; MOSCA, 2020) sejam úteis para avaliar os limites inferiores do custo computacional e identificar a classe de complexidade, as implementações dos mesmos devem ser consideradas na prática. Em particular, a vantagem quântica pode desaparecer sem uma implementação eficiente da preparação de estados quando os algoritmos quânticos dependem do carregamento de dados clássicos em uma superposição quântica não uniforme (AARONSON, 2015; BIAMONTE et al., 2017; HARROW; HASSIDIM; LLOYD, 2009; TANG, 2021). Exemplos de tais casos incluem Aprendizagem de Máquina Quântica (QML) (LLOYD; MOHSENI; REBENTROST, 2013; BIAMONTE et al., 2017; STOUDENMIRE; SCHWAB, 2016; SCHULD; FINGERHUTH; PETRUCCIONE, 2017; SCHULD; PETRUCCIONE, 2018; BENEDETTI et al., 2019; LEVINE et al., 2019; AARONSON, 2015; BLANK et al., 2020), Memórias Quânticas (QMem) (TRUGENBERGER, 2001; VENTURA; MARTINEZ, 2000; TRUGENBERGER, 2002; GIOVANNETTI; LLOYD; MACCONE, 2008b; SILVA; OLIVEIRA; LUDERMIR, 2010; de Paula Neto et al., 2019; PARK; PETRUCCIONE; RHEE, 2019) e Álgebra Linear Quântica (QLA) (HARROW; HASSIDIM; LLOYD, 2009; LLOYD; MOHSENI; REBENTROST, 2014; CHILDS; KOTHARI; SOMMA, 2017; BIAMONTE et al., 2017; WOSSNIG; ZHAO; PRAKASH, 2018; REBENTROST et al., 2018). Para que essas aplicações quânticas tornem-se realidade, é imperativo alcançar uma inicialização de estados quânticos de baixa complexidade.

5.1 PRINCIPAIS RESULTADOS

As principais contribuições desta tese resultaram em três artigos e colaboraram para a criação de uma biblioteca de computação quântica de código aberto (*open-source*). O primeiro artigo está publicado (ARAUJO et al., 2021b) e os dois restantes (ARAUJO et al., 2021a; ARAUJO; BLANK; SILVA, 2022) foram submetidos a periódicos internacionais. Cada contribuição busca aperfeiçoar resultados anteriores. A Tabela 5 resume os resultados das contribuições, listando as complexidades temporal e espacial dos circuitos produzidos pelos algoritmos apresentados nesta tese para a inicialização de estados quânticos.

Tabela 5 – Sumário das contribuições com as complexidades temporal e espacial dos circuitos produzidos pelos algoritmos.

algoritmo	DCSP	BDSP	LRSP+BAA
temporal	$O(n^2)$	$O(2^s + n^2 - s^2)$	$O(2^{m+n_e})$
espacial	$O(2^n)$	$O((s+1)2^{n-s})$	$O(n)$
comentário	Troca entre complexidade temporal e espacial. Informação emaranhada com qubits auxiliares.	Troca configurável entre complexidade temporal e espacial. Informação emaranhada com qubits auxiliares.	Complexidade depende de propriedades do estado. Pode ser reduzida arbitrariamente em troca da perda de fidelidade. Informação não emaranhada com qubits auxiliares.

Fonte: Elaborada pelo autor (2022)

5.1.1 Algoritmo do tipo divisão e conquista para a preparação de estados quânticos

A primeira contribuição, um algoritmo do tipo divisão e conquista para a preparação de estados (*Divide-and-Conquer quantum State Preparation* ou DCSP), produz um circuito quântico com profundidade polilogarítmica $O(\log_2^2(N))$, uma redução exponencial em relação a resultados anteriores (Seção 4.1 e Apêndice A). Em troca, a largura do circuito é exponencialmente maior, $O(N)$ contra $O(\log_2(N))$, e a informação é emaranhada com qubits auxiliares.

5.1.2 Algoritmo bidirecional para a preparação de estados quânticos

O aumento exponencial na largura do circuito causado pelo DCSP é reduzido pela preparação bidirecional de estados quânticos (*Bidirectional quantum State Preparation* ou BDSP), segunda contribuição da nossa pesquisa (Seção 4.2 e Apêndice B). Essa estratégia permite regular a troca entre tempo $O(2^s + n^2 - s^2)$ e espaço $O((s+1)2^{n-s})$ através do parâmetro s, alcançando complexidade sublinear $O(\sqrt{N})$ simultânea em ambas as dimensões quando $s = n/2$ e $N \gg 1$. Embora reduzido, ainda há emaranhamento entre a informação e qubits auxiliares.

5.1.3 Algoritmo baseado na decomposição de Schmidt e na aproximação de baixo posto para a preparação de estados quânticos

Para solucionar o aumento na complexidade espacial do circuito e o emaranhamento com qubits auxiliares, a terceira contribuição usa o emaranhamento como um recurso computacional

(Seção 4.3 e apêndices C e D). Ao associar o emaranhamento à complexidade da preparação de estados quânticos, há maior controle sobre o número de CNOTs necessários para a inicialização de estados. Tal controle permite reduzir a complexidade temporal $O(2^{m+n_e})$ mantendo a largura do circuito $O(n)$, sem a necessidade de qubits auxiliares. Por exemplo, um estado completamente separável (sem emaranhamento, $m = 0$ e $n_e = 1$) é codificado usando apenas duas camadas de rotações e nenhum CNOT. Ainda é possível impor uma redução de emaranhamento para obter um estado aproximado em troca de um erro (*Low-Rank quantum State Preparation* ou LRSP).

5.1.4 Algoritmo de aproximação com erro limitado para a preparação de estados quânticos

A abordagem do algoritmo LRSP, em particular sua aproximação, pode ser generalizada por um algoritmo de busca recursivo que fornece uma aproximação até um determinado erro (*Bounded Approximation error state preparation Algorithm* ou BAA) e assim economiza o uso dos operadores CNOT, ao mesmo tempo em que fornece uma garantia de proximidade com o estado original.

O LRSP é baseado na decomposição de Schmidt e seu desempenho pode ser aprimorado através de uma escolha cuidadosa das bipartições. A função do BAA é encontrar a sequência de bipartições que permite ao LRSP produzir o circuito para a inicialização de estados com o menor número de portas CNOT.

5.1.5 Biblioteca de código aberto para computação quântica

Por fim, a pesquisa desta tese contribuiu com uma biblioteca de código aberto (ARAUJO et al., 2022) que implementa todos os códigos usados nesta pesquisa, mas não limitada a eles (Seção 4.4).

A Tabela 6 compara numericamente os circuitos produzidos por diferentes métodos para codificar estados exatos de 14 e 15 qubits. Os métodos LRSP, BDSP e DCSP são contribuições desta pesquisa e estão implementadas na biblioteca qclib (ARAUJO et al., 2022). O método baseado na decomposição de Schmidt (Seção 2.3.2) (PLESCH; BRUKNER, 2011) também está implementado na biblioteca qclib. Embora o qclib implemente as preparações de estados baseadas na isometria CCD (Seção 2.4.2) (ITEN et al., 2016) e em multiplexadores (SHENDE;

Tabela 6 – Comparativo numérico entre número de qubits, CNOTs e profundidade de circuitos construídos pelas bibliotecas qclib e IBM Qiskit usando diferentes algoritmos para a codificação de vetores complexos com dimensão 2^n.

		$n = 14$			$n = 15$		
método	lib	qubits	cnots	profund.	qubits	cnots	profund.
LRSP	qclib	14	15425	17719	15	30999	53643
Schmidt	qclib	14	15454	17821	15	38814	71581
Isometria	Qiskit	14	16369	65479	15	32752	131011
Multiplexador	Qiskit	14	32738	65492	15	65504	131025
BDSP	qclib	1023	38792	1046	1151	72320	1603
DCSP	qclib	16383	130952	783	32767	262016	899

Fonte: Elaborada pelo autor (2022)

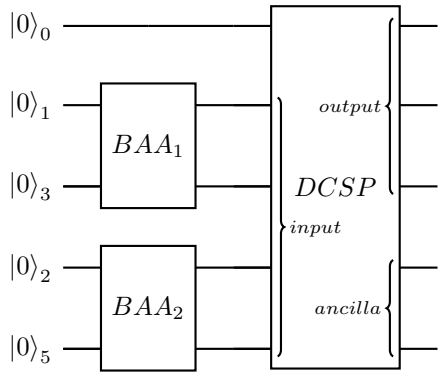
BULLOCK; MARKOV, 2006), para esse comparativo decidimos usar a implementação da biblioteca Qiskit da IBM (ALEKSANDROWICZ; AL., 2021). O método Multiplexador implementa o algoritmo de Mottonen et al. (2005) (Seção 2.3.1) otimizado com o uso dos multiplexadores de Shende, Bullock e Markov (2006). Esse último é a preparação de estados padrão das bibliotecas Qiskit (ALEKSANDROWICZ; AL., 2021) e PennyLane (BERGHOLM et al., 2020). A isometria (ITEN et al., 2016) e o método baseado na decomposição de Schmidt (PLESCH; BRUKNER, 2011), ambos com as otimizações sugeridas em (SHENDE; BULLOCK; MARKOV, 2006) (Seção 2.1.4), eram os métodos mais eficientes para a codificação exata de estados quânticos arbitrários (PLESCH; BRUKNER, 2011; ITEN et al., 2016; MALVETTI; ITEN; COLBECK, 2021).

5.2 TRABALHOS FUTUROS

5.2.1 Big-data state preparation

Todos os métodos para a preparação de estados quânticos apresentados até o momento possuem custo de pré-processamento clássico exponencial, limitando seu uso a estados com poucos qubits. n qubits implica armazenar 2^n amplitudes. Usando uma estimativa simplista - 16 bytes por amplitude complexa - são necessários 2^{n+4} bytes para armazenar um estado quântico de n qubits. Portanto, para 50 qubits são necessários $2^{54} \approx 18 \times 10^{15}$ bytes (18 petabytes), o que é inviável mesmo para os supercomputadores atuais. A normalização de tal estado é um desafio ainda maior. Mas é justamente nesta escala que os algoritmos quânticos conseguirão demonstrar a vantagem quântica (PRESKILL, 2018), quando executarão soluções impraticáveis para os computadores clássicos.

Figura 30 – Método bidirecional usando BAA.



Fonte: Elaborada pelo autor (2022)

Para contornar esse problema podemos aplicar o BDSP (Seção 4.2), combinando o BAA (Seção 4.3) com o DCSP (Seção 4.1). Dessa maneira, estados muito largos podem ser divididos em estados menores e combinados no próprio computador quântico (Figura 30), reduzindo exponencialmente o teto da memória clássica requerida para preparar o estado – 2^{n-s} subestados de 2^s amplitudes, com $1 \leq s \leq n$ (Apêndice B). Adicionalmente, a complexidade do circuito é dominada pela fase do DCSP, que possui custo polilogarítmico, pois a complexidade do BAA pode ser reduzida arbitrariamente em troca de uma perda de fidelidade.

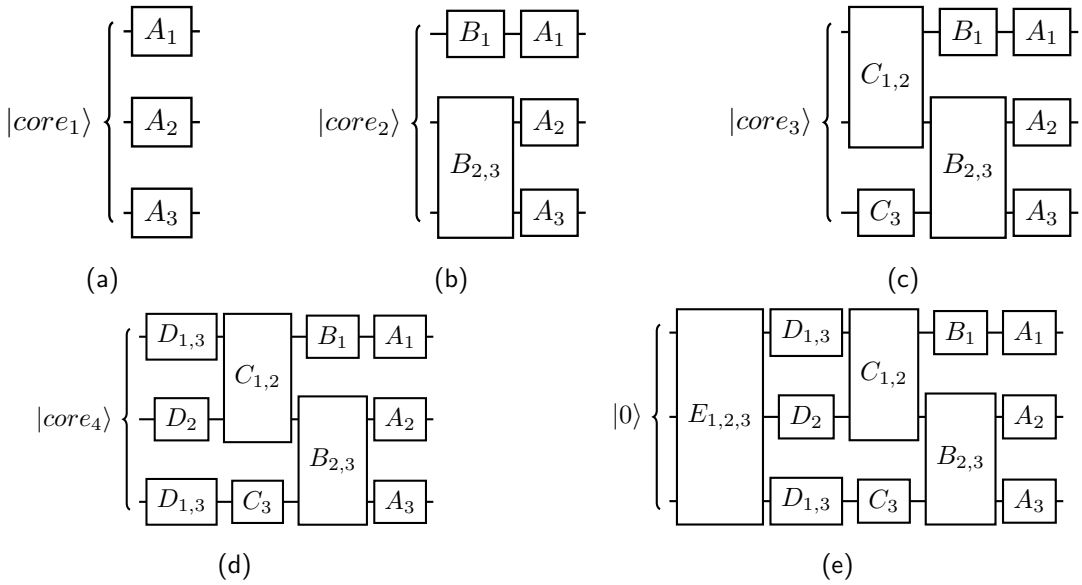
Por exemplo, sendo $n = 50$ e $s = 40$, o BDSP gera 2^{10} subestados com 2^{40} amplitudes (uma redução de 10 qubits em relação ao estado original). Cada subestado ocupa $2^{44} \approx 16 \times 10^{12}$ bytes (16 terabytes) de armazenamento (uma redução de aproximadamente 1000 vezes em relação ao teto do armazenamento original). Essa configuração produz um circuito quântico com $(s+1)2^{n-s} - 1 = 41 \times 2^{10} - 1 = 41983$ qubits. A fase DCSP, responsável por combinar os subestados, tem profundidade de $\sum_{i=s+1}^{n} i - 1 = \sum_{41}^{50} i - 1 = 445$ camadas de operadores swap controlados. Espera-se que nos próximos anos essa ordem no número de qubits seja alcançada (IBM, 2021a). A transferência de tais circuitos para o computador quântico ainda é um problema que precisa ser resolvido.

5.2.2 Tensor based quantum state preparation

O algoritmo de Plesch (PLESCH; BRUKNER, 2011) utiliza o SVD para executar uma decomposição de Schmidt, dividindo o estado em duas partições iguais e sequenciais (se a dimensão do estado for ímpar, uma das partes será maior, pois $\dim(\mathcal{H}_1) = \lfloor n/2 \rfloor$ e $\dim(\mathcal{H}_2) = \lceil n/2 \rceil$). O Low-Rank quantum State Preparation (LRSP) flexibiliza o algoritmo, permitindo que as

partições possuam qualquer tamanho (mínimo de 1 qubit e máximo de n-1 qubits) e sejam compostas de qubits não sequenciais. Mas ainda temos a limitação de apenas duas partições. O novo método baseado na decomposição tensorial generaliza o LRSP, tornando configurável o número de partições.

Figura 31 – Decomposição tensorial do estado em uma sequência de operadores atuando em um número crescente de qubits.



Fonte: Elaborada pelo autor (2022)

A principal consequência do método tensorial é que mais CNOTs poderão ser manipulados para gerenciar o emaranhamento do estado, abrindo novas possibilidades para algoritmos de busca como o BAA. A Figura 31 exibe exemplos dos circuitos produzidos pela nova abordagem. A decomposição total, calculada através da decomposição recursiva de vetores tipo *core*, produz um circuito com operadores atuando em todas as combinações de qubits (Figura 31e). Um vetor core é construído pelo redimensionamento de um tensor core. No primeiro passo do algoritmo (Figura 31a), o vetor $|core_1\rangle$ e os operadores A_j são calculados a partir da decomposição tensorial do vetor de entrada. No segundo passo (Figura-31b), o vetor $|core_2\rangle$ e os operadores B_j são calculados a partir da decomposição tensorial do $|core_1\rangle$. A recursão continua até o último passo (Figura 31e), onde o vetor core é igual ao primeiro vetor da base computacional $|0\rangle$, encerrando a recursão. Idealmente, os operadores maiores devem possuir correlação apenas entre os qubits mais distantes, pois as correlações entre qubits mais próximos são estabelecidas pelos operadores menores.

Um algoritmo de busca, como o BAA, pode alcançar uma aproximação limitada por um erro máximo removendo operadores sistematicamente. Outra alternativa é guiar a aproximação

segundo o mapa de acoplamento dos qubits do dispositivo alvo, mantendo apenas os operadores que atuam em qubits conectados diretamente no mapa ou minimizando as distâncias das conexões. As conexões podem até mesmo ser priorizadas de acordo com sua qualidade. Ainda é possível selecionar os operadores de acordo com o erro de CNOT entre os qubits, medida que incorpora a distância e a qualidade das conexões.

Assim como no LRSP, o método tensorial também permite aproximações tipo low-rank do estado original. Por exemplo, o *low multilinear rank* (VANNIEUWENHOVEN; VANDEBRIL; MEERBERGEN, 2012) modifica o método HOSVD aplicando uma série de aproximações low-rank (alterando a linha 3 do Algoritmo 4). Como resultado dessa aproximação, os fatores da decomposição são isometrias, reduzindo a complexidade dos circuitos produzidos.

De maneira geral, o rank de um tensor $\mathcal{X}$ de ordem N é definido como o número mínimo de tensores rank-1 necessários para produzir $\mathcal{X}$ como sua soma.

$$\begin{aligned}\mathcal{X} &= \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \otimes \mathbf{a}_r^{(2)} \otimes \cdots \otimes \mathbf{a}_r^{(N)} \\ &= \left[\boldsymbol{\lambda}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}\right]\end{aligned}$$

Consequentemente, o número de partições (portanto, de subespaços) é dado pelo modo (ou ordem) do tensor.

Para construirmos algoritmos precisamos de métodos para elaborar a decomposição tensorial de maneira eficiente. O método pretendido é o *HOSVD*, também conhecido como *Multilinear Singular Value Decomposition* (LATHAUWER; MOOR; VANDEWALLE, 2000). O HOSVD é uma decomposição de Tucker ortogonal. A ortogonalidade indica que os fatores $\mathbf{A}^{(n)}$ obtidos através desse método são matrizes unitárias. O Algoritmo 4 descreve seu funcionamento.

Algoritmo 4: Decomposição HOSVD de um tensor $\mathcal{X}$ em um tensor core $\mathcal{G}$ e seus fatores $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}$.

input : Um tensor $\mathcal{X}$ e uma lista de ranks R_i $(1 \leq i \leq N)$.
output: Tensor core $\mathcal{G}$ e os fatores $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}$.

1 Faz $n = 1$
2 Faz $\mathbf{X}_{(n)}$ igual ao unfold de ordem n do tensor $\mathcal{X}$
3 Aplica o SVD à matriz $\mathbf{X}_{(n)} = \mathbf{A}^{(n)}\mathbf{\Sigma}_n\mathbf{V}_n^T$ e armazena o fator $\mathbf{A}^{(n)}$
4 Se $n < N$, faz $n = n + 1$ e retorna para 2, caso contrário continua
5 Calcula o tensor core $\mathcal{G} = \mathcal{X} \times_1 \mathbf{A}^{(1)\dagger} \times_2 \cdots \times_N \mathbf{A}^{(N)\dagger}$
6 Retorna $\mathcal{G}, \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}$.

Fonte: Elaborado pelo autor (2022)

5.3 OBSERVAÇÕES FINAIS

A maioria dos algoritmos quânticos não apresenta vantagem no carregamento de dados clássicos em um estado quântico (BIAMONTE et al., 2017; AARONSON, 2015). Os métodos apresentados neste trabalho propõem novos paradigmas para a preparação de estados quânticos que podem complementar ou melhorar os métodos conhecidos, tais como a codificação nos ângulos, codificação na base, e a codificação nas amplitudes (seções 2.2.1, 2.2.3 e 2.2.2).

O primeiro conjunto de algoritmos (seções 4.1 e 4.2 e apêndices A e B), baseados em Mottonen et al. (2005) e na abordagem de divisão e conquista, alcança uma redução exponencial no tempo para carregar um vetor de dados clássicos em um estado quântico em comparação a soluções anteriores como as baseadas em rotações (Seção 2.3.1) e na decomposição de Schmidt (Seção 2.3.2). Tal ganho possui impacto potencial na solução de problemas de aprendizagem de máquinas quântica e outros algoritmos quânticos que precisam carregar dados de dispositivos clássicos.

Como prova de conceito, a validade e a viabilidade desses métodos foram verificadas utilizando a plataforma de nuvem quântica da IBM (IBM, 2021b). Além disso, os experimentos numéricos mostraram que os novos métodos de codificação oferecem vantagens, reduzindo a complexidade e os recursos computacionais quando aplicado em conjunto com os algoritmos existentes. Espera-se que estas vantagens se estendam a outros casos.

O segundo conjunto de algoritmos (Seção 4.3 e apêndices C e D) é fundamentado em Plesch e Brukner (2011) e utiliza a relação entre emaranhamento quântico e preparação de estados (MORA; BRIEGEL, 2005; MORA; BRIEGEL, 2006) para fornecer uma melhor compreensão da complexidade da inicialização de estados. Demonstrou-se que o grau de emaranhamento do estado está diretamente associado ao número de operadores CNOT necessários para a sua inicialização. Portanto, a manipulação do emaranhamento através da aproximação de low-rank tem impacto direto na complexidade do circuito.

Experimentos com sete dispositivos quânticos (IBM, 2021b) mostram que as medições do estado gerado pelos circuitos aproximados pelo low-rank são mais próximas dos resultados esperados do que as medições usando o circuito exato. Isso ocorre porque o algoritmo encontra um estado com emaranhamento reduzido o mais próximo possível do original. Isto leva a uma redução na complexidade dos circuitos, diminuindo o número de operações ruidosas. Uma aproximação de low-rank do estado pode viabilizar o circuito final em dispositivos ruidosos para aplicações que dependem do carregamento de dados em um estado quântico arbitrário.

Um exemplo de aplicação de aprendizagem de máquina quântica do tipo Variational Quantum Classifier (VQC) (HAVLíčEK et al., 2019; SCHULD et al., 2020) mostra que o ajuste dos dados de entrada conforme a capacidade do dispositivo em gerenciar o emaranhamento, melhora o desempenho do classificador (Apêndice C). Além disso, as aplicações VQC têm a característica de carregar o mesmo vetor de entrada muitas vezes, amortizando o tempo computacional do pré-processamento clássico.

REFERÊNCIAS

AARONSON, S. Read the fine print. *Nature Physics*, v. 11, n. 4, p. 291–293, 2015.

ALEKSANDROWICZ, G.; AL. et. *Qiskit: An Open-source Framework for Quantum Computing*. 2021.

AMIN, M. H.; ANDRIYASH, E.; ROLFE, J.; KULCHYTSKYY, B.; MELKO, R. Quantum boltzmann machine. *Phys. Rev. X*, American Physical Society, v. 8, p. 021050, May 2018.

ARAUJO, I. F.; BLANK, C.; SILVA, A. da; CESAR, I.; SILVA, L. *Quantum computing library (qclib)*. GitHub, 2022. Disponível em: <https://github.com/qclib/qclib>.

ARAUJO, I. F.; BLANK, C.; SILVA, A. J. da. Approximated quantum-state preparation with entanglement dependent complexity. *arXiv:2111.03132 [quant-ph]*, maio 2022. ArXiv: 2111.03132.

ARAUJO, I. F.; PARK, D. K.; LUDERMIR, T. B.; OLIVEIRA, W. R.; PETRUCCIONE, F.; SILVA, A. J. da. Configurable sublinear circuits for quantum state preparation. arXiv, 2021.

ARAUJO, I. F.; PARK, D. K.; PETRUCCIONE, F.; SILVA, A. J. da. A divide-and-conquer algorithm for quantum state preparation. *Scientific Reports*, v. 11, n. 1, p. 6329, 2021.

ARUNACHALAM, S.; GHEORGHIU, V.; JOCHYM-O'CONNOR, T.; MOSCA, M.; SRINIVASAN, P. V. On the robustness of bucket brigade quantum RAM. *New Journal of Physics*, IOP Publishing, v. 17, n. 12, p. 123010, 2015.

ARUTE, F.; ARYA, K.; BABBUSH, R.; BACON, D.; BARDIN, J. C.; BARENDS, R.; BISWAS, R.; BOIXO, S.; BRANDAO, F. G.; BUELL, D. A. et al. Quantum supremacy using a programmable superconducting processor. *Nature*, Nature Publishing Group, v. 574, n. 7779, p. 505–510, 2019.

BARENCO, A. A universal two-bit gate for quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, v. 449, n. 1937, p. 679–683, jun. 1995. ISSN 0962-8444, 2053-9177.

BARENCO, A.; BENNETT, C. H.; CLEVE, R.; DIVINCENZO, D. P.; MARGOLUS, N.; SHOR, P.; SLEATOR, T.; SMOLIN, J. A.; WEINFURTER, H. Elementary gates for quantum computation. *Phys. Rev. A*, American Physical Society, v. 52, p. 3457–3467, 1995.

BARNUM, H.; LINDEN, N. Monotones and invariants for multi-particle quantum states. *Journal of Physics A: Mathematical and General*, v. 34, n. 35, p. 6787, 2001. ISSN 0305-4470.

BARREIRO, J. T.; SCHINDLER, P.; GüHNE, O.; MONZ, T.; CHWALLA, M.; ROOS, C. F.; HENNRICH, M.; BLATT, R. Experimental multiparticle entanglement dynamics induced by decoherence. *Nature Physics*, v. 6, p. 943–946, 2010. ISSN 1745-2473.

BENEDETTI, M.; LLOYD, E.; SACK, S.; FIORENTINI, M. Parameterized quantum circuits as machine learning models. *Quantum Science and Technology*, 2019.

BENIOFF, P. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *Journal of Statistical Physics*, v. 22, n. 5, p. 563–591, maio 1980. ISSN 0022-4715, 1572-9613.

BERGHOLM, V.; IZAAC, J.; SCHULD, M.; GOGOLIN, C.; ALAM, M. S.; AHMED, S.; ARRAZOLA, J. M.; BLANK, C.; DELGADO, A.; JAHANGIRI, S.; MCKIERNAN, K.; MEYER, J. J.; NIU, Z.; SZáVA, A.; KILLORAN, N. *PennyLane: Automatic differentiation of hybrid quantum-classical computations*. 2020.

BERGHOLM, V.; VARTIAINEN, J. J.; MöTTöNEN, M.; SALOMAA, M. M. Quantum circuits with uniformly controlled one-qubit gates. *Physical Review A*, v. 71, n. 5, p. 052330, 2005.

BIAMONTE, J.; WITTEK, P.; PANCOTTI, N.; REBENTROST, P.; WIEBE, N.; LLOYD, S. Quantum machine learning. *Nature*, v. 549, n. 7671, p. 195–202, 2017.

BLANK, C.; PARK, D. K.; PETRUCCIONE, F. Quantum-enhanced analysis of discrete stochastic processes. *npj Quantum Information*, Nature Publishing Group, v. 7, n. 1, p. 1–9, 2021.

BLANK, C.; PARK, D. K.; RHEE, J.-K. K.; PETRUCCIONE, F. Quantum classifier with tailored quantum kernel. *npj Quantum Information*, v. 6, n. 1, p. 1–7, 2020.

BRAND, J. 2019. [Online; accessed March 3, 2022]. Disponível em: <https://unsplash.com/photos/dbz3gmdubqw>.

CALDERBANK, A. R.; SHOR, P. W. Good quantum error-correcting codes exist. *Phys. Rev. A*, American Physical Society, v. 54, p. 1098–1105, 1996.

CASARES, P. A. M. Circuit implementation of bucket brigade qram for quantum state preparation. arXiv, 2020.

CHILDS, A. M.; KOTHARI, R.; SOMMA, R. D. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM Journal on Computing*, Society for Industrial Applied Mathematics (SIAM), v. 46, n. 6, p. 1920–1950, 2017.

CLEVE, R.; GOTTESMAN, D.; LO, H.-K. How to share a quantum secret. *Phys. Rev. Lett.*, American Physical Society, v. 83, p. 648–651, 1999.

CORTESE, J. A.; BRAJE, T. M. *Loading Classical Data into a Quantum Computer*. 2018.

de Paula Neto, F. M.; da Silva, A. J.; de Oliveira, W. R.; LUDERMIR, T. B. Quantum probabilistic associative memory architecture. *Neurocomputing*, v. 351, p. 101–110, 2019.

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

DEUTSCH, D.; PENROSE, R. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, v. 400, n. 1818, p. 97–117, jul. 1985.

DEUTSCH, D. E.; BARENCO, A.; EKERT, A. Universality in quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, v. 449, n. 1937, p. 669–677, jun. 1995. ISSN 0962-8444, 2053-9177.

DIVINCENZO, D. P. Two-bit gates are universal for quantum computation. *Phys. Rev. A*, American Physical Society, v. 51, p. 1015–1022, Feb 1995.

DIVINCENZO, D. P. The physical implementation of quantum computation. *Fortschritte der Physik*, v. 48, n. 9-11, p. 771–783, 2000.

DONNE, J. *Devotions upon Emergent Occasions*. 2007. ed. London: Project Gutenberg, 1624.

DUNJKO, V.; TAYLOR, J. M.; BRIEGEL, H. J. Quantum-enhanced machine learning. *Phys. Rev. Lett.*, American Physical Society, v. 117, p. 130501, Sep 2016.

DüR, W.; CIRAC, J. I. Multiparty teleportation. *Journal of Modern Optics*, Taylor Francis, v. 47, n. 2-3, p. 247–255, 2000.

ECKART, C.; YOUNG, G. The approximation of one matrix by another of lower rank. *Psychometrika*, Springer, v. 1, n. 3, p. 211–218, 1936.

ENRíQUEZ, M.; PUCHAłA, Z.; ŻYCZKOWSKI, K. Minimal rényi–ingarden–urbanik entropy of multipartite quantum states. *Entropy*, v. 17, n. 7, p. 5063–5084, 2015. ISSN 1099-4300.

FEYNMAN, R. P. Simulating physics with computers. *International Journal of Theoretical Physics*, v. 21, n. 6-7, p. 467–488, jun. 1982. ISSN 0020-7748, 1572-9575.

GIDNEY, C.; EKERÅ, M. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum*, Verein zur Förderung des Open Access Publizierens in den Quantenwissenschaften, v. 5, p. 433, abr. 2021. ISSN 2521-327X.

GIOVANNETTI, V.; LLOYD, S.; MACCONE, L. Architectures for a quantum random access memory. *Phys. Rev. A*, American Physical Society, v. 78, p. 052310, 2008.

GIOVANNETTI, V.; LLOYD, S.; MACCONE, L. Quantum random access memory. *Phys. Rev. Lett.*, American Physical Society, v. 100, p. 160501, 2008.

GLEINIG, N.; HOEFLER, T. An Efficient Algorithm for Sparse Quantum State Preparation. In: *2021 58th ACM/IEEE Design Automation Conference (DAC)*. San Francisco, CA, USA: IEEE, 2021. p. 433–438. ISBN 9781665432740.

GRASEDYCK, L. Hierarchical singular value decomposition of tensors. *SIAM J. Matrix Anal. Appl.*, Society for Industrial and Applied Mathematics, USA, v. 31, n. 4, p. 2029–2054, 2010. ISSN 0895-4798.

GROVER, L. K. A fast quantum mechanical algorithm for database search. In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing*. Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996. (STOC '96), p. 212–219.

GROVER, L. K. Synthesis of Quantum Superpositions by Quantum Computation. *Physical Review Letters*, v. 85, n. 6, p. 1334–1337, 2000.

GUNDLAPALLI, P.; LEE, J. Deterministic, scalable, and entanglement efficient initialization of arbitrary quantum states. *arXiv*, 2021.

GUO, S.-F.; CHEN, F.; LIU, Q.; XUE, M.; CHEN, J.-J.; CAO, J.-H.; MAO, T.-W.; TEY, M. K.; YOU, L. Faster state preparation across quantum phase transition assisted by reinforcement learning. *Phys. Rev. Lett.*, American Physical Society, v. 126, p. 060401, 2021.

HARROW, A. W.; HASSIDIM, A.; LLOYD, S. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, American Physical Society, v. 103, p. 150502, 2009.

HAVLíčEK, V.; CóRCOLES, A. D.; TEMME, K.; HARROW, A. W.; KANDALA, A.; CHOW, J. M.; GAMBETTA, J. M. Supervised learning with quantum-enhanced feature spaces. *Nature*, v. 567, p. 209–212, 2019.

HILL, S. A.; WOOTTERS, W. K. Entanglement of a pair of quantum bits. *Phys. Rev. Lett.*, American Physical Society, v. 78, p. 5022–5025, Jun 1997.

HOGG, T.; HUBERMAN, B. A.; WILLIAMS, C. P. Phase transitions and the search problem. *Artificial Intelligence*, v. 81, n. 1, p. 1–15, 1996. Frontiers in Problem Solving: Phase Transitions and Complexity.

IBM. *IBM's Roadmap For Scaling Quantum Technology*. 2020. Disponível em: <https://research.ibm.com/blog/ibm-quantum-roadmap>.

IBM. *Expanding the IBM Quantum Roadmap to anticipate the future of quantum-centric supercomputing*. 2021. Disponível em: <https://research.ibm.com/blog/ibm-quantum-roadmap-2025>.

IBM. *IBM Quantum*. 2021. Disponível em: <https://quantum-computing.ibm.com/>.

ITEN, R.; COLBECK, R.; KUKULJAN, I.; HOME, J.; CHRISTANDL, M. Quantum circuits for isometries. *Physical Review A*, APS, v. 93, n. 3, p. 032318, 2016.

JOZSA, R.; LINDEN, N. On the role of entanglement in quantum-computational speed-up. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, The Royal Society, v. 459, n. 2036, p. 2011–2032, 2003.

KHANEJA, N.; BROCKETT, R.; GLASER, S. J. Time optimal control in spin systems. *Phys. Rev. A*, American Physical Society, v. 63, p. 032308, 2001.

KIEFEROVÁ, M.; WIEBE, N. Tomography and generative training with quantum boltzmann machines. *Phys. Rev. A*, American Physical Society, v. 96, p. 062327, Dec 2017.

KNILL, E. Approximation by quantum circuits. arXiv, 1995.

KOLDA, T. G.; BADER, B. W. Tensor Decompositions and Applications. *SIAM Review*, v. 51, n. 3, p. 455–500, 2009. ISSN 0036-1445.

KRAUS, B.; CIRAC, J. I. Optimal creation of entanglement using a two-qubit gate. *Phys. Rev. A*, American Physical Society, v. 63, p. 062309, May 2001.

KUZMIN, V. V.; SILVI, P. Variational quantum state preparation via quantum data buses. *Quantum*, Verein zur Förderung des Open Access Publizierens in den Quantenwissenschaften, v. 4, p. 290, 2020. ISSN 2521-327X.

LATHAUWER, L. D.; MOOR, B. D.; VANDEWALLE, J. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, v. 21, n. 4, p. 1253–1278, 2000.

LEVINE, Y.; SHARIR, O.; COHEN, N.; SHASHUA, A. Quantum Entanglement in Deep Learning Architectures. *Physical Review Letters*, v. 122, n. 6, p. 065301, 2019.

LEYMANN, F.; BARZEN, J. The bitter truth about gate-based quantum algorithms in the NISQ era. *Quantum Science and Technology*, IOP Publishing, v. 5, n. 4, p. 044007, 2020.

LLOYD, S. Almost any quantum logic gate is universal. *Phys. Rev. Lett.*, American Physical Society, v. 75, p. 346–349, Jul 1995.

LLOYD, S.; MOHSENI, M.; REBENTROST, P. Quantum algorithms for supervised and unsupervised machine learning. *arXiv:1307.0411 [quant-ph]*, 2013.

LLOYD, S.; MOHSENI, M.; REBENTROST, P. Quantum principal component analysis. *Nature Physics*, v. 10, n. 9, p. 631–633, set. 2014. ISSN 1745-2481.

LONG, G.-L.; SUN, Y. Efficient scheme for initializing a quantum register with an arbitrary superposed state. *Physical Review A*, v. 64, n. 1, p. 014303, 2001.

LOW, G. H.; KLIUCHNIKOV, V.; SCHAEFFER, L. *Trading T-gates for dirty qubits in state preparation and unitary synthesis*. 2018.

LOW, G. H.; YODER, T. J.; CHUANG, I. L. Quantum inference on bayesian networks. *Phys. Rev. A*, American Physical Society, v. 89, p. 062315, Jun 2014.

MALVETTI, E.; ITEN, R.; COLBECK, R. Quantum circuits for sparse isometries. *Quantum*, v. 5, p. 412, 2021.

MANIN, Y. I. *Computable and Non-Computable*. Sovetskoe Radio, Moscow: [s.n.], 1980. 128 p.

MARIN-SANCHEZ, G.; GONZALEZ-CONDE, J.; SANZ, M. Quantum algorithms for approximate function loading. arXiv, 2021.

MATTEO, O. D.; GHEORGHIU, V.; MOSCA, M. Fault-tolerant resource estimation of quantum random-access memories. *IEEE Transactions on Quantum Engineering*, v. 1, p. 1–13, 2020.

MOLL, N.; BARKOUTSOS, P.; BISHOP, L. S.; CHOW, J. M.; CROSS, A.; EGGER, D. J.; FILIPP, S.; FUHRER, A.; GAMBETTA, J. M.; GANZHORN, M.; KANDALA, A.; MEZZACAPO, A.; MüLLER, P.; RIESS, W.; SALIS, G.; SMOLIN, J.; TAVERNELLI, I.; TEMME, K. Quantum optimization using variational algorithms on near-term quantum devices. *Quantum Science and Technology*, IOP Publishing, v. 3, n. 3, p. 030503, 2018.

MORA, C. E.; BRIEGEL, H. J. Algorithmic complexity and entanglement of quantum states. *Phys. Rev. Lett.*, American Physical Society, v. 95, p. 200503, 2005.

MORA, C. E.; BRIEGEL, H. J. Algorithmic complexity of quantum states. *International Journal of Quantum Information*, v. 04, p. 715–737, 2006. ISSN 0219-7499.

MOTTONEN, M.; VARTIAINEN, J. J.; BERGHOLM, V.; SALOMAA, M. M. Transformation of Quantum States Using Uniformly Controlled Rotations. *Quantum Info. Comput.*, v. 5, n. 6, p. 467–473, 2005.

MOZAFARI, F.; RIENER, H.; SOEKEN, M.; MICHELI, G. D. Efficient Boolean Methods for Preparing Uniform Quantum States. *IEEE Transactions on Quantum Engineering*, v. 2, p. 1–12, 2021. ISSN 2689-1808.

MOZAFARI, F.; SOEKEN, M.; RIENER, H.; MICHELI, G. D. Automatic Uniform Quantum State Preparation Using Decision Diagrams. In: *2020 IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL)*. Miyazaki, Japan: IEEE, 2020. p. 170–175. ISBN 9781728154060.

NAKAJI, K.; UNO, S.; SUZUKI, Y.; RAYMOND, R.; ONODERA, T.; TANAKA, T.; TEZUKA, H.; MITSUDA, N.; YAMAMOTO, N. Approximate amplitude encoding in shallow parameterized quantum circuits and its application to financial market indicator. arXiv, 2021.

NIELSEN, M. A.; CHUANG, I. L. *Quantum computation and quantum information*. 10th anniversary ed. ed. Cambridge ; New York: Cambridge University Press, 2010. ISBN 9781107002173.

ORúS, R. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, v. 349, p. 117–158, 2014. ISSN 0003-4916.

PAIGE, C.; WEI, M. History and generality of the cs decomposition. *Linear Algebra and its Applications*, v. 208-209, p. 303–326, 1994. ISSN 0024-3795.

PALER, A.; OUMAROU, O.; BASMADJIAN, R. Parallelizing the queries in a bucket-brigade quantum random access memory. *Physical Review A*, v. 102, n. 3, p. 032608, 2020. ISSN 2469-9926, 2469-9934.

PARK, D. K.; PETRUCCIONE, F.; RHEE, J.-K. K. Circuit-Based Quantum Random Access Memory for Classical Data. *Scientific Reports*, v. 9, n. 1, p. 3949, 2019.

PARK, D. K.; SINAYSKIY, I.; FINGERHUTH, M.; PETRUCCIONE, F.; RHEE, J.-K. K. Parallel quantum trajectories via forking for sampling without redundancy. *New Journal of Physics*, IOP Publishing, v. 21, n. 8, p. 083024, 2019.

PEREZ-GARCIA, D.; VERSTRAETE, F.; WOLF, M. M.; CIRAC, J. I. Matrix Product State Representations. *arXiv*, 2006.

PLESCH, M.; BRUKNER, C. Quantum-state preparation with universal gate decompositions. *Physical Review A*, v. 83, n. 3, p. 032302, 2011.

PRESKILL, J. Quantum computing and the entanglement frontier. arXiv, 2012.

PRESKILL, J. Quantum Computing in the NISQ era and beyond. *Quantum*, v. 2, p. 79, 2018.

RAUSSENDORF, R.; BRIEGEL, H. J. A one-way quantum computer. *Phys. Rev. Lett.*, American Physical Society, v. 86, p. 5188–5191, 2001.

REBENTROST, P.; MOHSENI, M.; LLOYD, S. Quantum Support Vector Machine for Big Data Classification. *Physical Review Letters*, v. 113, n. 13, p. 130503, 2014.

REBENTROST, P.; STEFFENS, A.; MARVIAN, I.; LLOYD, S. Quantum singular-value decomposition of nonsparse low-rank matrices. *Physical Review A*, American Physical Society (APS), v. 97, n. 1, 2018.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, Association for Computing Machinery, New York, NY, USA, v. 21, n. 2, p. 120–126, feb 1978. ISSN 0001-0782.

RøNNOW, T. F.; WANG, Z.; JOB, J.; BOIXO, S.; ISAKOV, S. V.; WECKER, D.; MARTINIS, J. M.; LIDAR, D. A.; TROYER, M. Defining and detecting quantum speedup. *Science*, v. 345, n. 6195, p. 420–424, 2014.

SANDERS, Y. R.; LOW, G. H.; SCHERER, A.; BERRY, D. W. Black-box quantum state preparation without arithmetic. *Phys. Rev. Lett.*, American Physical Society, v. 122, p. 020502, 2019.

SCHULD, M.; BOCHAROV, A.; SVORE, K. M.; WIEBE, N. Circuit-centric quantum classifiers. *Phys. Rev. A*, American Physical Society, v. 101, p. 032308, 2020.

SCHULD, M.; FINGERHUTH, M.; PETRUCCIONE, F. Implementing a distance-based classifier with a quantum interference circuit. *EPL (Europhysics Letters)*, v. 119, n. 6, p. 60002, 2017.

SCHULD, M.; PETRUCCIONE, F. *Supervised Learning with Quantum Computers*. 1st ed. 2018. ed. Cham: Springer International Publishing : Imprint: Springer, 2018. (Quantum Science and Technology).

SHENDE, V.; BULLOCK, S.; MARKOV, I. Synthesis of quantum-logic circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, v. 25, n. 6, p. 1000–1010, 2006.

SHENDE, V.; MARKOV, I.; BULLOCK, S. Smaller two-qubit circuits for quantum communication and computation. In: *Proceedings Design, Automation and Test in Europe Conference and Exhibition*. IEEE Xplore: IEEE, 2004b. v. 2, p. 980–985.

SHENDE, V. V.; BULLOCK, S. S.; MARKOV, I. L. Recognizing small-circuit structure in two-qubit operators. *Phys. Rev. A*, American Physical Society, v. 70, p. 012310, 2004c.

SHENDE, V. V.; MARKOV, I. L.; BULLOCK, S. S. Minimal universal two-qubit controlled-NOT-based circuits. *Physical Review A*, American Physical Society (APS), v. 69, n. 6, 2004a.

SHIMONY, A. Degree of Entanglement. *Annals of the New York Academy of Sciences*, v. 755, n. 1, p. 675–679, 1995. ISSN 1749-6632.

SHOR, P. W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Review*, v. 41, n. 2, p. 303–332, 1999.

SILVA, A.; OLIVEIRA, W. d.; LUDERMIR, T. A Weightless Neural Node Based on a Probabilistic Quantum Memory. In: *2010 Eleventh Brazilian Symposium on Neural Networks*. Sao Paulo, Brazil: IEEE, 2010. p. 259–264. ISSN: 2375-0235.

SIMON, D. R. On the Power of Quantum Computation. *SIAM Journal on Computing*, v. 26, n. 5, p. 1474–1483, 1997.

STAMATOPOULOS, N.; EGGER, D. J.; SUN, Y.; ZOUFAL, C.; ITEN, R.; SHEN, N.; WOERNER, S. Option Pricing using Quantum Computers. *Quantum*, v. 4, p. 291, 2020.

STEANE, A. Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, v. 452, p. 2551–2577, 1996. ISSN 1364-5021.

STEWART, G. W.; SUN, J.-g. *Matrix perturbation theory*. Boston: Academic Press, 1990. (Computer science and scientific computing). ISBN 9780126702309.

STOUDENMIRE, E.; SCHWAB, D. J. Supervised learning with tensor networks. In: LEE, D.; SUGIYAMA, M.; LUXBURG, U.; GUYON, I.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems*. Centre Convencions Internacional Barcelona, Barcelona Spain: Curran Associates, Inc., 2016. v. 29, p. 9.

TANG, E. Quantum principal component analysis only achieves an exponential speedup because of its state preparation assumptions. *Phys. Rev. Lett.*, American Physical Society, v. 127, p. 060503, 2021.

TENG, P. Accurate calculation of the geometric measure of entanglement for multipartite quantum states. *Quantum Information Processing*, v. 16, n. 7, p. 181, 2017. ISSN 1570-0755.

TERHAL, B. M.; SMOLIN, J. A. Single quantum querying of a database. *Physical Review A*, v. 58, n. 3, p. 1822–1826, 1998.

TRUGENBERGER, C. A. Probabilistic quantum memories. *Physical Review Letters*, v. 87, n. 6, 2001.

TRUGENBERGER, C. A. Quantum Pattern Recognition. *Quantum Information Processing*, v. 1, n. 6, p. 471–493, 2002.

TUCCI, R. R. A rudimentary quantum compiler(2cnd ed.). arXiv, 1999.

VANNIEUWENHOVEN, N.; VANDEBRIL, R.; MEERBERGEN, K. A new truncation strategy for the higher-order singular value decomposition. *SIAM Journal on Scientific Computing*, v. 34, n. 2, p. A1027–A1052, 2012.

VASUDEVAN, V.; RAMAKRISHNA, M. A hierarchical singular value decomposition algorithm for low rank matrices. arXiv, 2017.

VENTURA, D.; MARTINEZ, T. Initializing the Amplitude Distribution of a Quantum State. *Foundations of Physics Letters*, v. 12, n. 6, p. 547–559, 1999.

VENTURA, D.; MARTINEZ, T. Quantum associative memory. *Information Sciences*, v. 124, n. 1, p. 273–296, 2000.

VERAS, T. L. de; ARAUJO, I. S. de; PARK, D. K.; SILVA, A. J. da. Circuit-based quantum random access memory for classical data with continuous amplitudes. *IEEE Transactions on Computers*, IEEE Computer Society, Los Alamitos, CA, USA, v. 70, n. 12, p. 2125–2135, dec 2021. ISSN 1557-9956.

VERAS, T. M. L. de; SILVA, L. D. da; SILVA, A. J. da. Double sparse quantum state preparation. *Quantum Information Processing*, v. 21, n. 6, p. 204, jun. 2022. ISSN 1573-1332.

WANG, S.; WANG, Z.; CUI, G.; SHI, S.; SHANG, R.; FAN, L.; LI, W.; WEI, Z.; GU, Y. Fast black-box quantum state preparation based on linear combination of unitaries. *Quantum Information Processing*, v. 20, n. 8, p. 270, 2021. ISSN 1573-1332.

WEI, T.-C.; GOLDBART, P. M. Geometric measure of entanglement and applications to bipartite and multipartite quantum states. *Physical Review A*, v. 68, n. 4, p. 042307, 2003. ISSN 1050-2947.

WIEBE, N.; BRAUN, D.; LLOYD, S. Quantum algorithm for data fitting. *Phys. Rev. Lett.*, American Physical Society, v. 109, p. 050505, Aug 2012.

WIEBE, N.; GRANDADE, C. Can small quantum systems learn. *Quantum Info. Comput.*, Rinton Press, Incorporated, Paramus, NJ, v. 17, n. 7–8, p. 568–594, jun 2017. ISSN 1533-7146.

WIEBE, N.; KAPOOR, A.; SVORE, K. M. Quantum deep learning. arXiv, 2014.

WIEBE, N.; KAPOOR, A.; SVORE, K. M. Quantum perceptron models. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2016. (NIPS'16), p. 4006–4014. ISBN 9781510838819.

WILLIAMS, C. P. Quantum gates. In: _____. *Explorations in Quantum Computing*. London: Springer London, 2011. p. 51–122. ISBN 978-1-84628-887-6.

WILLSCH, D.; NOCON, M.; JIN, F.; RAEDT, H. D.; MICHIELSEN, K. Gate-error analysis in simulations of quantum computers with transmon qubits. *Phys. Rev. A*, American Physical Society, v. 96, p. 062302, Dec 2017.

WOERNER, S.; EGGER, D. J. Quantum Risk Analysis. *npj Quantum Information*, v. 5, n. 1, p. 15, 2019.

WOSSNIG, L.; ZHAO, Z.; PRAKASH, A. Quantum linear system algorithm for dense matrices. *Phys. Rev. Lett.*, American Physical Society, v. 120, p. 050502, 2018.

XIN, T.; WEI, S.; CUI, J.; XIAO, J.; ARRAZOLA, I. n.; LAMATA, L.; KONG, X.; LU, D.; SOLANO, E.; LONG, G. Quantum algorithm for solving linear differential equations: Theory and experiment. *Phys. Rev. A*, American Physical Society, v. 101, p. 032307, Mar 2020.

ZHANG, X.-M.; YUNG, M.-H.; YUAN, X. Low-depth quantum state preparation. *Phys. Rev. Research*, American Physical Society, v. 3, p. 043200, 2021.

ZHAO, Z.; FITZSIMONS, J. K.; REBENTROST, P.; DUNJKO, V.; FITZSIMONS, J. F. Smooth input preparation for quantum and quantum-inspired machine learning. *Quantum Machine Intelligence*, v. 3, n. 1, p. 14, 2021. ISSN 2524-4914.

ZOUFAL, C.; LUCCHI, A.; WOERNER, S. Quantum generative adversarial networks for learning and loading random distributions. *npj Quantum Information*, Nature Publishing Group, v. 5, n. 1, p. 1–9, 2019.

APÊNDICE A – A DIVIDE-AND-CONQUER ALGORITHM FOR QUANTUM STATE PREPARATION

www.nature.com/scientificreports

scientific reports

Check for updates

OPEN

A divide-and-conquer algorithm for quantum state preparation

Israel F. Araujo[1], Daniel K. Park[2], Francesco Petruccione[3,4,5]✉ & Adenilton J. da Silva[1]

Advantages in several fields of research and industry are expected with the rise of quantum computers. However, the computational cost to load classical data in quantum computers can impose restrictions on possible quantum speedups. Known algorithms to create arbitrary quantum states require quantum circuits with depth *O*(*N*) to load an *N*-dimensional vector. Here, we show that it is possible to load an *N*-dimensional vector with exponential time advantage using a quantum circuit with polylogarithmic depth and entangled information in ancillary qubits. Results show that we can efficiently load data in quantum devices using a divide-and-conquer strategy to exchange computational time for space. We demonstrate a proof of concept on a real quantum device and present two applications for quantum machine learning. We expect that this new loading strategy allows the quantum speedup of tasks that require to load a significant volume of information to quantum devices.

The development of quantum computers can dramatically reduce the time to solve certain computational tasks[1]. However, in practical applications, the cost to load the classical information in a quantum device can dominate the asymptotic computational cost of the quantum algorithm[2,3]. Loading information into a device is a common task in computer science applications. For instance, deep neural networks[4] learning algorithms run in specialized hardware[5], and the computational cost to transfer the information needs to be considered in the total computational cost as data loading can dominate the training time on large-scale systems[6]. In classical devices, we can use the loaded information several times while we do not erase it. The situation is not the same in quantum devices because of the no-cloning theorem[7], noisy quantum operations[8], and the decoherence of quantum information[9]. The no-cloning theorem shows that it is not possible to perform a copy of an arbitrary quantum state. When a quantum operation is applied, its input is transformed or is destroyed (collapsed). Even if we represent the information in a basis state that we can copy, the noisy operations and decoherence will corrupt the stored state, and it will be necessary to reload the information from the classical to the quantum device.

Loading an input vector $\vec{x} = (x_0, \ldots, x_{N-1})$ to the amplitudes of a quantum system corresponds to create the state with $\log_2(N)$ quantum bits described in Eq. (1).

$$x_0|0\rangle + \cdots + x_{N-1}|N-1\rangle \tag{1}$$

Circuits to load an N-dimensional classical unit vector in quantum devices use $n = \log_2(N)$ qubits and have an exponential depth in relation to the number of qubits (or polynomial in the data size)[10–13].

Here we propose a new format of data encoding. Namely, we load an N-dimensional vector in probability amplitudes of computational basis state with entangled information in ancillary qubits as

$$x_0|0\rangle|\psi_0\rangle + \cdots + x_{N-1}|N-1\rangle|\psi_{N-1}\rangle, \tag{2}$$

where $|\psi_j\rangle$ are unit vectors. We propose an algorithm to load an N-dimensional vector in a quantum state as shown in Eq. (2) using a circuit with $O(\log_2^2(N))$ depth and $O(N)$ qubits. The devised method is based on quantum forking[13,14] and uses a divide-and-conquer strategy[15]. The circuit depth is decreased at the cost of increasing the circuit width and creating entanglement between data register qubits and an ancillary system. Thus when the data register is considered alone (i.e. by tracing out the ancilla qubits), the resulting state is mixed and not equal to the pure state shown in Eq. (1). However, it is important to note that in Eq. (2) the classical data is still encoded as probability amplitudes of an orthonormal basis set. Useful applications can be constructed based on this, and we provide two example applications in machine learning and statistical analysis.

The divide-and-conquer paradigm is used in efficient algorithms for sorting[16], computing the discrete Fourier transform[17], and others[15]. The main idea is to divide a problem into subproblems of the same class and combine the

[1]Centro de Informática, Universidade Federal de Pernambuco, Recife, Pernambuco, Brazil. [2]Sungkyunkwan University Advanced Institute of Nanotechnology, Suwon 16419, South Korea. [3]School of Electrical Engineering, KAIST, Daejeon 34141, Republic of Korea. [4]Quantum Research Group, School of Chemistry and Physics, University of KwaZulu-Natal, Durban, KwaZulu-Natal 4001, South Africa. [5]National Institute for Theoretical Physics (NITheP), Durban, KwaZulu-Natal 4001, South Africa. ✉email: petruccione@ukzn.ac.za

solutions of the subproblems to obtain the solution of the original problem. The circuit based divide-and-conquer state preparation algorithm has computational cost $O(N)$ and the total complexity time is $O_c(N) + O_q(\log_2^2(N))$, where $O_c(N)$ is classical pre-computation time to create the quantum circuit that will load the information in the quantum device and $O_q(\log_2^2(N))$ is the depth of the quantum circuit. With the supposition that we will load the input vector $m \gg N$ times, the amortized computational time to load the real vector is $O_q(\log_2^2(N))$. The modified version of the loading problem allows an exponential advantage in the depth of the quantum circuit using $O(N)$ qubits.

The remainder of this paper is organized into 3 sections. "Transformation of quantum states" section reviews one of the standard methods for loading information in a quantum device using controlled rotations[10], which we set out to modify to reduce its quantum circuit depth exponentially. "Divide-and-conquer loading data" section shows the main result, a quantum circuit with depth $O(\log_2^2(N))$, and $O(N)$ qubits to load an N-dimensional vector in a quantum state with entangled information in the ancillary qubits. "Discussion" section presents the conclusion and possible future works.

Transformation of quantum states

In this section, we review a strategy for loading a real vector into the amplitudes of a quantum state using a sequence of controlled one-qubit rotations[10]. Given an N-dimensional vector x, where $n = \log_2(N)$ is an integer, we can create a circuit to load this vector in a quantum computer using Algorithm 1. The task of amplitude encoding (Algorithm 1) has two parts: (1) Function **gen_angles** (Line 1) finds angles to perform rotations that lead $|0\rangle_n \equiv |0\rangle^{\otimes n}$ to the state in Eq. (1), and (2) Function **gen_circuit** (Line 18) generates a quantum circuit from these rotations.

Algorithm 1: Amplitude encoding

input : A vector x with dimension $N = 2^n$
output : A quantum circuit to load x in the amplitudes of a quantum system

1 `gen_angles` (x) **:**
  input : A vector x with dimension $N = 2^n$
  output : Angles to generate the amplitude encoding circuit
2 **if** $size(x) > 1$ **then**
3 Create an auxiliary vector *new_x* with dimension $N/2$
4 **for** $k \leftarrow 0$ **to** $length(new\_x)$ **do**
5 new_x[k] = $\sqrt{|x[2k]|^2 + |x[2k+1]|^2}$
6 inner_angles = `gen_angles` (*new_x*)
7 Create a vector *angles* with dimension $N/2$
8 **for** $k \leftarrow 0$ **to** length(*new_x*) **do**
9 **if** $new\_x[k] \neq 0$ **then**
10 **if** $x[2k] > 0$ **then**
11 angles[k] = $2\,\mathrm{asin}\left(\frac{x[2k+1]}{new\_x[k]}\right)$
12 **else**
13 angles[k] = $2\pi - 2\,\mathrm{asin}\left(\frac{x[2k+1]}{new\_x[k]}\right)$
14 **else**
15 angles[k] = 0
16 angles = inner_angles + angles
17 **return** angles

18 `gen_circuit` (*angles*) **:**
  input : $N-1$ dimensional vector angles = `gen_angles` (x)
  output : Quantum circuit to load x in the amplitudes of a quantum system
19 circuit = quantum circuit with $n = \log_2(N)$ qubits q[0], ..., q[$n-1$]
20 **for** $k \leftarrow 0$ **to** $N-2$ **do**
21 j = level(k)
22 index(k, j, q)
23 CR_y(angle[k], [q[0], ..., q[j-1], q[j])
24 index(k, j, q)
25 **return** circuit

26 angles = `gen_angles` (x)
27 circuit = `gen_circuit` (*angles*)
28 **return** circuit

Function gen_angles (Algorithm 1, Line 1) divides the 2^n-dimensional input vector into 2^{n-1} 2-dimensional subvectors and creates a 2^{n-1}-dimensional vector *new_x* with the norms of the subvectors. While the size of *new_x* is greater than 1, the *new_x* vector is recursively passed as the input of function gen_angles . This procedure is described in lines 3 to 6 of Algorithm 1. An example of the inputs in the recursive calls with the initial input

$$(\sqrt{0.03}, \sqrt{0.07}, \sqrt{0.15}, \sqrt{0.05}, \sqrt{0.1}, \sqrt{0.3}, \sqrt{0.2}, \sqrt{0.1})$$

is presented in the binary tree named state-tree in Fig. 1b.

After the last recursive call of the function gen_angles , the algorithm starts to compute the vector angles. For each k between 0 and the size of vector *new_x*, we append an angle θ such that $\sin(\theta/2) = \frac{x[2k+1]}{new\_x[k]}$ and $\cos(\theta/2) = \frac{x[2k]}{new\_x[k]}$ to the vector angles. Lines 7 to 16 generate the vector angles in the recursive calls. For the input in Fig. 1b and using two decimal points the algorithm outputs angles = (1.98, 1.91, 1.43, 1.98, 1.05, 2.09, 1.23). The angles vector is used as a complete binary tree named angles-tree. For instance, with $\alpha_k = angles[k]$, the angles-tree created by gen_angles with an eight-dimensional input vector is described in Fig. 1a. Each call of gen_angles will perform $\log_2(N)$ recursive calls and the cost of each call for $k = 1, \ldots, \log_2(N)$ is $N/2^{k-1}$. The costs of the recursive calls to generate the angles-vector is $1 + 2 + 2^2 + \cdots + 2^{\log_2 N} = O(N)$.

Function gen_circuit (Algorithm 1, Line 18) receives the $N-1$ dimensional vector angles, generated by the function gen_angles with input x, and outputs a quantum circuit to load the vector x in the amplitudes of a quantum state. The state in level j of the tree-state in Fig. 1b can be constructed from the state in the level $j-1$ of the states-tree and controlled rotations from the level $j-1$ in angles-tree. The root of the angles-tree defines the first rotation and the algorithm follows a top-down approach where the rotation of angle $angle[k]$ is controlled by the qubits in range $[0, level(k))$ and is applied if the qubits $q[0], \ldots, q[level(k)-1]$ are in the state $|k - (2^{level(k)} - 1)\rangle$. With $\alpha_k = angle[k]$, the circuit to load an eight-dimensional input vector is described in Fig. 2. The computational cost to compute the angles and to generate the circuit is $O(N)$. The quantum circuit uses $O(N)$ multi-controlled gates that are applied sequentially and the circuit depth is $O(N)$. We have a $O(N)$ cost in the classical host machine and a $O(N)$ cost in the quantum device and spatial cost $O(\log_2(N))$. An amortized computational cost is $O(N)$ if we need to load the vector several times.

Divide-and-conquer loading data

The construction of the quantum state in the previous section starts in the root of state-tree $|0\rangle_n$ and build the states in each level of the state-tree in a top-down strategy until to build the state described by the last level of the state-tree. In this Section, we propose a divide-and-conquer load strategy, and the desired quantum state is built following a bottom-up strategy. First, we divide the input into bidimensional subvectors and load qubits corresponding to the normalized bidimensional subvectors. In the next steps, we generate the subvectors of the previous levels.

For instance, to load the state in the leafs of the state-tree in Fig. 1b, we load four one-qubit states

$$\frac{\sqrt{0.03}}{\sqrt{0.1}}|0\rangle + \frac{\sqrt{0.07}}{\sqrt{0.1}}|1\rangle, \frac{\sqrt{0.15}}{\sqrt{0.2}}|0\rangle + \frac{\sqrt{0.05}}{\sqrt{0.2}}|1\rangle, \frac{\sqrt{0.1}}{\sqrt{0.4}}|0\rangle + \frac{\sqrt{0.3}}{\sqrt{0.4}}|1\rangle \text{ and } \frac{\sqrt{0.2}}{\sqrt{0.3}}|0\rangle + \frac{\sqrt{0.1}}{\sqrt{0.3}}|1\rangle$$

representing the leafs of the state-tree. To load the two two-qubit states in the previous level, the single-qubit states are weighted with the value of their fathers, obtaining the state $|\psi_l\rangle$ representing the state in the half left part of the state-tree in Eq. (3) and the state $|\psi_r\rangle$ representing the state in the right part of the state-tree in Eq. (4).

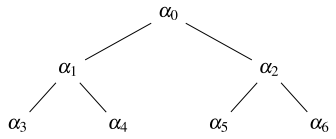
$$\begin{aligned} |\psi_l\rangle &= \frac{\sqrt{0.1}}{\sqrt{0.3}}|0\rangle\left(\frac{\sqrt{0.03}}{\sqrt{0.1}}|0\rangle + \frac{\sqrt{0.07}}{\sqrt{0.1}}|1\rangle\right) + \frac{\sqrt{0.2}}{\sqrt{0.3}}|1\rangle\left(\frac{\sqrt{0.15}}{\sqrt{0.2}}|0\rangle + \frac{\sqrt{0.05}}{\sqrt{0.2}}|1\rangle\right) \\ &= \frac{\sqrt{0.03}}{\sqrt{0.3}}|00\rangle + \frac{\sqrt{0.07}}{\sqrt{0.3}}|01\rangle + \frac{\sqrt{0.15}}{\sqrt{0.3}}|10\rangle + \frac{\sqrt{0.05}}{\sqrt{0.3}}|11\rangle \end{aligned} \tag{3}$$

$$\begin{aligned} |\psi_r\rangle &= \frac{\sqrt{0.4}}{\sqrt{0.7}}|0\rangle\left(\frac{\sqrt{0.1}}{\sqrt{0.4}}|0\rangle + \frac{\sqrt{0.3}}{\sqrt{0.4}}|1\rangle\right) + \frac{\sqrt{0.3}}{\sqrt{0.7}}|1\rangle\left(\frac{\sqrt{0.2}}{\sqrt{0.3}}|0\rangle + \frac{\sqrt{0.1}}{\sqrt{0.3}}|1\rangle\right) \\ &= \frac{\sqrt{0.1}}{\sqrt{0.7}}|00\rangle + \frac{\sqrt{0.3}}{\sqrt{0.7}}|01\rangle + \frac{\sqrt{0.2}}{\sqrt{0.7}}|10\rangle + \frac{\sqrt{0.1}}{\sqrt{0.7}}|11\rangle \end{aligned} \tag{4}$$

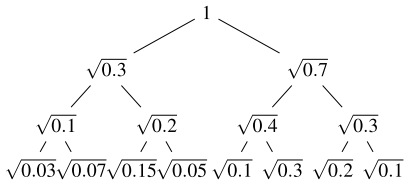
Combining states $|\psi_l\rangle$ and $|\psi_r\rangle$ weighted with the values of the state in the previous layer generates the desired quantum state described in Eq. (5).

$$\begin{aligned} \sqrt{0.3}|\psi_l\rangle + \sqrt{0.7}|\psi_r\rangle = &\sqrt{0.03}|000\rangle + \sqrt{0.07}|001\rangle + \sqrt{0.15}|010\rangle + \sqrt{0.05}|011\rangle \\ &+ \sqrt{0.1}|100\rangle + \sqrt{0.3}|101\rangle + \sqrt{0.2}|110\rangle + \sqrt{0.1}|111\rangle \end{aligned} \tag{5}$$

To load the classical data using this bottom-up approach we need to combine two m-qubits states $|\psi\rangle, |\phi\rangle$ and one one-qubit state $a|0\rangle + b|1\rangle$ as $a|0\rangle|\psi\rangle + b|1\rangle|\phi\rangle$ with a circuit that does not depend on the input states. Using the circuit in Fig. 3 with $m-1$ controlled-swap (CSWAP) operations, we generate the desired output in



(a) Binary tree angles vector generated by function `gen_angles` with an 8-dimensional input vector.



(b) Bottom up recursive calls of the function generate angles used to compute the α angles of Fig. 1a

Figure 1. Data representation of information in function generate angles.

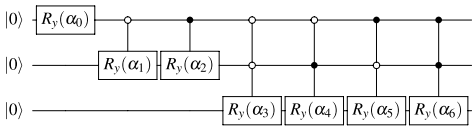


Figure 2. Circuit to load an 8 dimensional real vector in a quantum device.

the first m qubits, but with unit entangled information in the $m-1$ ancillary qubits. Namely, for the example with Fig. 3, the conventional amplitude encoding in the form of Eq. (1) would aim to prepare an m-qubit state $a|0\rangle|\psi\rangle + b|1\rangle|\phi\rangle$ while our method prepares $a|0\rangle|\psi\rangle|\phi\rangle + b|1\rangle|\phi\rangle|\psi\rangle$.

Loading complex data. The divide-and-conquer strategy can be generalized to load a complex vector $(|x_0|e^{i\omega_0}, |x_1|e^{i\omega_1}, \ldots, |x_{N-1}|e^{i\omega_{N-1}})$ into the probability amplitudes of a quantum state as

$$|x_0|e^{i\omega_0}|0\rangle|\psi_0\rangle + \cdots + |x_{N-1}|e^{i\omega_{N-1}}|N-1\rangle|\psi_{N-1}\rangle. \tag{6}$$

To explain the process, we introduce two parameters used in Ref.[18]

$$\lambda_{j,v} = \sum_{l=1}^{2^{v-1}} (\omega_{(2j-1)2^{v-1}+l} - \omega_{(2j-2)2^{v-1}+l})/2^{v-1} \text{ and } \beta_{j,v} = \sqrt{\sum_{l=1}^{2^{v-1}} |a_{(2j-1)2^{v-1}+l}|^2} / \sqrt{\sum_{l=1}^{2^{v}} |a_{(j-1)2^{v}+l}|^2},$$

where $v = 1, 2, \ldots, n$ is the level of the tree in reverse order (i.e. 1 for the leaf nodes and n for the root node) and $j = 1, 2, \ldots, 2^{n-v}$ is the qubit index in the layer v. Next, one needs $N/2$ one-qubit states corresponding to the leaf nodes of the state-tree (see Fig. 1b for example) to be prepared as

$$|\psi_{j,1}\rangle = e^{-i\frac{\lambda_{j,1}}{2}} \sqrt{1 - |\beta_{j,1}|^2}|0\rangle + e^{i\frac{\lambda_{j,1}}{2}} \beta_{j,1}|1\rangle. \tag{7}$$

To load the states in the previous levels (represented by v on the expression below), the states of the current level ($v-1$, since v is in reverse order) are weighted with the values of their parents, obtaining the state

$$|\psi_{j,v}\rangle = e^{-i\frac{\lambda_{j,v}}{2}} \sqrt{1 - |\beta_{j,v}|^2}|0\rangle|\psi_{2j-1,v-1}\rangle + e^{i\frac{\lambda_{j,v}}{2}} \beta_{j,v}|1\rangle|\psi_{2j,v-1}\rangle. \tag{8}$$

After recursively updating the state $|\psi_{j,v}\rangle$ for $v = 2, \ldots, n$ and $j = 1, 2, \ldots .2^{n-v}$, the desired quantum state is generated as

$$|\psi_{1,n}\rangle = |x_0|e^{i\omega_0}|0\rangle + |x_1|e^{i\omega_1}|1\rangle + \cdots + |x_{N-1}|e^{i\omega_{N-1}}|N-1\rangle. \tag{9}$$

Combining two states at children nodes in the state-tree as shown in Eq. (8) is done with controlled-swap operations as explained in the previous section, and we will need N qubits with entangled auxiliary qubits to generate the state in Eq. (6). Thus the only modification in the quantum circuit is the introduction of the $R_z(\lambda_{j,v})$ rotations to set the phases, following the R_y rotations. The pseudocode for generating the angles for the R_z rotations is given in Algorithm 2.

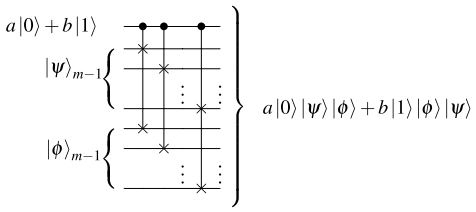


Figure 3. Combining states with controlled-swap operations.

Algorithm 2: Generate angles for R_z rotations

```
gen_angles_z (x) :
    input  : A vector x of dimension N = 2^n containing the phases of the input vector.
    output : Phases to generate the amplitude encoding circuit
    if size(x) > 1 then
        Create an auxiliary vector new_x with dimension N/2
        for k ← 0 to length(new_x) do
            new_x[k] = (x[2k] + x[2k+1])/2
        inner_angles_z = gen_angles_z (new_x)
        Create a vector angles_z with dimension N/2
        for k ← 0 to length(new_x) do
            angles_z[k] = x[2k+1] − x[2k]
        angles_z = inner_angles_z + angles_z
        return angles_z
```

Algorithm 3 presents the complete pseudocode for the divide-and-conquer state preparation algorithm. The for loop in line 2 initializes the qubit $q[k]$ with the value $R_y(\alpha_k)$. After this step, the qubits with index $k > \lfloor (N-1)/2 \rfloor$ (in the leaf of the angle tree) are normalized versions of the states in the leafs of the state-tree. The next subroutine with R_z rotations (Line 4 to Line 5) is used to encode phase information. Line 6 calculates the index of the first angle that has a right children in the angle-tree data structure. The while loop starting at line 7 combines the states generated in the subtree rooted in the angle α_{actual}. To combine the states, we first apply a cswap(q[actual], q[left_child], q[right_child]), and then we update the values of left and right child with the value of their left child and apply another cswap(q[actual], q[left_child], q[right_child]) while the left_child and right_child have valid values. With the input described by the angle-tree in Fig. 1a, Algorithm 3 generates the circuit described in Fig. 4.

Algorithm 3: Divide-and-conquer load circuit

```
input  : N − 1 dimensional vector angle = gen_angles (abs(x))
input  : N − 1 dimensional vector angle_z = gen_angles_z (phase(x))
output : Quantum circuit to load x in the amplitudes of a quantum system entangled with ancillary qubits
1  circuit = quantum circuit with N − 1 qubits q[0], ..., q[N − 2]
2  for k ← 0 to N − 2 do
3      R_y(angle[k], q[k])
4  for k ← 0 to N − 2 do
5      R_z(angle_z[k], q[k])
6  actual = parent(N − 2)
7  while actual ≥ 0 do
8      left_index = left(actual)
9      right_index = right(actual)
10     while right_index < N − 1 do
11         cswap(q[actual], q[left_index], q[right_index])
12         left_index = left(left_index)
13         right_index = left(right_index)
14     actual = actual - 1
```

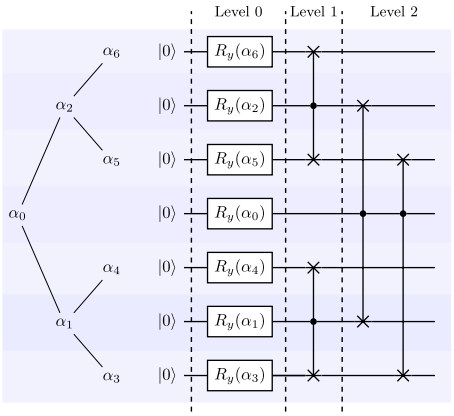


Figure 4. Rotated angle-tree and a circuit generated by the divide-and-conquer strategy described in Algorithm 3. The quantum bit $q[k]$ in the circuit is aligned with the angle $\alpha[k]$ in the angle-tree, this organization allows to draw the quantum gates in each layer in parallel. In this example, the desired state is stored in qubits $q[0]$, $q[1]$ and $q[3]$ to generate the quantum state with entangles ancilla as in Eq. (2).

The process to load each state in the same layer of the state tree can be performed in parallel, because the control swap gates use different qubits. The controls are qubits in one layer of the angle-tree and targets are qubits in their subtrees. Layer with height k contributes to the depth of the circuit with the tree height minus height of the layer. The circuit will have a depth of $O(1+2+\cdots+\log_2(N)-1)$ with an overall depth in order $O(\log_2^2(N))$. This result is stated in Theorem 1.

Theorem 1 *Algorithm 3 generates a quantum circuit with depth* $O(\log_2^2(N))$.

Orthonormal ancillary. The ancillary states $|\psi_0\rangle,\ldots,|\psi_{N-1}\rangle$ in Eq. (2) are not necessarily orthogonal to each other, but we can modify the divide-and-conquer state preparation adding label qubits to ensure orthonormality of the ancillary states with the addition of label quantum register with $\log_2(N)$ qubits. The label register is prepared in $|0\rangle^{\otimes \log_2(N)}$, and $\log_2(N)$ controlled-NOT (CNOT) gates are applied to the label qubits, each controlled by a data qubit. With this modification, the final state becomes

$$x_0|0\rangle|\psi_0\rangle|0\rangle+\cdots+x_{N-1}|N-1\rangle|\psi_{N-1}\rangle|N-1\rangle=\sum_{k=0}^{N-1}x_k|k\rangle|\tilde{\psi}_k\rangle, \tag{10}$$

where $\{|\tilde{\psi}_k\rangle\}_{k=0}^{N-1}=\{|\psi_k\rangle|k\rangle\}_{k=0}^{N-1}$ is a set of orthonormal states.

Experiments. To evaluate the proposed method we perform two sets of experiments. In the first set of experiments, we use a quantum computing simulator and a NISQ computer to show as a proof of concept that the proposed method can be applied in near future. In the second set of experiments, we compare the depth of the circuits generated by the proposed method and other state preparation algorithms[10,11] with a random input.

Proof of concept with a NISQ device. In this experiment we load a four-dimensional data into a two qubit state $|\psi\rangle=\sqrt{0.6}|00\rangle+\sqrt{0.2}|01\rangle+\sqrt{0.1}|10\rangle+\sqrt{0.1}|11\rangle$ in a NISQ device as a proof of concept. For this experimental validation, we chose dimension of data to be small to be compatible with currently available quantum devices, although the time advantage of the proposed method will manifest when a large number of qubits are required for loading high-dimensional data. We use qubits 1, 2 and 3 of the ibmq_rome device. The CNOT error rates were 8.832e-3 (qubits 1 and 2) and 8.911e-3 (qubits 2 and 3). The single-qubit error was in the order of 1e-4.

Figure 5a presents the output of the experiment with 1024 executions using a quantum device simulator and the Rome quantum device. The Rome NISQ device has an output very close to the expected result. The circuit used to obtain this result is described in Fig. 5b, where c is a classical register. We remove the last CNOT of the controlled operation because the qubit 2 will be discarded. The resulting circuit has 10 CNOT operators because a quantum swap was necessary to run this circuit in the real quantum device with a limited qubit connectivity. The circuit used in the quantum device is described in Fig. 7.

Circuit depth. The main difference between the divide-and-conquer state preparation and previous approaches is an exchange between circuit depth by circuit width. Table 1 presents the depth of the circuits generated using the proposed strategy, implementation of a version of[11] available at[19] and a non optimized version of the algorithm described in[10]. The proposed strategy and[10] implementation are publicly available. The implementation of the proposed method shows its theoretical asymptotic time advantage to load a vector when the dimension is larger than 32. The proposed method has two main disadvantages: the linear number of qubits in relation to the logarithmic number in other methods, and the information entangled in the ancillary qubits.

The higher depth of circuits using the divide-and-conquer strategy with small vectors occurs because of the use of three-qubits gates to combine the vectors. In other works, it is only necessary to use $O(n)$ qubits to load a 2^n-dimensional vector while requiring sequential applications of $O(2^n)$ n-controlled gates. To improve the performance of the divide-and-conquer loading strategy and to reduce the number of qubits one can combine algorithm[11] with the divide-and-conquer strategy. Instead of divide the vector in parts with size 2, we can divide the vector in parts with size k (equal to a power of 2), load the normalized k-dimensional vectors using a sequential algorithm and combine the small vectors with the divide-and-conquer approach.

Example applications. *Hierarchical quantum classifier.* This section compares the divide-and-conquer algorithm with two other approaches in which input data encoding in a quantum state can be achieved to initialize a quantum circuit, namely qubit encoding and amplitude encoding. In the former, data is encoded in the amplitudes of individual qubits in a fully separable state, performed using single-qubit rotations[20]. In the later, data is encoded in the amplitudes of an entangled state[11,18], similarly to the divide-and-conquer. We use the accuracy of a quantum variational classifier as a metric to evaluate the state preparations. The divide-and-conquer algorithm is expected to produce results similar to the amplitude encoding. The results of the classifier using qubit encoding are also presented for completeness, albeit our main objective is to compare the divide-and-conquer and amplitude encoding schemes.

The classifier is based on a tree-like circuit known as tree tensor network (TTN)[20]. This choice is based on the fact that tensor networks can represent both neural networks and quantum circuits, acting as a link between these fields[21,22]. Initially, it applies a set of two-qubit unitaries to each pair of qubits from the initial state, discarding one output from each unitary, leaving half the number of qubits left for the next layer. The process is repeated until only one qubit remains. Multiple measurements are carried on this last qubit to approximate the expectation value.

Following Grant et al.[20], we built the circuits using single-qubit rotations around the y-axis of the Bloch sphere, denoted by $R_y(\theta)$, and CNOT gates, composing two-qubit unitary blocks $CNOT \cdot (R_y(\theta_0) \otimes R_y(\theta_1))$. The single-qubit rotation angle θ is subject to training by some optimization procedure. Examples of the resulting circuits are represented in Fig. 6a–c.

We follow the general classical-quantum hybrid approach in which the optimization procedure is processed on a classical computer to determine a set of parameters, i.e. rotation angles for the $R_y(\theta)$ operation, for the parametrized quantum circuit. The quantum device prepares a quantum state as prescribed by the circuit pipeline and performs measurements. The measurement outcomes are processed by a classical device to generate a forecast, using it to update the model parameters via a learning algorithm. This whole process is repeated towards the goal.

Four datasets were used in this work: Iris, Haberman's Survival, Banknote Authentication[23], and Pima Indians Diabetes[24]. Three binary datasets were extracted from the original Iris dataset (paired combinations of the original three classes). Mean test accuracy and one standard deviation are computed on ten random initializations for each dataset and encoding. The simulation results are presented in Table 2, where the test accuracy of the qubit and amplitude encodings are compared against the results obtained using the divide-and-conquer encoding.

The results show similar classification accuracy for all encodings, favoring qubit encoding due to the greater number of circuit parameters for the optimization. The main advantage of divide-and-conquer encoding over qubit encoding is the representation of encoded data in a quantum state of a reduced number of qubits, $\log_2(N)$, compared to the initial state $N-1$. This also results in a lower depth classifier. Moreover, when the data is given by qubit encoding, TTN circuits can be evaluated efficiently using classical techniques[20]. This is not true when the input data is amplitude encoded. The advantage over amplitude encoding is a lower depth encoding circuit for $N \geq 64$ (Table 1).

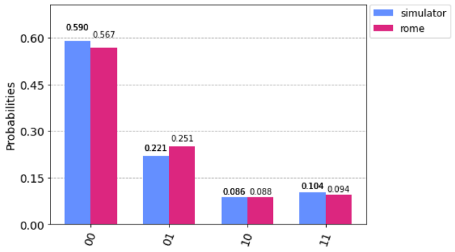
To verify that the above comparison of the models is appropriate, a nonparametric statistical test was employed. We used the Wilcoxon paired signed-rank test[25] with $\alpha = 0.05$ to check whether there exist significant differences between the classification performances of compared encoders over the chosen datasets. As expected, we verified that amplitude encoding and divide-and-conquer encoding are statistically equivalent for all datasets.

Swap test. Some metric between two data set encoded as $\sum_i x_i|i\rangle = \sum_i |\tilde{x}_i\rangle$ and $\sum_j y_j|j\rangle = \sum_j |\tilde{y}_j\rangle$ can be calculated with the state prepared by the divide-and-conquer state preparation and the swap test. The required state is

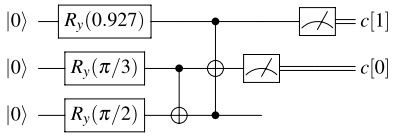
$$|0\rangle \sum_{ij} |\tilde{x}_i\rangle|\tilde{y}_j\rangle|\psi_i\rangle|\phi_j\rangle, \tag{11}$$

where $\sum_i |\tilde{x}_i\rangle|\psi_i\rangle$ and $\sum_j |\tilde{y}_i\rangle|\phi_j\rangle$ are prepared by the encoding scheme explained in Sec. 3.2 so as to make the ancillary states orthonormal.

After applying the swap test circuit to the above state, i.e. the Hadamard on the first (ancilla) qubit, the swap operation between the test register and the data register controlled by the ancilla qubit, and the Hamadard on the first qubit, one obtains



(a) Output of the circuit in Fig. 5b with 1024 runs in a simulator (blue) and ibmq_rome (red).



(b) Circuit to prepare the quantum state $|\psi\rangle = \sqrt{0.6}\,|00\rangle + \sqrt{0.2}\,|01\rangle + \sqrt{0.1}\,|10\rangle + \sqrt{0.1}\,|11\rangle$ with an ancilla qubit before compilation. The measurement outcomes are stored in classical registers denoted by $c[0]$ and $c[1]$.

Figure 5. Proof of concept experiment with a IBM quantum device (ibmq_rome) on the cloud platform.

n	dc depth	dc width	[11] depth	[11] width	[10] depth	[10] width
4	12	4	3	3	5	3
8	31	8	17	4	53	4
16	58	16	47	5	277	5
32	93	32	105	6	1237	6
64	136	64	239	7	5205	7
128	187	128	493	8	21333	8
256	246	256	982	9		9
512	313	512	2025	10		10
1024	388	1024	4009	11		11

Table 1. A comparison between Refs.[10,11] and divide-and-conquer strategy to load a n-dimensional real vector into a quantum computer.

$$\frac{1}{2}\left(|0\rangle\sum_{ij}\left(|\tilde{x}_i\rangle|\tilde{y}_j\rangle + |\tilde{y}_j\rangle|\tilde{x}_i\rangle\right)|\psi_i\rangle|\phi_j\rangle + |1\rangle\sum_{ij}\left(|\tilde{x}_i\rangle|\tilde{y}_j\rangle - |\tilde{y}_j\rangle|\tilde{x}_i\rangle\right)|\psi_i\rangle|\phi_j\rangle\right). \tag{12}$$

Now, when the σ_z measurement is performed on the ancilla qubit, the probability to measure $z = \pm 1$, i.e. $z = +1$ if the ancilla qubit is $|0\rangle$ and $z = -1$ if the ancilla qubit is $|1\rangle$, is

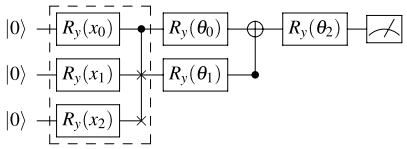
$$\begin{aligned}\Pr(z=\pm 1) &= \frac{1}{4}\sum_{ijkl} 2\left(\langle\tilde{x}_k|\tilde{x}_i\rangle\langle\tilde{y}_l|\tilde{y}_j\rangle \pm \langle\tilde{y}_l|\tilde{x}_i\rangle\langle\tilde{x}_k|\tilde{y}_j\rangle\right)\langle\psi_k|\psi_i\rangle\langle\phi_l|\phi_j\rangle \\ &= \frac{1}{2}\left(\sum_{ij}\langle\tilde{x}_i|\tilde{x}_i\rangle\langle\tilde{y}_j|\tilde{y}_j\rangle \pm |\langle\tilde{y}_j|\tilde{x}_i\rangle|^2\right) \\ &= \frac{1 \pm \sum_{ij}|\langle\tilde{y}_j|\tilde{x}_i\rangle|^2}{2}.\end{aligned} \tag{13}$$

Therefore, measuring the expectation value of σ_z on the ancilla qubit yields

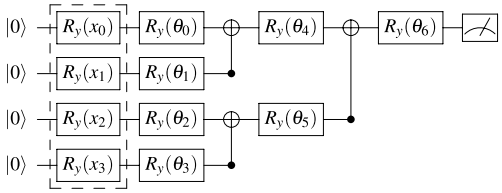
$$\sum_{ij}|\langle\tilde{y}_j|\tilde{x}_i\rangle|^2 = \sum_i |x_i y_i|^2. \tag{14}$$

Several measures in statistics can be derived from the above result. First, by setting $|x_i|^2$ to be the possible values of a discrete random variable $X : \Omega \to \mathbb{R}$ with the probability $\Pr(X = |x_i|^2) = |y_i|^2$, the above equation becomes an expectation value of the random variable X. The above equation can be also viewed as the second moment of a discrete random variable X, i.e. $E(X^2)$, with the probability $\Pr(X = x_i) = |y_i|^2$. This can be used to calculate the variance of X given $E(X)^2$. Alternatively, the above equation can be viewed as $E(XY)$ of two uniformly-distributed discrete random variables X and Y that satisfy $\Pr(X = |x_i|^2) = \Pr(Y = |y_i|^2) = 1/N$. This can be used with $E(X) = \sum_i^N |x_i|^2/N = E(Y)\sum_i^N |y_i|^2/N = 1/N$ to calculate the covariance, $E(XY) - E(X)E(Y)$.

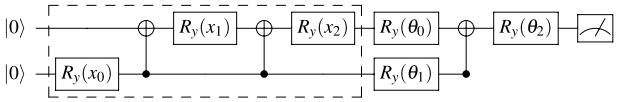
The idea above can be extended for calculating the covariance of two discrete random variables X and Y with any known probability distribution. Let possible outcomes of X and Y be $(|x_0|^2, \ldots, |x_{N-1}|^2)$ and



(a) TTN classifier using divide-and-conquer encoding. Gates in the highlighted area encode each element of a length 3 classical data vector in the amplitudes of the entangled state composed by the first two qubits.



(b) TTN classifier using qubit encoding. Rotations in the highlighted area encode each element of a length 4 classical data vector in the amplitudes of individual qubits.



(c) TTN classifier using amplitude encoding. Gates in the highlighted area encode each element of a length 3 classical data vector in the amplitudes of circuit's entangled state.

Figure 6. TTN classifier with (**a**) divide-and conquer encoding, (**c**) amplitude encoding and (**b**) qubit encoding.

$(|y_0|^2, \ldots, |y_{N-1}|^2)$, respectively, and the probability distribution be $(p_0^x, \ldots, p_{N-1}^x)$ and $(p_0^y, \ldots, p_{N-1}^y)$, respectively. Then the divide-and-conquer algorithm can be used to prepare a state

$$|0\rangle \sum_{ijkl} |\tilde{p}_i^x\rangle |\tilde{x}_j\rangle |\tilde{p}_k^y\rangle |\tilde{y}_l\rangle |\psi_{ijkl}\rangle, \tag{15}$$

where $|\tilde{p}_i^x\rangle = \sqrt{p_i^x}|i\rangle$, $|\tilde{p}_k^y\rangle = \sqrt{p_k^y}|k\rangle$, $|\tilde{x}_j\rangle = x_j|j\rangle$, $|\tilde{y}_l\rangle = y_l|l\rangle$, and $|\psi_{ijkl}\rangle$ is the orthonormal ancillary state as before. Now, the swap test circuit is applied with a small modification such that $3n$ controlled-swap gates are applied to transform $|\tilde{p}_i^x\rangle|\tilde{x}_j\rangle|\tilde{p}_k^y\rangle|\tilde{y}_l\rangle$ to $|\tilde{x}_j\rangle|\tilde{p}_k^y\rangle|\tilde{y}_l\rangle|\tilde{p}_i^x\rangle$ when the ancilla qubit for the swap test is $|1\rangle$. Measuring the expectation value of the σ_z observable on the ancilla qubit yields

$$\sum_{ijkl} \langle\tilde{p}_i^x|\tilde{x}_j\rangle\langle\tilde{x}_j|\tilde{p}_k^y\rangle\langle\tilde{p}_k^y|\tilde{y}_l\rangle\langle\tilde{y}_l|\tilde{p}_i^x\rangle = \sum_i p_i^x p_i^y |x_i|^2 |y_i|^2 = E(XY). \tag{16}$$

$E(X)$ and $E(Y)$ can be calculated from the swap test algorithm presented in the beginning of this section, which provided Eq. (14), by choosing the input vectors appropriately.

The total time complexity for the aforementioned quantum algorithms is still $O_q(\log_2^2(N))$, since the swap test only requires additional $O(\log_2(N))$ controlled-swap gates. The quantum speedup can be manifested when constructing a covariance matrix for two multivariate random variables $\mathbf{X}$ and $\mathbf{Y}$, each containing m discrete random variables of size N. Since there are m^2 entries in the matrix, the classical time cost is $O_c(Nm^2)$, while the quantum approach requires $O_c(N) + O_q(\log_2^2(N)m^2)$.

Discussion

One of the major open problems for practical applications of quantum computing is to develop an efficient means to encode classical data in a quantum state[3]. Most quantum algorithms do not present advantages in loading data[2]. The method proposed in this work fills this gap by proposing a new quantum state preparation paradigm, which can complement or enhance the known methods, such as qubit encoding and amplitude encoding. Our approach was based on the Möttönen et al. algorithm[10] and a divide-and-conquer approach using controlled swap gates and ancilla qubits. With this modification, we obtain an exponential quantum speedup in time to load a N-dimensional real vector in the amplitude of a quantum state with a quantum circuit of depth $O(\log_2^2(N))$

Dataset	Classes	Encoding		
		Qubit	Amplitude	Divide-and-conquer
Haberman	0 or 1	60.33 ± 2.02	59.02 ± 0.00	59.02 ± 0.00
Banknote	0 or 1	91.28 ± 3.11	87.15 ± 0.74	87.45 ± 1.12
Pima	0 or 1	77.19 ± 2.08	70.78 ± 1.88	71.11 ± 1.79
Iris	0 or 1	100 ± 0.00	100 ± 0.00	100 ± 0.00
	0 or 2	100 ± 0.00	100 ± 0.00	100 ± 0.00
	1 or 2	98.50 ± 2.42	93.00 ± 2.58	93.00 ± 2.58

Table 2. Mean test accuracy and one standard deviation for TTN classifiers with ten different random parameter initializations. Three binary datasets were extracted from the original Iris dataset.

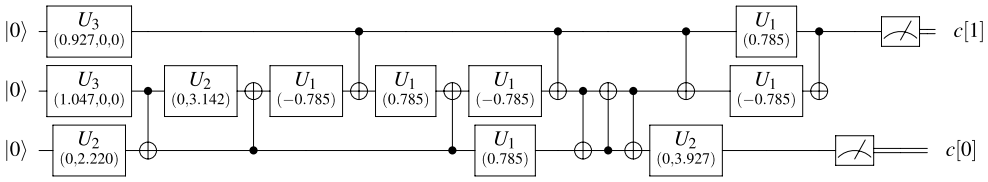


Figure 7. The transpiled circuit of the divide-and-conquer state preparation circuit in Fig. 5b in accordance with the physical qubit layout of the ibmq_rome device. The gates U_1, U_2, and U_3 are physical single-qubit gates of IBM Quantum Experience that take in one, two, and three parameters, respectively. The measurement outcomes are stored in classical registers denoted by $c[0]$ and $c[1]$.

and space $O(N)$. The exponential speedup to load data in quantum devices has a potential impact on speeding up the solution of problems in quantum machine learning and other quantum algorithms that need to load data from classical devices.

The speedup is achieved at the cost of using ancilla qubits that are entangled to the data register qubits. However, we showed that some interesting problems such as quantum supervised machine learning and statistical analysis can be performed with the input quantum state given by our method. The tradeoff between time and space complexities that our method provides is favorable when increasing the circuit width is easier than increasing the circuit depth, which is a likely scenario to occur during the development of near-term quantum devices.

We demonstrated the proof-of-principle using the IBM quantum cloud platform to verify the validity and the feasibility of our method. Furthermore, the numerical experiments showed that the new encoding method offers advantages, reducing complexity and computational resources when applied in conjunction with existing algorithms. Our perspective is that these advantages will extend to other cases.

This work leaves some open questions. What are other problems that can be solved with a divide-and-conquer quantum strategy? What are the implications to efficiently load a quantum vector with entangled information in the ancillary qubits for machine learning? And how to combine sequential with parallel strategies to create a robust algorithm with respect to input size? Also, finding an efficient means to uncompute the ancillary information remains as an interesting future work that will broaden the applicability of our method.

Methods

We performed the proof of concept experiment with a publicly available IBM quantum device consisting of five superconducting qubits, named as ibmq_rome. The quantum circuit used in this experiment is depicted in Fig. 5b. The circuit in Fig. 5b is compiled to the physical qubit layout of ibmq_rome and the resulting circuit is depicted in Fig. 7 that is executed 1024 times to obtain the data used to generate Fig. 5a. We used the quantum information science kit (qiskit). Python implementation of gen_angles and Algorithm 3 are used to generate the quantum circuit in Figs. 4 and 5b.

The depth of the quantum circuits for state preparations described in Table 1 is obtained using a python implementation of Algorithm 3, the qiskit implementation of[11] and a non-optimized version of the algorithm[10] available at the GitHub repository. For each input size we generated a random vector used for all methods. In these first two set of experiments we used qiskit version 0.14.1 and python version 3.7.7.

In "Hierarchical quantum classifier" section, simulations of the hybrid classification algorithms were performed using Xanadu's Pennylane[26] default qubit plugin state simulator. We used 2/10 of the datasets as a test set, 2/10 as a validation set, and the remaining as a training set. As preparation for qubit encoding, each data vector element of all datasets was re-scaled within the range of $[0, \pi]$. Also, for amplitude encoding and divide-and-conquer encoding, the data vectors were normalized. Our simulation employs the Adaptative Moment Estimation (Adam) for the optimization process[27] with a learning rate of 0.1 and a batch size of 1/10 of the training set size. Training stops when validation set accuracy does not increase for 30 consecutive tests or 200 iterations is reached.

Data availibility

The site https://www.cin.ufpe.br/~ajsilva/dcsp/ contains all the data and software generated during the current study.

Received: 24 September 2020; Accepted: 2 March 2021
Published online: 18 March 2021

References

1. Arute, F. *et al.* Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
2. Biamonte, J. *et al.* Quantum machine learning. *Nature* **549**, 195–202 (2017).
3. Aaronson, S. Read the fine print. *Nat. Phys.* **11**, 291–293. https://doi.org/10.1038/nphys3272 (2015).
4. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).
5. Dong, Y. *et al.* Stochastic quantization for learning accurate low-bit deep neural networks. *Int. J. Comput. Vis.* **127**, 1629–1642 (2019).
6. Yang, C.-C. & Cong, G. Accelerating data loading in deep neural network training. In *2019 IEEE 26th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, 235–245 (IEEE, 2019).
7. Wootters, W. K. & Zurek, W. H. A single quantum cannot be cloned. *Nature* **299**, 802–803 (1982).
8. Preskill, J. Quantum computing in the nisq era and beyond. *Quantum* **2**, 79 (2018).
9. Hughes, R. J., James, D. F., Knill, E. H., Laflamme, R. & Petschek, A. G. Decoherence bounds on quantum computation with trapped ions. *Phys. Rev. Lett.* **77**, 3240 (1996).
10. Möttönen, M., Vartiainen, J. J., Bergholm, V. & Salomaa, M. M. Transformation of quantum states using uniformly controlled rotations. *Quant. Inf. Comput.* **5**, 467–473 (2005).
11. Shende, V. V., Bullock, S. S. & Markov, I. L. Synthesis of quantum-logic circuits. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.* **25**, 1000–1010 (2006).
12. Iten, R., Colbeck, R., Kukuljan, I., Home, J. & Christandl, M. Quantum circuits for isometries. *Phys. Rev. A* **93**, 032318 (2016).
13. Park, D. K., Petruccione, F. & Rhee, J.-K.K. Circuit-based quantum random access memory for classical data. *Sci. Rep.* **9**, 1–8 (2019).
14. Park, D. K., Sinayskiy, I., Fingerhuth, M., Petruccione, F. & Rhee, J.-K.K. Parallel quantum trajectories via forking for sampling without redundancy. *New J. Phys.* **21**, 083024. https://doi.org/10.1088/1367-2630/ab35fb (2019).
15. Cormen, T. H., Leiserson, C. E., Rivest, R. L. & Stein, C. *Introduction to algorithms* (MIT press, 2009).
16. Hoare, C. A. R. Algorithm 64: quicksort. *Commun. ACM* **4**, 321 (1961).
17. Gentleman, W. M. & Sande, G. Fast fourier transforms: for fun and profit. In *Proceedings of the November 7-10, 1966, fall joint computer conference*, 563–578 (1966).
18. Mottonen, M., Vartiainen, J. J., Bergholm, V. & Salomaa, M. M. Transformation of quantum states using uniformly controlled rotations. *Quant. Inf. Comput.* **5**, 467–473 (2005).
19. Aleksandrowicz, G. A. *et al.* Qiskit: An Open-source Framework for Quantum Computing, https://doi.org/10.5281/zenodo.2562111 (2019).
20. Grant, E. *et al.* Hierarchical quantum classifiers. *npj Quant. Inf.***4**, 65. https://doi.org/10.1038/s41534-018-0116-9 (2018).
21. Cohen, N. & Shashua, A. Convolutional rectifier networks as generalized tensor decompositions. *Int. Conf. Mach. Learn.* **955–963**, (2016).
22. Levine, Y., Sharir, O., Cohen, N. & Shashua, A. Quantum entanglement in deep learning architectures. *Phys. Rev. Lett.* **122**, 065301. https://doi.org/10.1103/PhysRevLett.122.065301 (2019).
23. Dua, D. & Graff, C. UCI machine learning repository (2017).
24. Rossi, R. A. & Ahmed, N. K. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, 4292–4293 (AAAI Press, 2015).
25. Demšar, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**, 1–30 (2006).
26. Bergholm, V. *et al.* Pennylane: Automatic differentiation of hybrid quantum-classical computations (2020). 1811.04968.
27. Kingma, D. P. & Ba, J, (A method for stochastic optimization, Adam, 2017) ((**1412.6980**)).

Acknowledgements

This research is supported by CNPq (Grant No. 308730/2018-6), CAPES- Finance Code 001 and FACEPE (Grant No. IBPG 0834-1.03/19), the National Research Foundation of Korea (Grant Nos. 2019R1I1A1A01050161 and 2018K1A3A1A09078001), the Ministry of Science and ICT, Korea, under an ITRC Program, IITP-2019-2018-0-01402, and the South African Research Chair Initiative of the Department of Science and Innovation and National Research Foundation (UID: 64812). We acknowledge use of IBM Q for this work. The views expressed are those of the authors and do not reflect the official policy or position of IBM or the IBM Q team.

Author contributions

A.J.S. devised the divide-and-conquer state preparation strategy, performed the proof-of-concept experiments and wrote a first version of the manuscript. D.K.P. suggested the way to make ancillary states orthogonal, and conceived the combination of swap test with the divide-and-conquer state preparation to calculate metrics between two datasets. I.F.A. performed the experiments with the variational quantum circuits and extended the algorithm to complex input vectors. All authors reviewed and discussed the analyses and results, and contributed towards writing the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to F.P.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

APÊNDICE B – CONFIGURABLE SUBLINEAR CIRCUITS FOR QUANTUM STATE PREPARATION

Springer Nature 2021 LaTeX template

Configurable sublinear circuits for quantum state preparation

Israel F. Araujo[1*], Daniel K. Park[2,3], Teresa B. Ludermir[1], Wilson R. Oliveira[4], Francesco Petruccione[5,6,7] and Adenilton J. da Silva[1]

[1*]Centro de Informática, Universidade Federal de Pernambuco, Recife, 50740-560, Pernambuco, Brazil.
[2]Department of Applied Statistics, Yonsei University, Seoul, 03722, Republic of Korea.
[3]Department of Statistics and Data Science, Yonsei University, Seoul, 03722, Republic of Korea.
[4]Departamento de Estatística e Informática, Universidade Federal Rural de Pernambuco, Recife, Pernambuco, Brazil.
[5]School of Data Science and Computational Thinking, Stellenbosch University, Stellenbosch, 7600, South Africa.
[6]National Institute for Theoretical and Computational Sciences (NITheCS), 4001, South Africa.
[7]Quantum Research Group, University of KwaZulu-Natal, Durban, 4001, South Africa.

*Corresponding author(s). E-mail(s): ifa@cin.ufpe.br;

Abstract

The theory of quantum algorithms promises unprecedented benefits of harnessing the laws of quantum mechanics for solving certain computational problems. A prerequisite for applying quantum algorithms to a wide range of real-world problems is loading classical data to a quantum state. Several circuit-based methods have been proposed for encoding classical data as probability amplitudes of a quantum state. However, in these methods, either quantum circuit depth or width must grow linearly with the data size, nullifying the advantage of representing exponentially-many classical data in a quantum

state. In this paper, we present a configurable bidirectional procedure that addresses this problem by tailoring the resource trade-off between quantum circuit width and depth. In particular, we show a configuration that encodes an N-dimensional classical data using a quantum circuit whose width and depth both grow sublinearly with N. We demonstrate proof-of-principle implementations on five quantum computers accessed through the IBM and IonQ quantum cloud services.

Keywords: quantum computing, state preparation, bidirectional, circuit optimization

1 Introduction

In 1981, Richard Feynman suggested a computational architecture based on cellular automata with quantum features and conjectured that it could accelerate simulations of quantum physics and chemistry beyond the capability of classical computers [1]. Since then, algorithms such as Deutsch-Jozsa [2], Grover [3], Simon [4], and Shor [5] demonstrated quantum advantages. However, some challenges prevent such advantages from applying to all quantum algorithms [6–8]. Among these challenges is the development of an efficient encoding of classical data into a quantum state [6–9].

Quantum algorithms assume an initial quantum state prepared before the computation. The worst case complexity of preparing an arbitrary quantum state is exponential with the number of qubits [10]. For this reason, the most significant quantum speed-ups occur when the quantum algorithm [2–5, 11, 12] operates on an input state that is easy to prepare, such as the uniform superposition of all computational basis states. For algorithms that rely on loading data into an arbitrary quantum state, an efficient means to prepare input states is a prerequisite to quantum speed-ups [6, 7, 9, 13].

While the quantum state preparation models based on quantum oracles [14–16] or quantum random access memory [17–24] are useful for evaluating the lower bounds of the computational cost and identifying the complexity class, implementations of them must be considered in practice. In particular, the quantum speed-up can vanish without an efficient implementation of quantum state preparation when quantum algorithms carry classical data in a non-uniform quantum superposition. Examples of such instances include Quantum Machine Learning (QML) [6, 7, 25–31], Quantum Memories (QMem) [17–24], and Quantum Linear Algebra (QLA) [7, 13, 32–35]. Quantum machine learning algorithms try to estimate a target function from a finite set of example points by unveiling correlations between inputs and outputs of the correspondent function [7, 28, 36]. Quantum memories must store a set of samples from a configuration space as a superposition state before the information is retrieved using the algorithm [17]. Quantum linear algebra

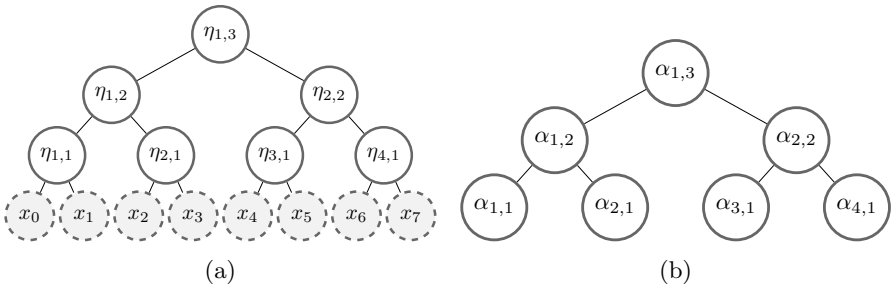


Fig. 1: Tree representations of quantum state preparation algorithms. (a) State decomposition tree generated by Algorithm 1 with an 8-dimensional input vector $\mathbf{x}$ (dashed nodes). The complex argument terms $\Omega_{i,k}$ were omitted for readability. (b) Angle tree generated by Algorithm 2 with an 8-dimensional input vector. The correspondent phase angles $\lambda_{j,v}$ were omitted for readability.

algorithms operate with a critical assumption that classical data has been efficiently encoded as probability amplitudes of a quantum state without which the quantum speed-up vanishes [6, 7, 9, 13]. All of the above emphasizes the importance of developing efficient quantum state preparation algorithms for broad application of quantum computing techniques on classical data.

Several solutions to the problem of quantum state preparation have been proposed [10, 14, 37–42], but all produce circuits with width or depth growing at least linearly with the size of the input vector [10]. For example, the top-down method proposed in Ref. [37] achieves the exponential compression of the quantum circuit width while requiring $O(N)$ quantum circuit depth for N-dimensional data. On the other extreme end, the bottom-up method [42] achieves the exponential compression of the quantum circuit depth while requiring $O(N)$ quantum circuit width and entangled information in ancillary qubits. Since there is an extra resource overhead in many quantum algorithms due to the quantum measurement postulate [23, 43], such linear cost can impose restrictions on possible speed-ups, dominating the computational cost of the intended quantum application. Other approaches have reduced circuit complexity to initialize an approximate quantum state [14, 44–46], but this paper targets the exact state preparation with entangled ancillary qubits.

This work presents a quantum state preparation method that achieves sublinear scaling on both quantum circuit resources. More specifically, a bidirectional strategy that effectively combines the aforementioned approaches in a way that the trade-off between computational time and space can be configured. Both temporal and spatial complexities depend on the parameter $s \in [1..n]$, which adjusts the trade-off between computational time and space. Given an N-dimensional input vector, the total time complexity of the bidirectional algorithm is $O_c(N) + O_d(2^s + \log_2^2(N) - s^2)$, where $O_c(N)$ is the time of the classical preprocessing to create the quantum circuit and $O_d(2^s + \log_2^2(N) - s^2)$ is the quantum circuit depth. Typically the same input

vector is loaded $l \gg N$ times, and hence the amortized computational time is $O_d(2^s + \log_2^2(N) - s^2)$. Note that classical preprocessing is also common in classical computing and is necessary in other quantum state preparation methods as well. The spatial complexity (i.e. the width) of the circuit is $O_w((s+1)N/2^s)$.

Besides the sublinear circuit cost, the ability to customize the exchange between these quantum resources is advantageous when realistic quantum hardware specifications are considered as one resource can be cheaper than the other to scale up. For instance, it is a useful feature for future Noisy Intermediate-Scale Quantum (NISQ) devices with the promise of computers with a large number of physical qubits [47], albeit noise limits the depth of the circuits [48].

This paper is divided into four sections. Section 2 reviews two strategies for loading classical information into quantum devices, namely top-down [39] and bottom-up [42] approaches. The former is used by quantum computing libraries [49, 50] as the method for general quantum amplitude initialization. These methods are at the two opposite ends of the quantum circuit cost spectrum requiring either the maximal circuit depth or width to minimize the other resource. Section 3 presents the main result, a bidirectional method that combines the top-down and bottom-up strategies in a configurable manner. Complexity expressions for the bidirectional method are established in Section 3.1, which shows that the bottom-up and the top-down strategies are recovered when $s = 1$ and $s = n$, respectively, and that sublinear scaling on both depth and width is possible when $s = n/2$. Proof-of-principle experiments performed on superconducting and ion-trap based quantum devices are presented in Section 3.2. Section 5 presents the conclusion and perspectives for future work.

2 Quantum state preparation with linear cost

2.1 Tree representation

Quantum state preparation algorithms aim to create a state $\sum_p |x_p| e^{i\omega_p} |p\rangle$ that encodes a normalized vector $\mathbf{x} = (|x_0| e^{i\omega_0}, \ldots, |x_{N-1}| e^{i\omega_{N-1}})$ as the probability amplitudes. Several of the existing methods can be understood as a walk on a binary tree [10, 39, 41, 42, 51]. Each tree node corresponds to a controlled gate operation and the height increases with the number of qubits (see Fig. 1a and Fig. 1b). Two edges stemming from each node indicate that each controlled gate operation splits the Hilbert space into two subspaces. Therefore, after n layers, there can be 2^n subspaces with distinct probability amplitudes. Depending on the choice of the walk direction, different state preparation strategies, such as top-down and bottom-up approaches, can be constructed.

To explain the procedure, four parameters [39] defined by the target vector $\mathbf{x}$ are introduced as

$$\Omega_{i,k} = \sum_{l=0}^{2^k-1} \omega_{(i-1)2^k+l}/2^{k-1} \tag{1}$$

$$\eta_{i,k} = \sqrt{\sum_{l=0}^{2^k-1} |x_{(i-1)2^k+l}|^2} \tag{2}$$

$$\lambda_{j,v} = \Omega_{2j,v-1} - \Omega_{j,v} \tag{3}$$

$$\beta_{j,v} = \eta_{2j,v-1}/\eta_{j,v} \tag{4}$$

where $j = 1, 2, \ldots, 2^{n-v}$, $v = 1, 2, \ldots, n$, and $n = \log_2(N)$. These parameters are used to construct the tree representations of the state preparation algorithms, namely the state tree (Fig. 1a) and the angle tree (Fig. 1b). Indices k and v indicate a tree level in ascending order from the leaf nodes to the root, i and j are node indices at a given level. The nodes of these trees are complex values that represent the amplitudes of the quantum state to be encoded and the rotation angles for the construction of the encoding quantum circuit. The magnitude and complex argument of the state tree amplitudes are obtained through $\eta_{i,k}$ and $\Omega_{i,k}$, respectively. When $k = 0$, the parameters point to the input vector $\mathbf{x}$. Equations (3) and (4) determine rotation values of the angle tree nodes. The phase arguments of the vector $|x\rangle$ are encoded through z-rotations of angles $\lambda_{j,v}$, and the magnitudes through y-rotations of angles $\alpha_{j,v} = 2\,\mathrm{asin}(\beta_{j,v})$.

Algorithms 1 and 2 describe the construction of a state tree and an angle tree. Respective pseudocodes 1 and 2 are presented in the appendix.

Algorithm 1: State tree construction

1. Initialize the state tree by the leafs, where each node value is a complex amplitude from a 2^n length state vector
2. Set $k = 1$
3. Create a new level with 2^{n-k} nodes, where each node i value is $\eta_{i,k} e^{i\Omega_{i,k}}$ (Eq. (1) and Eq. (2), $i = 1, \ldots, 2^{n-k}$)
4. If $k < n$, set $k = k + 1$ and return to Step 2, otherwise output the state tree

2.2 Top-down approach

The top-down amplitude encoding approach to quantum state initialization is a linear transformation consisting of a sequence of uniformly controlled rotations [39, 51] that takes the initial basis vector $|0\rangle^{\otimes n}$ to some arbitrary vector

Algorithm 2: Angle tree construction

1 Set $v = n$
2 Create a new level with 2^{n-v} nodes, where each node j value is $\alpha_{j,v}e^{i\lambda_{j,v}}$ (Eq. (4) and Eq. (3), $j = 1, \ldots, 2^{n-v}$) using data from a state tree generated by Alg. 1
3 If $v > 1$, set $v = v - 1$ and return to Step 1, otherwise output the angle tree

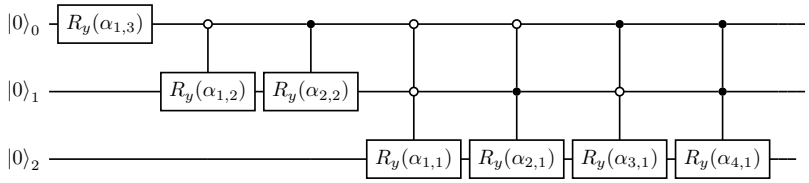


Fig. 2: Quantum circuit to load an 8-dimensional real vector in a quantum device using the top-down amplitude encoding strategy [10, 39, 51] (Alg. 3). The qubit index indicated by the subscript corresponds to the tree level in Fig. 1b.

$|x\rangle = (|x_0|e^{i\omega_0}, \ldots, |x_{N-1}|e^{i\omega_{N-1}})^T$. This generates a quantum circuit with complexity of $O_d(N)$ and $O_w(\log_2(N))$ [10, 39, 51].

The top-down state preparation (TDSP) algorithm begins by preparing the following state at the root ($v = n$) of the angle tree (see Fig. 1b for an example)

$$|\psi_n\rangle = e^{-i\frac{\lambda_{1,n}}{2}}\sqrt{1 - |\beta_{1,n}|^2}\,|0\rangle + e^{i\frac{\lambda_{1,n}}{2}}\beta_{1,n}\,|1\rangle\,. \tag{5}$$

To load states into the next level (indicated by v in Eq. (6)), the current state (indicated by $v + 1$ because v is in reverse order, decreasing from n to 1) is sequentially combined with the values of the next state in Eq. (6).

$$|\psi_v\rangle = \sum_{j=1}^{2^{n-v}} |j-1\rangle\langle j-1|\psi_{v+1}\rangle \left(e^{-i\frac{\lambda_{j,v}}{2}}\sqrt{1 - |\beta_{j,v}|^2}\,|0\rangle + e^{i\frac{\lambda_{j,v}}{2}}\beta_{j,v}\,|1\rangle\right) \tag{6}$$

The update of state $|\psi_v\rangle$ is repeated for $v = (n-1), \ldots, 1$, thereby obtaining the desired state

$$|\psi_1\rangle = |x_0|e^{i\omega_0}\,|0\rangle + \ldots + |x_{N-1}|e^{i\omega_{N-1}}\,|N-1\rangle\,.$$

The summation in Equation (6) expresses the sequential characteristic of the top-down approach, since the state of each layer of the tree needs to be loaded on one qubit through a sequence of rotations. Figure 2 presents an example

quantum circuit for encoding 8-dimensional vector using the top-down state preparation method.

Algorithm 3: Top-down state preparation

1 Generate a state tree from the input vector
2 Generate an angle tree from the state tree
3 Create a quantum circuit with n qubits (one qubit for each angle tree level)
4 Perform one y-rotation and one z-rotation on the first qubit (qubits are 0-indexed) using the angle tree root values $\alpha_{1,n}$ and $\lambda_{1,v}$ (Eq. (5))
5 Set $v = n - 1$ (starts at the root)
6 Perform a *Uniformly Controlled Rotation* controlled by qubits $0, 1, \ldots, n - v - 1$ (corresponding to the previous levels) with the current qubit $n - v$ as target, using the current level nodes values $\alpha_{j,v}$ and $\lambda_{j,v}$ $(1 \leq j \leq 2^{n-v})$ as rotation angles (Eq. (6))
7 If $v > 1$, set $v = v - 1$ and return to Step 5, otherwise output the encoding quantum circuit

The name *top-down* comes from the way this approach walks through the tree from the root to the leaves to build a quantum circuit. The combination of states is done with multi-controlled rotations, and it takes $\log_2(N)$ qubits to generate the complete state. At each level, it assembles a sequence of rotations targeting one qubit and is controlled by the qubits of the previous levels. First, y-rotations are applied to set the magnitudes, followed by z-rotations to set the phases. These steps are presented in Algorithm 3 with its Pseudocode 3 provided in the appendix.

2.3 Bottom-up approach

The bottom-up state preparation algorithm constructs a quantum circuit with complexity $O_d(\log_2^2(N))$ and $O_w(N)$ [42]. It starts by preparing $N/2$ single-qubit states, corresponding to the leaves of the tree (Fig.1a). Equations (3) and (4) are used starting from the lowest level of the tree ($v = 1$), which corresponds to starting from the initial state

$$|\psi_{j,1}\rangle = e^{-i\frac{\lambda_{j,1}}{2}}\sqrt{1-|\beta_{j,1}|^2}\,|0\rangle + e^{i\frac{\lambda_{j,1}}{2}}\beta_{j,1}\,|1\rangle\,. \tag{7}$$

Loading the states in the upper levels of the tree is done by recursive updates of

$$\begin{aligned}|\psi_{j,v}\rangle = \quad & e^{-i\frac{\lambda_{j,v}}{2}}\sqrt{1-|\beta_{j,v}|^2}\,|0\rangle\,|\psi_{2j-1,v-1}\rangle\,|\psi_{2j,v-1}\rangle + \\ & e^{i\frac{\lambda_{j,v}}{2}}\beta_{j,v}\,|1\rangle\,|\psi_{2j,v-1}\rangle\,|\psi_{2j-1,v-1}\rangle\,,\end{aligned} \tag{8}$$

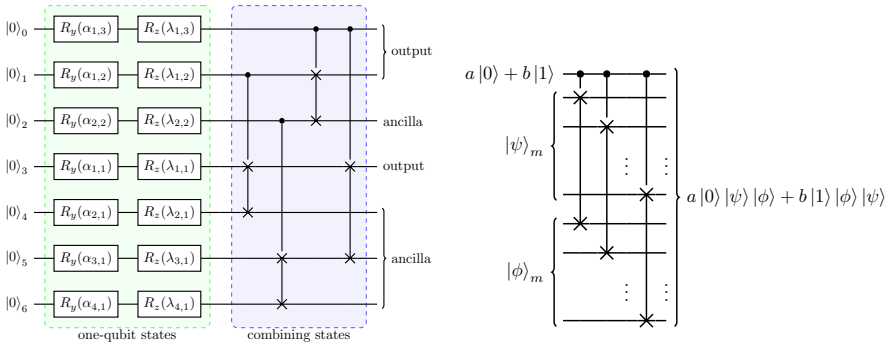


Fig. 3: Divide-and-conquer bottom-up load strategy. (a) Circuit generated by the divide-and-conquer [42] bottom-up strategy (Alg. 4) to load an 8-dimensional complex vector in a quantum device. The indexes of the qubits correspond to the tree nodes indexes in Fig. 1b. The circuit starts with the simultaneous preparation of $(N-1)$ one-qubit states associated with all tree nodes, followed by the combination of states through CSWAPs. (b) Combining states with controlled-swap operations.

where $v = 2, \ldots, n$. The desired state, with ancilla $|\phi\rangle$, is obtained when $v = n$ as

$$|\psi_{1,n}\rangle = |x_0|e^{i\omega_0}\,|0\rangle\,|\phi_0\rangle + \cdots + |x_{N-1}|e^{i\omega_{N-1}}\,|N-1\rangle\,|\phi_{N-1}\rangle\,. \tag{9}$$

Updating the states in Equation (8) requires a method that entangles each of the two states $|\psi_{2j-1,v-1}\rangle$ and $|\phi_{2j,v-1}\rangle$ to orthonormal subspaces $|0\rangle$ and $|1\rangle$, respectively, with designated amplitudes. As demonstrated by Araujo et al. [42], m controlled-swap (CSWAP) operations can combine two m-qubit states in the form of Equation (8) (see Fig. 3b) to encode the desired set of amplitudes in the orthonormal subspaces of the first $m+1$ qubits. Since each node of the level is represented by one qubit, multiple loading within a layer can be performed in parallel. Thus, all states in the given layer can be loaded simultaneously. This is an advantage in comparison to the top-down approach which loads each node state sequentially. Since the underlying idea of the bottom-up approach is recursive combination of single-qubit states that are easy to prepare, it was named as divide-and-conquer state preparation (DCSP) when first introduced [42]. An example quantum circuit for encoding 8-dimensional vector using the DCSP method is depicted in Fig. 3a. Algorithm 4 describes these steps and Pseudocode 4 is provided in the appendix.

Algorithm 4: Bottom-up state preparation

1 Generate a state tree from the input vector
2 Generate an angle tree from the state tree
3 Create a quantum circuit with $2^n - 1$ qubits (one qubit for each angle tree node)
4 Perform 2^{n-1} y-rotations and z-rotations on qubits $2^{n-1} + j - 2$ $(1 \leq j \leq 2^{n-1})$ using the leaf values $\alpha_{j,1}$ and $\lambda_{j,1}$ to prepare 2^{n-1} initial single-qubit states (Eq. (7), Fig. 3a)
5 Set $v = 2$ and $j = 1$ (starts at the bottom)
6 Perform one y-rotation and one z-rotation on qubit $2^{n-v} + j - 2$ using the node values $\alpha_{j,v}$ and $\lambda_{j,v}$ to prepare a single-qubit state to control *CSWAPs* operations
7 Perform *Controlled SWAPs* controlled by qubit $2^{n-v} + j - 2$ to combine the previous states prepared with the qubits associated to the sub-tree started by the current node (Eq. (8), Fig. 3b)
8 If $j < 2^{n-v}$, set $j = j + 1$ and return to Step 5, otherwise continue
9 If $v < n$, set $v = v + 1$ and return to Step 5, otherwise output the encoding quantum circuit

3 Bidirectional quantum state preparation

This section presents a bidirectional state preparation (BDSP) method combining both bottom-up and top-down strategies as walking on the tree in both directions. This new strategy can interchange depth and space cost in a configurable manner, thereby allowing for the sublinear cost in both quantum circuit depth and width. In particular, the equilibrium point between these costs achieves the quadratic reduction in both space and time. The algorithm is depicted in Fig. 4 and the detailed explanation is provided as follows.

The bidirectional state preparation algorithm starts by informing a level $v = s$ (enumerated from bottom to top, where $1 \leq s \leq n$) at which the angle tree is split, followed by two stages. In the first stage, it segments the tree section below s into 2^{n-s} sub-trees of height s. The 2^{n-s} nodes at level s are the roots of these sub-trees. The number of sub-trees determines how many initial sub-states should be prepared in the first stage of the algorithm. The amplitude values of these sub-states $a_j = (a_{j,1}, \ldots, a_{j,2^s})$ $(1 \leq j \leq 2^{n-s})$ are loaded concurrently using a sequential algorithm [10, 39, 51] based on the TDSP method as

$$|\psi_{j,s}\rangle = \sum_{k=1}^{2^s} a_{j,k} \, |k-1\rangle \quad ; \quad j = 1, 2, \ldots, 2^{n-s}. \tag{10}$$

The initial sub-states are the input of the second stage of BDSP. They reproduce the state that would be created by the bottom-up steps up to the split level s. In the second stage, the sub-states are combined to generate the

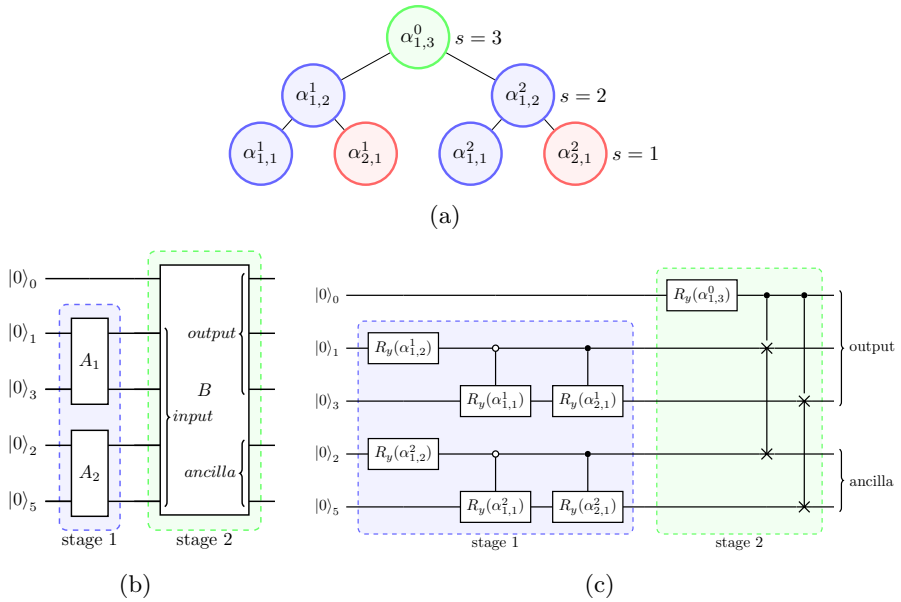


Fig. 4: Schematics of the bidirectional algorithm. (a) Angle tree example with a split at level $s = 2$. The blue and red nodes (α^1 and α^2) correspond to the bidirectional procedure first stage. In each of the two sub-trees of the first stage, 4 of the 8 amplitudes expected as input by stage 2 are encoded using a top-down method. The green node (α^0) above the tree split correspond to the second stage single sub-tree, subject to a partial DCSP bottom-up procedure. The first stage red nodes ($j > 1$) are no longer associated with an ancilla since they are now encoded through a top-down approach. (b) Block diagram circuit, corresponding to the tree in (a). In stage 1, the A_k operators (the index k is related to angle vectors α^k upper index) are responsible for encoding the amplitudes that will be used as input by stage 2. In this example, each A_k operator encodes 4 amplitudes from a total of 8. The B operator is the partial DCSP for 8 amplitudes, which is initialized with the expected state for the split level 2 and continues with the traditional algorithm. (c) Detailed view of (b), generated by the bidirectional strategy described in Algorithm 5 for a real and positive 8-dimensional input vector.

complete state by the divide-and-conquer approach (Fig. 4c). The bottom-up algorithm takes the state prepared in the first stage as the input, and starts walking on the tree from the split level (Eq. (8), where $v = s + 1, \ldots, n$). In other words, the BDSP follows the bottom-up DCSP algorithm starting from states $|\psi_{j,s}\rangle$ (Eq. (10)) instead of starting from the single-qubit leaf states (Eq. (7)). The BDSP algorithm is described in Algorithm 5 below with Pseudocode 5 provided in the appendix.

Algorithm 5: Bidirectional state preparation

1 Generate a state tree from the input vector
2 Generate an angle tree from the state tree
3 Create a quantum circuit with $(s+1)2^{n-s}-1$ qubits (Eq. (12))
4 Perform Algorithm 3 (top-down approach) starting from step 3 to prepare 2^{n-s} states of s-qubits (replacing n by s), using the 2^{n-s} sub-trees as input for each state (Eq. (10)). This step is named *Stage 1*
5 Perform Algorithm 4 (bottom-up approach) starting from step 5 and $v = s+1$ to combine the 2^{n-s} states prepared in *Stage 1* using the remaining $2^{n-s}-1$ qubits (Eq. (8)). This step is named *Stage 2*
6 Output the encoding quantum circuit

3.1 Complexity

In general, the BDSP algorithm builds quantum circuits whose depth and width are expressed respectively by

$$\underbrace{\frac{N}{2^{\log_2(N)-s}}}_{\text{stage 1}} + \underbrace{\sum_{i=s+1}^{\log_2(N)} i - 1}_{\text{stage 2}} = 2^s + \frac{1}{2}(\log_2^2(N) - \log_2(N) - s^2 + s) \tag{11}$$

and

$$\underbrace{s\frac{N}{2^s}}_{\text{stage 1}} + \underbrace{\frac{N}{2^s} - 1}_{\text{stage 2}} = (s+1)\frac{N}{2^s} - 1, \tag{12}$$

where N is the number of amplitudes (i.e. the dimension of the data vector) and s is a parameter indicating the tree splitting level (the tree level in reverse order). Stage 1 and 2 indicate the contribution from each stage of the bidirectional procedure to the circuit complexity stated in Theorem 1.

Theorem 1. *Algorithm 5 generates a quantum circuit with depth* $O_d\left(2^s + \log_2^2(N) - s^2\right)$ *and width* $O_w\left((s+1)\frac{N}{2^s}\right)$.

In Equation (11), first term (stage 1) is the leading-order approximation of the quantum circuit depth from existing top-down based algorithms [10, 39] for sub-states with s qubits. The exact expression depends on which of the two algorithms is used. The summation of the second term (stage 2) is the divide-and-conquer circuit depth from split level $s+1$ to n. Similarly, the first term in Equation (12) is the number of qubits occupied by all first stage sub-states and the second term is the number of qubits used by the second stage.

There are three noteworthy configuration values for the parameter s (see Table 1). Setting $s = \log_2(\sqrt{N})$ achieves asymptotic sublinearity, and $s = 1$ or $s = \log_2(N)$ recovers bottom-up or top-down approaches.

	bottom-up $s = 1$	top-down $s = n$	sublinear $s = n/2$
O_d	n^2	2^n	$2^{n/2}$
O_w	2^n	n	$2^{n/2}$

Table 1: Bidirectional quantum circuit complexity for different configurations. These expressions were obtained from Eq. (11) and Eq. (12).

The condition for quadratic reduction in both depth and width is obtained through asymptotic analysis of the minimum distance between Eq. (11) and Eq. (12). The first (second) equation is a monotonically increasing (decreasing) function $\forall s \in \{x \in \mathbb{R} | 4 \leq x \leq \log_2(N)\}$ and there is a point s where the distance is zero when $N \to \infty$. Thus the minimum distance point is given by finding s that satisfies

$$\lim_{N\to\infty} O_w - O_d = 0. \tag{13}$$

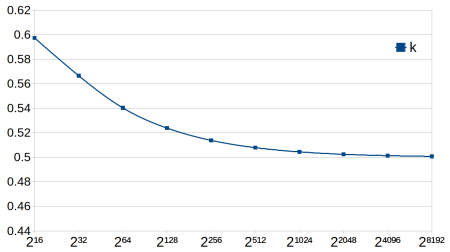


Fig. 5: The solution of the system $O_w - O_d = 0$ approaches $k = 1/2$ as N increases.

The asymptotic analysis starts by rewriting Eq. (11) and Eq. (12) using a more convenient parameterization,

$$s = f(k) = k \log_2(N) \quad \text{where} \quad k \in \left[\frac{4}{\log_2(N)}, 1\right].$$

Applying the limit of Eq. (13) results in the following simplified expression

$$N^{2k-1} = 1.$$

Solving the above equation for k gives the solution $k = 1/2$. Therefore, to achieve sublinear circuit complexity with quadratic reductions in both quantum circuit depth and width, the tree split must occur at $s = 1/2 \log_2(N) = n/2$, which leads to Theorem 2.

Theorem 2. *Algorithm 5 with $s = n/2$ and $N \gg 1$ generates a quantum circuit with sublinear depth $O_d(\sqrt{N})$ and width $O_w(\sqrt{N})$.*

When dealing with input vectors of small size, s can be calculated by solving Eq. (13) directly with N being a constant. If s cannot be calculated exactly, it can be approximated with the asymptotic result $s = \lceil n/2 \rceil$. The reason for the *ceiling* function is because s approximates $n/2$ from upper values (Fig. 5).

Corollary 2.1. *When $N \leq 8$ a top-down approach ($s = \log_2(N)$) should always be used, since space and depth both decrease as s increases in the interval $s \in [1..3]$ (see Table 3 for a numerical example). Circuit depth increases only when $s \geq 4$.*

3.2 Experiment

To evaluate the bidirectional method, proof-of-principle experiments were performed on a classical simulator provided by IBM, four superconducting-qubit based quantum devices provided by IBM, and an ion-trap based quantum device provided by IonQ. These are named as ibmq_qasm_simulator, ibmq_rome, ibmq_santiago, ibmq_casablanca, and ibmq_jakarta, and IonQ, respectively. The experiments aim to load the following 8 and 16-dimensional real input vectors:

$$(\sqrt{0.03}, \sqrt{0.06}, \sqrt{0.15}, \sqrt{0.05}, \sqrt{0.1}, \sqrt{0.3}, \sqrt{0.2}, \sqrt{0.11})$$

and

$$(\sqrt{0.01}, \sqrt{0.02}, \sqrt{0.04}, \sqrt{0.02}, \sqrt{0.07}, \sqrt{0.08}, \sqrt{0.04}, \sqrt{0.01},$$
$$\sqrt{0.08}, \sqrt{0.02}, \sqrt{0.21}, \sqrt{0.09}, \sqrt{0.12}, \sqrt{0.08}, \sqrt{0.05}, \sqrt{0.06}).$$

Three configurations of the bidirectional method are compared, namely top-down ($s = n$), bottom-up ($s = 1$), and sublinear ($s = \lceil n/2 \rceil$). The first case uses the least number of qubits $O_w(\log_2(N))$ and maximum depth $O_d(N)$. In the second configuration, depth is minimum $O_d(\log_2^2(N))$ and the number of qubits is maximum $O_w(N)$. The last configuration uses the best trade-off between the quantum circuit depth and width and achieves the sublinear scaling for both. In this case, the quantum circuit depth and width both grow as $O(\sqrt{N})$.

Table 2 lists the experimental results, presenting the number of runs per device and dimensionality of the input vector. The ibmq_rome and ibmq_santiago devices have only five qubits, and due to this limitation they are not suitable to encode the 8-dimensional vector with the bottom-up configuration (i.e. $s = 1$) or to perform sublinear (i.e. $s = \lceil n/2 \rceil$) and bottom-up experiments to encode the 16-dimensional vector (see Corollary 2.1). None of the quantum devices used in this work has the capacity to run the bottom-up configuration for the 16-dimensional input vector, which requires at least 15 qubits (i.e. $N - 1$ qubits).

Figure 7 presents the average output of the experiments with 8 and 16-dimensional input vectors. The height of blue and red bars is an average value obtained from a number of repetitions shown in the *runs* column in Table 2, and the error bars represent the standard deviation. The height of the yellow bar is the experimental result averaged over all quantum devices.

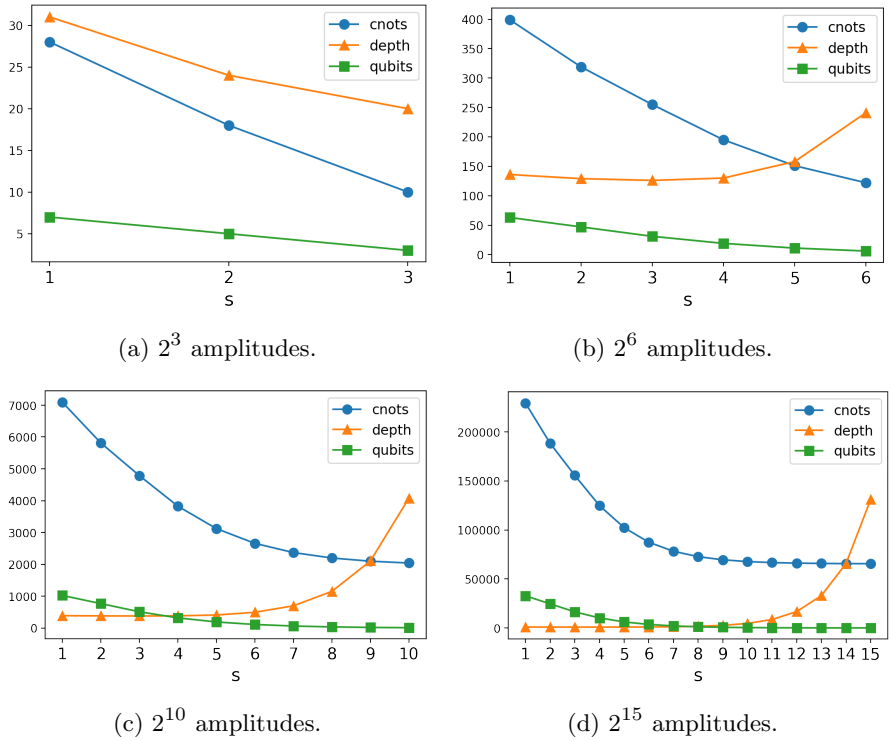
device	N	s	runs	MAE
ibmq_qasm_simulator 32 qubits	8	1	5	0.0016
		2	5	0.0005
		3	5	0.0015
	16	2	5	0.0010
		4	5	0.0010
ibmq_rome 5 qubits	8	2	10	0.0577
		3	10	**0.0429**
	16	4	5	0.0409
ibmq_santiago 5 qubits	8	2	10	0.0464
		3	10	**0.0233**
	16	4	10	0.0225
ibmq_casablanca 7 qubits	8	1	10	0.0710
		2	10	0.0691
		3	10	**0.0213**
ibmq_jakarta 7 qubits	8	1	10	0.0594
		2	10	0.0497
		3	10	**0.0289**
IonQ 11 qubits	8	1	5	0.0455
		2	5	0.0242
		3	5	**0.0217**
	16	2	5	0.0261
		4	5	**0.0107**

Table 2: Results of the BDSP experiments that encode N-dimensional input vectors in the amplitudes of quantum states using a classical simulator and quantum devices for $N = \{8, 16\}$. The acronym MAE stands for mean absolute error. The bold font indicates the smallest MAE, and hence the best performance, among different configurations of s for each device and input vector.

	$N = 8$			$N = 16$			$N = 32$			$N = 64$		
s	CNOTs	depth	qubits	CNOTs	depth	qubits	CNOTs	depth	qubits	CNOTs	depth	qubits
1	28	31	7	77	58	15	182	93	31	399	136	63
2	18	24	5	57	51	11	142	86	23	319	129	47
3	10	20	3	41	48	7	110	83	15	255	126	31
4				26	51	4	80	87	9	195	130	19
5							58	114	5	151	158	11
6										122	241	6

Table 3: Exchange between circuit depth, width (qubits), and number of CNOTs by adjusting the parameter s (split). s can be interpreted as a hyperparameter to fine-tune the encoding circuit to hardware characteristics such as relaxation time, dephasing time, and the CNOT gate error.

Table 3 and Figure 6 show the trade-off between quantum circuit depth, width and the number of CNOT gates as s is varied for randomly generated target vectors of various sizes. As expected through the analysis of the number of CNOT gates and the circuit depth in Tab. 3, the experimental results in Table 2 and Figure 7 show performance favoring the top-down configuration ($s = n$) for small input sizes ($N < 64$) due to the smaller number of CNOT gates and the smaller or approximately equal depth of the circuit. The number of CNOT gates, circuit depth, and number of qubits all decrease as s progresses to $s = 3$. The depth starts to increase when $s > 3$, as previously implied by



(a) 2^3 amplitudes.

(b) 2^6 amplitudes.

(c) 2^{10} amplitudes.

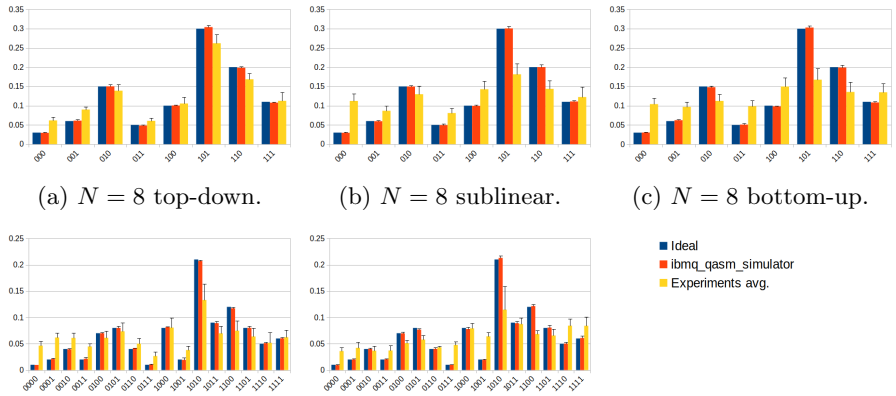
(d) 2^{15} amplitudes.

Fig. 6: Exchange between circuit depth, width (number of qubits), and number of CNOTs to load a 2^n-dimensional complex vector into a quantum computer by adjusting parameter s. The increasing number of CNOTs at lower depths is a consequence of exchanging computational time for space, given the combination of distant states. It also indicates an increase in concurrent operations.

Eq. (11). The comparison employs the mean absolute error (MAE). For each device and input size, the ranking is established where a smaller MAE indicates better performance (see Table 2).

Data from Table 3 and Figure 6 were obtained using the *transpile* method in the Quantum Information Science Kit (Qiskit [49]) version 0.26.2 to decompose the circuits into physical single-qubit gates and the CNOT gate. These circuits were generated by the bidirectional algorithm with random complex input vectors. The Python code used in this work for implementing Algorithm 3 employs functions *ucry* and *ucrz* from Qiskit. These functions are called uniformly controlled rotations (or multiplexers), and the corresponding code in Qiskit is based on the work of Shende et al. [10].

Note that algorithms 3 and 4 allocate logical qubits as they are needed without concerning their assignment to physical qubits of the quantum device. For NISQ devices with limited quantum device coupling map, the logical to



(a) $N = 8$ top-down. (b) $N = 8$ sublinear. (c) $N = 8$ bottom-up.

(d) $N = 16$ top-down. (e) $N = 16$ sublinear.

Fig. 7: Experimental results with 8- and 16-dimensional input vectors. Blue and red bars indicate respectively the ideal results and the ibm_qasm_simulator results. Yellow bars indicate the output average values from the experiments on all quantum devices. Error bars are the standard deviation.

physical qubit mapping should be optimized in order to minimize the overhead in the quantum circuit depth and the number of gates.

4 Sparse bidirectional quantum state preparation

The previous section explained the bidirectional method for encoding data in the amplitudes of a quantum state consisting of n qubits, where $2^n = N$ is the total amount of amplitudes of that state. Like other approaches for loading data in the amplitudes, the method presented in this paper has a classical preprocessing complexity $O(N)$. Moreover, such algorithms generate circuits whose spatial and depth costs also depend on N. This dependence on the total number of state amplitudes makes these methods inappropriate for encoding sparse input vectors, where the number of non-zero amplitudes M is much smaller than 2^n.

This section presents a sparse variant of the bidirectional algorithm (Pseudocode 6). This variant reduces the classical preprocessing complexity to $O(M)$ through a modification in the construction of the state tree (Pseudocode 7, Fig. 8b). Knowing that such a tree is binary, one can see that the value of each node is a combination of the pair at the lower level (see Eq. 4). If the zero-valued amplitudes are not present at the leafs of the state tree (Fig. 8a), the nodes at the levels above, whose children are absent, are not constructed (Fig. 8b). This eliminates some branches from the tree.

As seen in the previous section (Eq. 10), the size of a top-down sub-state is defined by the parameter s (split), which indicates the number of qubits of

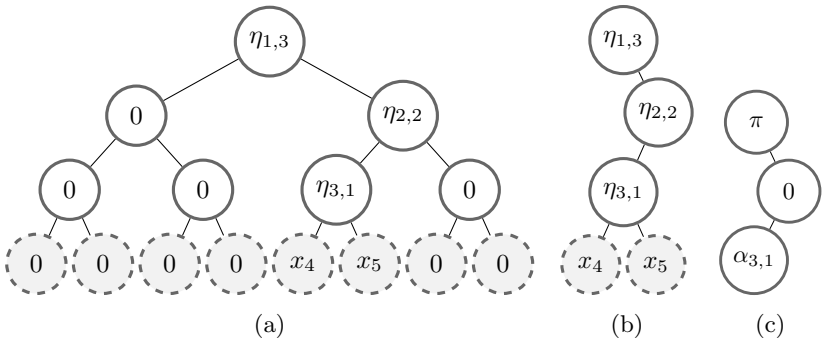


Fig. 8: Sparse state preparation. (a) Dense state decomposition of an 8-dimensional input vector with only data patterns 4 and 5 not equal to zero. Compare with Figure 1a. (b) Sparse state decomposition generated by Pseudocode 7. The information in this sparse decomposition is equal to that of Figure 8a. (c) Sparse angle tree generated by function angle_tree (Pseudocode 2) from the sparse state tree shown in Figure 8b. Node $\alpha_{1,3}$ is always equal to π because $\eta_{1,3}$ only has a child on the right (Pseudocode 7, Line 16, and Pseudocode 2, Line 6). Node $\alpha_{2,2}$ is always zero because $\eta_{2,2}$ only has a child on the left (Pseudocode 7, Line 25, and Pseudocode 2, Line 2). Node $\eta_{3,1}$ has left and right children, so $\alpha_{3,1}$ value depends on nodes x_4 and x_5 (Pseudocode 7, Line 20).

the sub-state and, therefore, the number of amplitudes 2^s that it can encode. The number of sub-states depends on two parameters, s (split) and n (total number of qubits of the complete state, which also indicates the height of the state tree). With both parameters, the number of sub-states is $2^n/2^s = 2^{n-s}$. Therefore, each block of 2^s amplitudes from the input vector makes up a sub-state and that for this sparse version if the block is empty (zero amplitudes) the sub-state is not necessary for the construction of the complete state (Fig. 8b). If a sub-state has all nodes eliminated, it is no longer needed nor represented in the quantum circuit, reducing the necessary space (qubits). Another consequence of eliminating a sub-state is that the second stage of the algorithm (bottom-up) will combine a smaller number of states (Eq. 8), therefore reducing the number of CSWAP operators and the total depth of the circuit.

The main difference between the dense algorithm (Pseudocode 5) and the sparse one (Pseudocode 6) is the construction of the state tree. Function `initialize_bidirectional` differs from function `initialize_sparse_bidirectional` only by Line 3, replacing the `state_decomposition` function call with `sparse_state_decomposition`. In the latter function, the main change that enables sparsity is the introduction of Line 16 *if* conditional statement. This statement has three possible outcomes. The first one (Line 20) occurs when two consecutive nodes of even and

odd index are present, being identical to the dense case (Pseudocode. 1). The other two outcomes, first (Line 16) and third (Line 25) conditions, occur when either of the pair nodes is absent and assumed to have zero amplitude. If the complete pair is absent, none of the statement outcomes are met, then the pair is ignored, and the respective parent node is not created. Node indices are guaranteed to be preserved by Line 12 of Pseudocode 7.

5 Conclusion

Existing state preparation methods, such as top-down and bottom-up approaches, require at least one quantum circuit resource between depth and width to grow linearly with the problem size. The BDSP algorithm presented in this work provides a general framework for configuring the trade-off between these resources that can be useful to manage them on NISQ devices. Looking at the state preparation algorithms as a walk on the state tree (see Section 2.1), the BDSP algorithm constitutes a systematic way to walk in two opposite directions. Previous methods are based on walking only in one direction. The bidirectional algorithm comes with a free parameter $s \in [1, n]$ that determines the balance between the top-down and the bottom-up approaches. At two extreme cases of setting $s = n$ and $s = 1$, the top-down and the bottom-up approaches are respectively recovered. At the equilibrium point $s = \lceil n/2 \rceil$, quadratic reduction in both quantum circuit depth and width can be achieved. The configuration parameter can be viewed as a hyperparameter that can tune circuit sizes and the number of CNOT gates according to the compound of application and hardware properties. The BDSP method is validated and demonstrated through experiments performed on five real quantum devices. The experiments behaved as expected, according to the asymptotic and numerical analyses of the circuit complexity.

A possible future work is to investigate whether the quantum circuit cost of the DCSP part can be futher reduced. Note that the structure of CSWAP operations in the DCSP step only depends on the dimensionality of the dataset N. Hence, the CSWAP operations can be interpreted as a single layer of fixed operation. Decomposing this fixed operation more efficiently than the naive application of CSWAP gates would achieve further reduction in the quantum circuit depth.

Acknowledgments. This work is based upon research supported by CNPq (Grant No. 308730/2018-6, No. 306727/2017-0, No. 409415/2018-9 and No. 421849/2016-9), CAPES – Finance Code 001, FACEPE (Grant No. IBPG-0834-1.03/19), National Research Foundation of Korea (Grant No. 2019R1I1A1A01050161, No. 2021M3H3A1038085 and No. 2022M3E4A1074591), the South African Research Chair Initiative, Grant No. 64812, of the Department of Science and Innovation and the National Research Foundation (NRF). Support of the NICIS (National Integrated Cyberinfrastructure System) e-research grant QICSA is kindly acknowledged. We acknowledge the use of IBM Quantum services for this work. The views

expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team.

Competing interests

The authors declare no competing interests.

Data availability

The sites https://github.com/qclib/qclib-papers and https://github.com/qclib/qclib contain all the data and the software generated during the current study.

Appendix A Pseudocode

Pseudocodes 1 to 5 expresses algorithms 1 to 5. Pseudocodes 1 and 2 construct the tree representations of the state preparation algorithms, namely the state tree and the angle tree, as described in Section 2.1. Pseudocodes 3 and 4, which algorithms are explained in sections 2.2 and 2.3, build quantum circuits using top-down and bottom-up approaches for encoding a complex input vector into the amplitudes of a quantum state. Pseudocode 5 employs pseudocodes 1 to 4 and expresses the bidirectional state preparation algorithm (Sec. 3, Alg. 5).

Lines 5 and 6 of Pseudocode 5 indicate the two stages of the BDSP algorithm. Line 5 at function `top_down_tree_walk` performs the first stage preparing the sub-states expected by the next stage, equivalent to what would be generated by bottom-up DCSP up to the tree split, but with the absence of ancilla due to the top-down approach. Line 6 at function `bottom_up_tree_walk` performs the second stage, starting at level $s+1$ with the sub-states initialized by the previous stage. Line 3 at function `top_down_tree_walk` configures the recurrence so that at split level s it divides the angle tree into 2^{n-s} (number of nodes at split level s) sub-trees of height s, loading all these sub-trees concurrently using the top-down strategy. Lines 11 and 12 of function `bottom_up_tree_walk` initialize $2^{n-s}-1$ qubits exclusive to the second stage with values $R_y(\alpha_{j,v})$ and $R_z(\lambda_{j,v})$. Then function `cswaps` combine the states through CSWAP gates controlled by the nodes above level s. With the tree described in Fig. 4a and $s=2$, the bidirectional procedure (Pseudocode 5) generates the circuit present in Fig. 4c.

Pseudocode 1: Generate a state tree by the decomposition of an amplitude input vector

```
1  state_decomposition(nqubits, data):
       input  : Number of qubits (nqubits) required to generate a state
                with the same length as the data vector (2^nqubits).
       input  : A list (data) representing the state to be decomposed,
                with exactly 2^nqubits pairs (index, amplitude).
       output: Root of the state tree.
       // Initialize an auxiliary vector new_nodes with data
          vector amplitudes
2      new_nodes = []
3      for k ← 0 to length(data) − 1 do
4          node.index = data[k].index
5          node.level = nqubits
6          node.amplitude = data[k].amplitude
7          new_nodes[k] = node
       // Build the state tree
8      for level ← nqubits to 1 step −1 do
9          nodes = new_nodes
10         new_nodes = []
11         for k ← 0 to length(nodes) − 1 step 2 do
12             mag = sqrt(|nodes[k].amplitude|^2 + |nodes[k+1].amplitude|^2)
13             arg = (∠nodes[k].amplitude + ∠nodes[k+1].amplitude)/2
14             node.index = nodes[k].index // 2
15             node.level = level
16             node.amplitude = mag × exp(1j × arg)
17             node.left = nodes[k]
18             node.right = nodes[k+1]
19             new_nodes[k//2] = node
20     return new_nodes[0] ;                    // return tree root
```

Pseudocode 2: Generate a angle tree that will be used to perform the state preparation

```
1  angle_tree(state_tree):
       input  : An output of state_decomposition function (state_tree).
       output: Tree with angles that will be used to perform the state
               preparation.
2      angle_y, angle_z = 0
3      if state_tree.right ≠ null then
4          amp = 0
5          if state_tree.amplitude ≠ 0 then
6              amp = state_tree.right.amplitude / state_tree.amplitude
7          angle_y = 2 arcsin(|amp|)
8          angle_z = 2∠amp
9      node.index = state_tree.index
10     node.level = state_tree.level
11     node.angle_y = angle_y
12     node.angle_z = angle_z
13     if state_tree.left ≠ null  & !is_leaf(state_tree.left) then
14         node.left = angle_tree(state_tree.left)
15     if state_tree.right ≠ null  & !is_leaf(state_tree.right) then
16         node.right = angle_tree(state_tree.right)
17     return node
```

Pseudocode 3: Construct a circuit that perform a top-down state preparation for the input vector *state*. The intended quantum state is encoded on the *output_qubits*.

```
1  top_down_tree_walk(angle_tree, circuit, start_level,
     control_nodes=null, target_nodes=null):
       input  : An output of angle_tree function (angle_tree).
       input  : A quantum circuit to apply the top-down encoding
                (circuit).
       input  : The tree level to start the walk (start_level).
       input  : Used in the recursive calls (control_nodes).
       input  : Used in the recursive calls (target_nodes).
       output: circuit after the application of the top-down encoding.
2      if angle_tree ≠ null then
3          if angle_tree.level < start_level then
4              top_down_tree_walk(angle_tree.left, circuit, start_level)
5              top_down_tree_walk(angle_tree.right, circuit, start_level)
6          else
7              angle_tree.qubit = add_qubit(circuit)
8              if target_nodes == null then
9                  control_nodes = [] ;      // initialize the controls
                    list
10                 target_nodes[0] = angle_tree ;        // start by the
                    sub-tree root
11             uniformly_controlled_rotation(circuit, control_nodes,
                target_nodes)
12             append(control_nodes, angle_tree) ; // add curr. node to
                the controls list
13             target_nodes = children(target_nodes) ;      // all the
                nodes in the next level
14             if angle_tree.left ≠ null then
15                 top_down_tree_walk(angle_tree.left, circuit, start_level,
                    control_nodes, target_nodes)
16             else
17                 top_down_tree_walk(angle_tree.left, circuit, start_level,
                    control_nodes, target_nodes)
18 initialize_top_down(circuit, state):
19     nqubits = log2(length(state))
20     state_tree = state_decomposition(nqubits, state)
21     angle_tree = angle_tree(state_tree)
22     top_down_tree_walk(angle_tree, circuit, 0)
23     output_nodes = left_view(angle_tree)
24     for k ← 0 to nqubits − 1 do
25         output_qubits[k] = output_nodes[k].qubit
26     return output_qubits
```

Pseudocode 4: Construct a circuit that perform a bottom-up state preparation for the input vector *state*. The intended quantum state is encoded on the *output_qubits*.

```
1  cswaps(angle_tree, circuit):
       input  : An output of angle_tree function (angle_tree).
       input  : A quantum circuit to apply the cswaps (circuit).
       output: circuit after the application of the cswaps.
2      left = angle_tree.left
3      right = angle_tree.right
4      while left ≠ null & right ≠ null do
5          circuit.cswap(angle_tree.qubit, left.qubit, right.qubit)
6          left = left.left
7          right = right.left
8  bottom_up_tree_walk(state_tree, circuit, start_level):
       input  : An output of state_decomposition function (state_tree).
       input  : A quantum circuit to apply the bottom-up encoding
                (circuit).
       input  : The tree level to start the bottom-up walk (start_level).
       output: circuit after the application of the bottom-up encoding.
9      if angle_tree ≠ null & angle_tree.level < start_level then
10         angle_tree.qubit = add_qubit(circuit)
11         circuit.ry(angle_tree.angle_y, angle_tree.qubit)
12         circuit.rz(angle_tree.angle_z, angle_tree.qubit)
13         bottom_up_tree_walk(angle_tree.left, circuit, start_level)
14         bottom_up_tree_walk(angle_tree.right, circuit, start_level)
15         cswaps(angle_tree, circuit)
16 initialize_bottom_up(circuit, state):
17     nqubits = log2(length(state))
18     state_tree = state_decomposition(nqubits, state)
19     angle_tree = angle_tree(state_tree)
20     bottom_up_tree_walk(angle_tree, circuit, nqubits)
21     output_nodes = left_view(angle_tree)
22     for k ← 0 to nqubits − 1 do
23         output_qubits[k] = output_nodes[k].qubit
24     return output_qubits
```

Pseudocode 5: Construct a circuit that perform a bidirectional state preparation for the input vector *state*. The intended quantum state is encoded on the *output_qubits*.

```
initialize_bidirectional(circuit, state, split):
    nqubits = log2(length(state))
    state_tree = state_decomposition(nqubits, state)
    angle_tree = angle_tree(state_tree)
    top_down_tree_walk(angle_tree, circuit, nqubits−split) ; // stage 1
    bottom_up_tree_walk(angle_tree, circuit, nqubits−split) ; // stage 2
    output_nodes = left_view(angle_tree)
    for k ← 0 to nqubits − 1 do
        output_qubits[k] = output_nodes[k].qubit
    return output_qubits
```

Pseudocode 6: Construct a circuit that perform a bidirectional state preparation for the sparse input vector *state*. The intended quantum state is encoded on the *output_qubits*.

```
initialize_sparse_bidirectional(circuit, state, split):
    nqubits = log2(length(state))
    state_tree = sparse_state_decomposition(nqubits, state)
    angle_tree = angle_tree(state_tree)
    top_down_tree_walk(angle_tree, circuit, nqubits−split) ; // stage 1
    bottom_up_tree_walk(angle_tree, circuit, nqubits−split) ; // stage 2
    output_nodes = left_view(angle_tree)
    for k ← 0 to nqubits − 1 do
        output_qubits[k] = output_nodes[k].qubit
    return output_qubits
```

Pseudocode 7: Generate a sparse state tree by the decomposition of an sparse amplitude input vector

```
1  sparse_state_decomposition(nqubits, data):
       input  : Number of qubits (nqubits) required to generate a state
                with the same length as the data vector (2^nqubits).
       input  : A list (data) representing the state to be decomposed,
                with exactly 2^nqubits pairs (index, amplitude).
       output: Root of the state tree.
       // Initialize an auxiliary vector new_nodes with data
          vector amplitudes
2      new_nodes = []
3      for k ← 0 to length(data) − 1 do
4          node.index = data[k].index
5          node.level = nqubits
6          node.amplitude = data[k].amplitude
7          new_nodes[k] = node
       // Build the state tree
8      for level ← nqubits to 1 step −1 do
9          nodes = new_nodes
10         new_nodes = []
11         for k ← 0 to length(nodes) − 1 step 2 do
12             node.index = nodes[k].index // 2
13             node.level = level
14             mag = |nodes[k].amplitude|
15             arg = (∠nodes[k].amplitude)/2
16             if nodes[k].index%2 == 1  then
17                 node.left = null
18                 node.right = nodes[k]
19                 k = k - 1
20             else if (k + 1) < length(nodes)  &  nodes[k + 1].index ==
                 nodes[k].index + 1  then
21                 mag =
                     sqrt(|nodes[k].amplitude|^2 + |nodes[k+1].amplitude|^2)
22                 arg = (∠nodes[k].amplitude+∠nodes[k+1].amplitude)/2
23                 node.left = nodes[k]
24                 node.right = nodes[k+1]
25             else
26                 node.left = nodes[k]
27                 node.right = null
28                 k = k - 1
29             node.amplitude = mag × exp(1j × arg)
30             new_nodes.append(node)
31     return new_nodes[0] ;                          // return tree root
```

References

[1] Feynman, R.P.: Simulating physics with computers. International Journal of Theoretical Physics **21**(6-7), 467–488 (1982)

[2] Deutsch, D., Jozsa, R.: Rapid solution of problems by quantum computation. Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences **439**(1907), 553–558 (1992)

[3] Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing. STOC '96, pp. 212–219. Association for Computing Machinery, Philadelphia, Pennsylvania, USA (1996)

[4] Simon, D.R.: On the Power of Quantum Computation. SIAM Journal on Computing **26**(5), 1474–1483 (1997)

[5] Shor, P.W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM Review **41**(2), 303–332 (1999)

[6] Aaronson, S.: Read the fine print. Nature Physics **11**(4), 291–293 (2015)

[7] Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., Lloyd, S.: Quantum machine learning. Nature **549**(7671), 195–202 (2017)

[8] Leymann, F., Barzen, J.: The bitter truth about gate-based quantum algorithms in the NISQ era. Quantum Science and Technology **5**(4), 044007 (2020)

[9] Tang, E.: Quantum principal component analysis only achieves an exponential speedup because of its state preparation assumptions. Phys. Rev. Lett. **127**, 060503 (2021)

[10] Shende, V.V., Bullock, S.S., Markov, I.L.: Synthesis of quantum-logic circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **25**(6), 1000–1010 (2006)

[11] Hogg, T., Huberman, B.A., Williams, C.P.: Phase transitions and the search problem. Artificial Intelligence **81**(1), 1–15 (1996). Frontiers in Problem Solving: Phase Transitions and Complexity

[12] Terhal, B.M., Smolin, J.A.: Single quantum querying of a database. Physical Review A **58**(3), 1822–1826 (1998)

[13] Harrow, A.W., Hassidim, A., Lloyd, S.: Quantum algorithm for linear systems of equations. Phys. Rev. Lett. **103**, 150502 (2009)

[14] Grover, L.K.: Synthesis of Quantum Superpositions by Quantum Computation. Physical Review Letters **85**(6), 1334–1337 (2000)

[15] Sanders, Y.R., Low, G.H., Scherer, A., Berry, D.W.: Black-box quantum state preparation without arithmetic. Phys. Rev. Lett. **122**, 020502 (2019)

[16] Wang, S., Wang, Z., Cui, G., Shi, S., Shang, R., Fan, L., Li, W., Wei, Z., Gu, Y.: Fast black-box quantum state preparation based on linear combination of unitaries. Quantum Information Processing **20**(8), 270 (2021)

[17] Trugenberger, C.A.: Probabilistic quantum memories. Physical Review Letters **87**(6) (2001)

[18] Ventura, D., Martinez, T.: Quantum associative memory. Information Sciences **124**(1), 273–296 (2000)

[19] Trugenberger, C.A.: Quantum Pattern Recognition. Quantum Information Processing **1**(6), 471–493 (2002)

[20] Giovannetti, V., Lloyd, S., Maccone, L.: Quantum random access memory. Phys. Rev. Lett. **100**, 160501 (2008)

[21] Silva, A., Oliveira, W.d., Ludermir, T.: A Weightless Neural Node Based on a Probabilistic Quantum Memory. In: 2010 Eleventh Brazilian Symposium on Neural Networks, pp. 259–264. IEEE, Sao Paulo, Brazil (2010). ISSN: 2375-0235

[22] de Paula Neto, F.M., da Silva, A.J., de Oliveira, W.R., Ludermir, T.B.: Quantum probabilistic associative memory architecture. Neurocomputing **351**, 101–110 (2019)

[23] Park, D.K., Petruccione, F., Rhee, J.-K.K.: Circuit-Based Quantum Random Access Memory for Classical Data. Scientific Reports **9**(1), 3949 (2019)

[24] Zidan, M., Abdel-Aty, A.-H., Khalil, A., Abdel-Aty, M., Eleuch, H.: A novel efficient quantum random access memory. IEEE Access **9**, 151775–151780 (2021)

[25] Lloyd, S., Mohseni, M., Rebentrost, P.: Quantum algorithms for supervised and unsupervised machine learning. arXiv:1307.0411 [quant-ph] (2013)

[26] Stoudenmire, E., Schwab, D.J.: Supervised learning with tensor networks. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (eds.)

Advances in Neural Information Processing Systems, vol. 29, p. 9. Curran Associates, Inc., Centre Convencions Internacional Barcelona, Barcelona Spain (2016)

[27] Schuld, M., Fingerhuth, M., Petruccione, F.: Implementing a distance-based classifier with a quantum interference circuit. EPL (Europhysics Letters) **119**(6), 60002 (2017)

[28] Schuld, M., Petruccione, F.: Supervised Learning with Quantum Computers, 1st ed. 2018 edn. Quantum Science and Technology. Springer, Cham (2018)

[29] Benedetti, M., Lloyd, E., Sack, S., Fiorentini, M.: Parameterized quantum circuits as machine learning models. Quantum Science and Technology (2019)

[30] Levine, Y., Sharir, O., Cohen, N., Shashua, A.: Quantum Entanglement in Deep Learning Architectures. Physical Review Letters **122**(6), 065301 (2019)

[31] Blank, C., Park, D.K., Rhee, J.-K.K., Petruccione, F.: Quantum classifier with tailored quantum kernel. npj Quantum Information **6**(1), 1–7 (2020)

[32] Lloyd, S., Mohseni, M., Rebentrost, P.: Quantum principal component analysis. Nature Physics **10**(9), 631–633 (2014)

[33] Childs, A.M., Kothari, R., Somma, R.D.: Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. SIAM Journal on Computing **46**(6), 1920–1950 (2017)

[34] Wossnig, L., Zhao, Z., Prakash, A.: Quantum linear system algorithm for dense matrices. Phys. Rev. Lett. **120**, 050502 (2018)

[35] Rebentrost, P., Steffens, A., Marvian, I., Lloyd, S.: Quantum singular-value decomposition of nonsparse low-rank matrices. Physical Review A **97**(1) (2018)

[36] Mitchell, T.M.: Machine Learning, Nachdr. edn. McGraw-Hill series in Computer Science. McGraw-Hill, New York (2013)

[37] Ventura, D., Martinez, T.: Initializing the Amplitude Distribution of a Quantum State. Foundations of Physics Letters **12**(6), 547–559 (1999)

[38] Long, G.-L., Sun, Y.: Efficient scheme for initializing a quantum register with an arbitrary superposed state. Physical Review A **64**(1), 014303 (2001)

[39] Mottonen, M., Vartiainen, J.J., Bergholm, V., Salomaa, M.M.: Transformation of Quantum States Using Uniformly Controlled Rotations. Quantum Info. Comput. **5**(6), 467–473 (2005)

[40] Plesch, M., Brukner, C.: Quantum-state preparation with universal gate decompositions. Physical Review A **83**(3), 032302 (2011)

[41] Cortese, J.A., Braje, T.M.: Loading Classical Data into a Quantum Computer (2018)

[42] Araujo, I.F., Park, D.K., Petruccione, F., da Silva, A.J.: A divide-and-conquer algorithm for quantum state preparation. Scientific Reports **11**(1), 6329 (2021)

[43] Park, D.K., Sinayskiy, I., Fingerhuth, M., Petruccione, F., Rhee, J.-K.K.: Parallel quantum trajectories via forking for sampling without redundancy. New Journal of Physics **21**(8), 083024 (2019)

[44] Low, G.H., Kliuchnikov, V., Schaeffer, L.: Trading T-gates for dirty qubits in state preparation and unitary synthesis (2018)

[45] Zoufal, C., Lucchi, A., Woerner, S.: Quantum generative adversarial networks for learning and loading random distributions. npj Quantum Information **5**(1), 1–9 (2019)

[46] Kuzmin, V.V., Silvi, P.: Variational quantum state preparation via quantum data buses. Quantum **4**, 290 (2020)

[47] IBM: IBM's Roadmap For Scaling Quantum Technology (2020). https://research.ibm.com/blog/ibm-quantum-roadmap

[48] Preskill, J.: Quantum Computing in the NISQ era and beyond. Quantum **2**, 79 (2018)

[49] Aleksandrowicz, G., et al.: Qiskit: An Open-source Framework for Quantum Computing (2021)

[50] Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., Alam, M.S., Ahmed, S., Arrazola, J.M., Blank, C., Delgado, A., Jahangiri, S., McKiernan, K., Meyer, J.J., Niu, Z., Száva, A., Killoran, N.: PennyLane: Automatic differentiation of hybrid quantum-classical computations (2020)

[51] Bergholm, V., Vartiainen, J.J., Möttönen, M., Salomaa, M.M.: Quantum circuits with uniformly controlled one-qubit gates. Physical Review A **71**(5), 052330 (2005)

APÊNDICE C – APPROXIMATED QUANTUM-STATE PREPARATION WITH ENTANGLEMENT DEPENDENT COMPLEXITY

Approximated quantum-state preparation with entanglement dependent complexity

Israel F. Araujo,[1,*] Carsten Blank,[2,†] Ismael Cesar,[1,‡] and Adenilton J. da Silva[1,§]

[1] *Centro de Informática, Universidade Federal de Pernambuco, 50740-560, Recife, Pernambuco, Brazil*
[2] *data cybernetics, 86899, Landsberg am Lech, Germany*

Ubiquitous in quantum computing is the step to encode data into a quantum state. This process is called quantum state preparation, and its complexity for non-structured data is exponential on the number of qubits. Several works address this problem, for instance, by using variational methods that train a fixed depth circuit with manageable complexity. These methods have their limitations as the lack of a back-propagation technique and barren plateaus. This work proposes an algorithm to reduce state preparation circuit depth by offloading computational complexity to a classical computer. The initialized quantum state can be exact or an approximation, and we show that the approximation is better on today's quantum processors than the initialization of the original state. We verified through experimental evaluation that the proposed method allows a more efficient initialization of distributions in a quantum state and approximate loading images for quantum machine learning tasks.

I. INTRODUCTION

Quantum devices can execute information processing tasks that classical computers cannot perform efficiently [1]. In some instances, this leads to exponential advantages in the solution of systems of linear equations [2] and principal component analysis [3]. Additionally, there are known advantages in Monte-Carlo sampling [4, 5] in which a squared increase of convergence can be attained. Furthermore, quantum machine learning applications [6–8] may exhibit heuristic advantages. For all these applications, the initialization of a n-qubit quantum state, commonly called quantum state preparation or initialization, see Fig.1, is an important step in quantum information processing. Encoding a N-dimensional (complex) vector requires n-qubits with $N = 2^n$ and quantum circuits with $O(2^n)$ controlled-NOT (CNOT) gates [9, 10]; these gates are the building blocks of today's quantum computers. Therefore, several works focus on the development of algorithms that supposes data-efficient initialization, as all above-mentioned quantum advantages could be undone when the conversion of classical data to quantum data becomes a bottleneck.

There are several quantum state preparation algorithms [11–14] with a lower bound of $O(2^n)$ CNOT gates to prepare an arbitrary quantum state with n qubits. Attempts to prepare quantum states more efficiently include a divide-and-conquer strategy that exchanges circuit depth by circuit width [14, 15], probabilistic approaches [16, 17], and strategies to initialize approximated quantum states [18–20]. Most recently, there are methods that focus on specific classes of quantum states. For instance, how to prepare uniform [21], sparse [13] or probability distribution [18] states. However, there is no clear understanding of which classes of quantum states can be created efficiently.

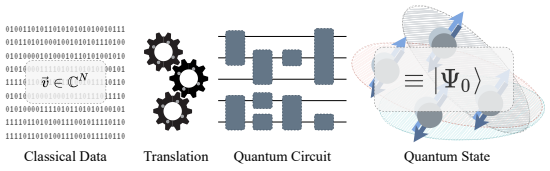


FIG. 1: The process of state preparation or initialization. Classical data is converted into vector form and a translation process encodes the information in operations performed on the quantum computer, commonly encoded by a circuit. Executing this circuit creates the initial state $|\Psi_0\rangle$.

Entanglement is one of the quantum resources that allow the development of more efficient algorithms and protocols. Its relationship to the algorithmic complexity of a quantum state has been demonstrated in previous work [22, 23] and plays a key role in many quantum applications, such as quantum communication, quantum error correction, and quantum secret sharing [24–29].

However, the circuit complexity of current algorithms to create a quantum state do not consider the resource of entanglement. The main goal of this article is to define a deterministic state preparation algorithm that creates circuits with depth as a function of entanglement, and show that for mapping the classical data into a Noisy Intermidiate-Scale Quantum (NISQ) device, one indeed wants to be in the regime of low-entanglement. After finalizing the state preparation, one can increase the amount of entanglement (exponential in the number of qubits) by applying unitary evolution. This path could lead to the solution of the so-called loading problem (LP) of classical-quantum information processing.

The proposed approach's advantage is that it can accelerate quantum applications that require the initialization of quantum states, in particular on noisy devices. Using

* ifa@cin.ufpe.br
† blank@data-cybernetics.com
‡ icsa@cin.ufpe.br
§ ajsilva@cin.ufpe.br

an approximation of the quantum state, the result is that the error introduced by the approximation is smaller than the error to encode the original state. The fundamental cause of this behavior is the difference in the number of noisy operations necessary between the circuits to encode the original versus the approximate state. But also apart from this obvious advantages in the NISQ-era, the proposed approach reduces the complexity to initialize a low-entangled quantum state into fault-tolerant quantum devices.

The remainder of this paper is organized into four sections. The main contributions of this work are described in sections II and III. Section II introduces the *Low-Rank State Preparation* (LRSP) algorithm, a modification of the so-called Plesch and Brukner (PB) algorithm [12]. LRSP algorithm reduces the computational cost when the input vector has a low Schmidt coefficient. In this way, the LRSP can be used to initialize general quantum states and also low rank quantum states. Section III introduces the *Bounded Approximation Error Algorithm* (BAA), a search algorithm that reduces the circuit depth of LRSP circuit given an allowable error. Some numerical tools and the connection to the geometric measure of entanglement are discussed. Section IV presents experimental results on how probability distributions can effectively be created and read out from actual pre-NISQ processors. Also, it shows how an end-to-end quantum machine learning approach can be improved through approximations of the initial quantum state. Section V summarizes the work and presents perspectives along with potential future research.

II. LOW-RANK STATE PREPARATION ALGORITHM

The proposed Low-Rank State Preparation algorithm is based on Plesch and Brukner (PB) state preparation algorithm [12]. The PB algorithm connects to the Schmidt-decomposition, given a quantum state $|\psi\rangle = \sum_j \lambda_j |j\rangle$, the first step of the algorithm runs a Schmidt decomposition

$$|\psi\rangle = \sum_{i=1}^{k} \sigma_i |i_A\rangle |i_B\rangle \tag{1}$$

where the state is factored in two quantum subsystems $\mathcal{H}_A$ and $\mathcal{H}_B$, values σ_i are the Schmidt coefficients, $\{|i_A\rangle\} \in \mathcal{H}_A$ and $\{|i_B\rangle\} \in \mathcal{H}_B$ are orthonormal basis, and $1 \leq i \leq \min(\dim(\mathcal{H}_A), \dim(\mathcal{H}_B))$. The second step performs the initialization of the quantum state $\sum_i \sigma_i |i\rangle |0\rangle$ in the first quantum register. The objective of this step is to initialize a state with the Schmidt coefficients. The third step applies $\lfloor n/2 \rfloor$ CNOTs to create the state $\sum_i \sigma_i |i\rangle |i\rangle$. Let U and V be unitaries, where $U|i\rangle = |i_A\rangle$ and $V^T|i\rangle = |i_B\rangle$. The last step of the algorithm applies U in the first register and V^T in the second register. A quick review of the algorithm is schematically shown in Fig. 2d.

The proposed *Low-Rank State Preparation* differs from PB algorithm when the Schmidt measure $m = \lceil \log_2(k) \rceil < \lfloor n/2 \rfloor$ and also by the use of isometries. Theorem 1 establishes the CNOT gates count needed when a low-rank representation of a state can be found using the Schmidt decomposition, as well as approximating the state by truncating the Schmidt coefficients.

Theorem 1 (Low-Rank State Preparation)**.** *Given Eqn. (1) with the Schmidt measure $m = \lceil \log_2(k) \rceil$ the low-rank state preparation has a complexity of*

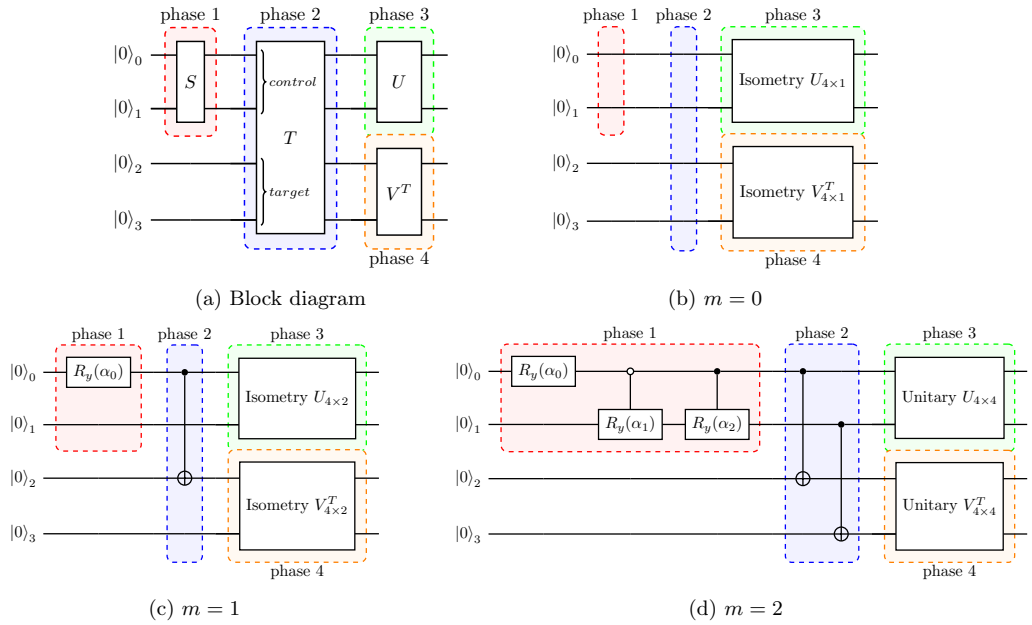
condition	CNOT count
$0 \leq m < n_A$	$O(2^{m+n_B})$
$m = n_A$	$O(2^n)$

Proof. Let $|\psi\rangle$ be a n-qubit quantum state with Schmidt decomposition $|\psi\rangle = \sum_{i=1}^{k} \sigma_i |i_A\rangle |i_B\rangle$ where subsystem $\mathcal{H}_A$ has n_A qubits ($1 \leq n_A \leq \lfloor n/2 \rfloor$), subsystem $\mathcal{H}_B$ has $n_B = n - n_A$ qubits and k is the Schmidt rank (also known as Schmidt number).

If the Schmidt measure is $m = \lceil \log_2(k) \rceil < n_A$, the operator S (Fig. 2a) initializes a state with m qubits in phase 1 (Fig. 2b and Fig 2c), instead of n_A qubits (Fig. 2d). The second phase requires m CNOT gates, as they are not needed where the control qubit is $|0\rangle$. In this circuit configuration, m is the Schmidt measure and quantifies the entanglement between subsystems $\mathcal{H}_A$ and $\mathcal{H}_B$. Finally, in phases 3 and 4, the matrices U and V^T are isometries 2^m–to–2^{n_A} and 2^m–to–2^{n_B} which require $O(2^{m+n_A})$ and $O(2^{m+n_B})$ CNOTs.

The number of CNOTs of the complete LRSP circuit is $O(2^{m+n_B})$ because the cost of the isometry V^T (phase 4) dominates the cost of the algorithm ($n_B \geq n_A$). The best-case occurs when the bipartition is not entangled $|\psi\rangle = |\psi_A\rangle |\psi_B\rangle$. Therefore, if rank = 1 ($m = 0$ e-bits) there are no operations in phase 1 (to encode singular values), no entanglement between the two subsystems in phase 2 (since there is 0 e-bits between the subsystems), and there are one 1–to–2^{n_A} and one 1–to–2^{n_B} isometries in phases 3 and 4 (Fig. 2b) (which is equivalent to two parallel sub-state preparations in n_A and n_B qubits). If all qubits of the state are separable, applying recursively the same algorithm to prepare the sub-states in phases 3 and 4 generates a circuit without CNOT gates. If the state is initially separable but not all sub-states of the recurrence (some qubits are not separable), the cost of the state preparation is $O(2^{n_e})$, where n_e is the number of qubits of the higher entangled subsystem. The worst case occurs when $m = n_A$, recovering the $O(2^n)$ complexity of the original PB circuit (Fig. 2). □

In the best case (separable states), the LRSP algorithm creates circuits with a constant depth. In the worst case, the LRSP algorithm creates circuits with depth $O(2^n)$ and is a competitive algorithm compared with the state-of-art deterministic state preparation algorithms for general state preparation without auxiliary



(a) Block diagram

(b) $m = 0$

(c) $m = 1$

(d) $m = 2$

FIG. 2: Schematics of the low-rank approximation algorithm ($n = 4$ and subsystem $\mathcal{H}_A = \{0, 1\}$). (a) Block diagram circuit. Operator S is responsible for encoding 2^{n_A} ($n_A = n/2$) SVD singular values. In this example, operator S encodes a maximum of four amplitudes (two qubits). Operator T is a sequence of CNOTs, controlled by the first half of qubits and targeting the second half (one by one). Operators U and V are the two SVD unitaries. For full-rank ($m = n_A$, in units of e-bits), U and V^T are encoded as unitaries of dimension $2^{n_A} \times 2^{n_A}$. For a lower rank ($m < n_A$), the operators are encoded as isometries of dimension $2^{n_A} \times 2^m$. When $m = 0$ (rank $= 1$), the isometries of dimension $2^{n_A} \times 1$ are equivalent to amplitude encoding with input vectors of length 2^{n_A}. Plesch's original work describes the four individual phases. (b)(c)(d) Detailed views of (a) for rank $= 1$ ($m = 0$), rank $= 2$ ($m = 1$) and rank $= 4$ (full-rank, $m = 2$). In this example, phase 1 uses Möttönen's state preparation but could use any amplitude encoding algorithm, including the low-rank state preparation algorithm.

qubits. Table I compares LRSP circuit depth and number of CNOTs with previous state preparation algorithms. Two experiments were performed. Column (Ent.) shows results for a random 10-qubit quantum state initialization, and column (Sep.) for a random separable 10-qubit quantum state. The LRSP results depend on the choice of unitary and isometry decompositions. The methods used are Quantum Shannon Decomposition (QSD) for unitaries [30], Cosine–Sine Decomposition (CSD) [31] for 2^{n-1}–to–2^n isometries, and Column–by–Column Decomposition (CCD) for other isometries [31].

Method	Script	Ent.		Sep.	
		CNOTs	Depth	CNOTs	Depth
Low Rank	[32]	915	1047	0	2
PB [12]	[32]	919	1057	919	1058
Isometry [31]	[33]	1013	4055	1013	3871
Multiplexor [30]	[33]	2026	4064	2026	4064

TABLE I: Depth and number of CNOTs comparison between LRSP and other state preparation algorithms. Experiments were performed without circuit optimization.

A. Low-Rank Approximation

The LRSP algorithm also allows a low-rank approximation limiting the Schmidt rank in exchange for an error. The fidelity loss can be used to quantify the loss by the approximation.

Definition 1. *Given the low-rank parameter r, the approximated state is denoted as*

$$\left|\psi^{(r)}\right\rangle = \sum_{i=1}^{r} \sigma_i \left|i_A\right\rangle \left|i_B\right\rangle \tag{2}$$

with coefficients for $1 \leq r \leq k$, i.e. $\sigma_j = 0$, $r < j \leq k$.

With the LRSP algorithm, it is possible to partially ($r > 1$) or completely ($r = 1$) disentangle subsets

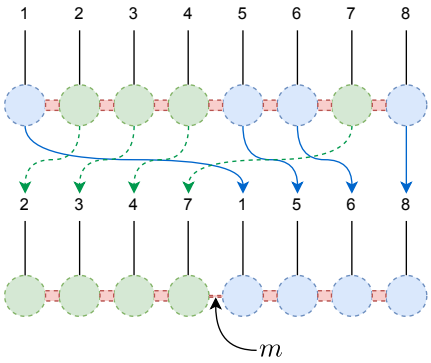


FIG. 3: Schematic of the logical swap of qubits. By rearranging the qubits it is possible to find a bipartition such that the bond dimension between the partitions is small, which means that these partitions are not strongly entangled. The indices indicate the position of the qubit in the quantum circuit.

of qubits while the introduced fidelity loss $l(r, |\psi\rangle) := (1 - |\langle\psi, \psi^{(r)}\rangle|^2) = \sum_{i=r+1}^{k} |\sigma_i|^2$ scales with the Schmidt coefficients that are dropped. The possibility of allowing arbitrary bipartitions – with not-necessarily connected qubits – is not as readily obvious. By rearranging sites (Fig. 3) between partitions of any size within the allowed range ($1 \leq n_A \leq \lfloor n/2 \rfloor$ and $n_B = n - n_A$), one can find the configuration that exhibit a low bond-dimension at some boundary. It is then possible to approximate with lower error, potentially disentangling these bipartitions. One then can recursively apply this approach on the two resulting partitions separately; a search algorithm with a given maximal approximation error can then find the optimal approximation of an arbitrary quantum state. This Bounded Approximation error Algorithm (BAA) is applicable to any quantum state of interest, including, for example, vector encoding for inverting matrices using HHL [3], loading data into a quantum machine learning model [6, 7] or using quantum simulation of stochastic processes [34].

III. BOUNDED APPROXIMATION ALGORITHM

By design, the low-rank approximation only applies to bipartite systems, yet it can be used hierarchically to enable the analysis of multipartite quantum systems [35] by a recursive algorithm. The proposed search algorithm allows an approximation up to a given error bound $l_{\max}$ – if possible – and thus saves the use of CNOT gates while providing a guarantee of proximity to the original state. It is a bounded approximation error state preparation algorithm (BAA) that potentially has a classical exponential run-time with respect to the number of qubits of the state. As it is a branch-and-bound algorithm using breadth-first search, the complexity usually converges faster [36, 37]. The full set of pseudocode which describes the algorithm is printed in the supplementary information. In what follows, we want to outline the core principles of the algorithm.

The approximate state preparation algorithm works as follows. Any (pure) n-qubit quantum state $|\psi\rangle$ will allow for a total of

$$\mathcal{B}(n) = \sum_{t=1}^{\lfloor n/2 \rfloor} \binom{n}{t} \tag{3}$$

bipartitions (also called branches), each is written as $b \in [1\,..\,\mathcal{B}(n)]$ with the Schmidt number bounded by $k(b) = \min\{\dim(\mathcal{H}_1^b), \dim(\mathcal{H}_2^b)\}$. By letting the low-rank parameter $1 \leq r \leq k(b)$ be $r = 1$, each bipartition creates a disentangled pair of smaller states. Combining this pair by the tensor-product returns an approximation of the original state. Additionally, it is possible by the complexity analysis of Theorem 1 to predict the number of CNOT gates saved.

Starting from $|\psi\rangle \in \mathcal{H}$, the algorithm is branching into all possible bipartitions, $(\mathcal{H}_1^b, \mathcal{H}_2^b)$. Recursively, each partition $\mathcal{H}_i^b$ can in turn be branched once again, as long as $\mathcal{H}_i^b$ $(i = 1, 2)$ is not one single qubit or the total fidelity loss has not exceeded the error bound $l_{\max}$. In order for the search-algorithm to work, each node needs to be able to know the total fidelity loss, the total saved number of CNOT gates and its partitioned Hilbert spaces. Due to this structure, the algorithm terminates in the worst case with an exponential number of steps. These mentioned properties are summarized.

Lemma 1. *The fidelity and the saved CNOT gates of each branch can be recursively calculated.*

A small result with an impact is the fact that the fidelity/fidelity loss and CNOT gate savings can be recursively calculated from one branch to another. The lemma is important for the computation of the search tree in the BAA algorithm, as it is possible to parallelize it.

Proof. One must show that the total fidelity loss over several steps in the tree can be recursively calculated. Let l_{b_p} and l_{b_c} be the fidelity loss of the parent and child node, respectively. Then the fidelity of both is $(1-l_{b_p})(1-l_{b_c})$, hence $l_{\text{tot}} = 1 - (1 - l_{b_p})(1 - l_{b_c})$. More generally, $l_{\text{tot}} = 1 - \prod_i (1 - l_{b_i})$ for a path on the tree $b_1, \ldots, b_t$. □

Proposition 2. *The BAA-Algorithm's search tree has n levels, and therefore terminates. The time and space complexity is exponential in n in its worst case.*

When spanning the search tree (see Suppl. Information Pseudocode 1), each child of a parent node will be created by choosing exactly one state and doing exactly one bipartition. The bipartitions of our focus are the t qubits vs. the rest ones, where t is defined in Eq. (3). Say, among the k states of the parent node, #1 is selected to be partitioned, with $\dim(\mathcal{H}_1) = n_1$. The split

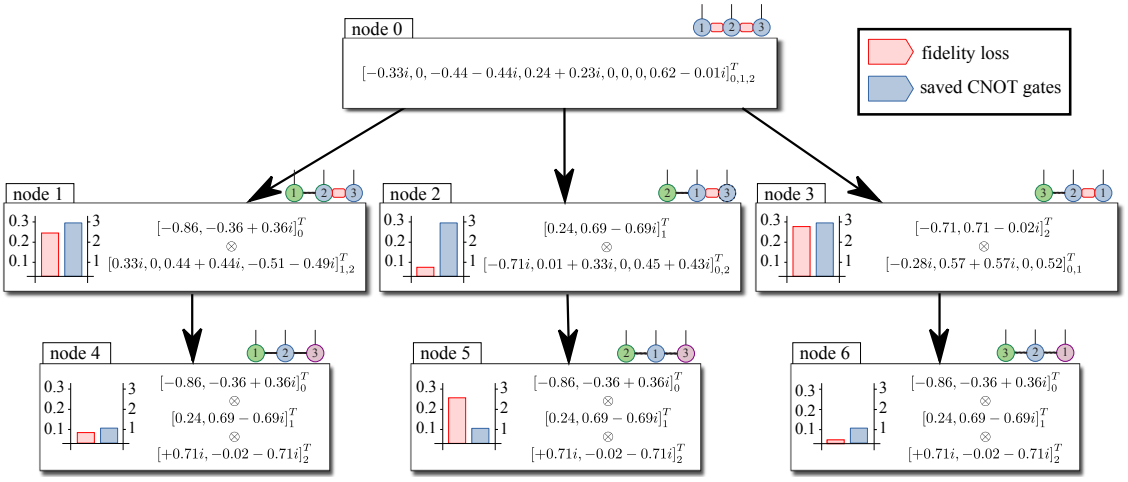


FIG. 4: An example of a three-qubit state that is disentangled into product states (leaves). It starts with one vector in the root and then all $\mathcal{B}(3) = 3$ bipartitions are branched off, creating each two states. From there, the larger subspace has two qubits, so it should have $\mathcal{B}(2) = 2$ branches. But in this case, the second branch is equal to the complement of the first, so it is redundant and omitted. Each node has a fidelity loss (red column) and the number of CNOT gates saved (blue column) using the proposed algorithm. The sub-indices of the vectors are the qubit partitions associated with each state. Adding the weights from root to any leaf gives the total fidelity loss of ≈ 0.307, meaning that the reduced product state to its original state have an overlap of ≈ 0.693, but with a saving of 4 CNOT gates. The adaptive approximation algorithm goes to ask if we can save some CNOT gates with an acceptable fidelity loss. If $l_{\max} = 0.1$, there is the second bipartition from the root node (node 2) that introduces only a fidelity loss of ≈ 0.058 by simultaneously saving three CNOT gates.

of t qubits vs. rest is done, and in this manner $n_1 - 1$ levels will be traversed until this state has been partitioned to a tensor product of single qubit states. Meanwhile, all $k - 1$ original states were kept as they were. Now the partition #2 is selected, and in turn takes $n_2 - 1$ steps to a tensor product of single qubit states. This goes on with all the remaining $k - 2$ partitions. So in total, there are

$$\sum_{l=1}^{k} (n_l - 1) = n - k \tag{4}$$

steps until all k partitions have been partitioned into tensor products of single qubit Hilbert spaces. As this is the longest path possible, the BAA algorithm terminates. Now, we show the runtime complexity of the BAA algorithm.

Lemma 2. *Given a k-fold partitioning of $\mathcal{H}$, denotes as $(\mathcal{H}_1, \ldots, \mathcal{H}_k)$, the number of children of this node is*

$$\mathcal{B}_k(n) = \sum_{i=1}^{k} \mathcal{B}(\dim \mathcal{H}_i) \tag{5}$$

As a special case, we see $k = 1$, which is directly $\mathcal{B}(\dim \mathcal{H})$.

Proof. Given k already partitioned subspaces, the search algorithm selects one of the partitions $(\mathcal{H}_i,\ i = 1, \ldots, k)$ and carries out all $\mathcal{B}(\dim \mathcal{H}_i)$ branches, while leaving all the other $k - 1$ original partitions intact. This is then repeated, so the total number of children this node has as claimed. □

After each step, the number of partitions thus changes from $k \mapsto k + 1$ as one partition is created. Of course, the sum of all dimensions must be equal to the dimension of the original Hilbert space n. Thus, each level of the search tree is identified by the number k of partitions. The maximum number of levels is therefore always n. The total number of nodes is then computed as follows.

The first level has exactly 1 node. The second level has $\mathcal{B}_1(n)$ nodes. The third level has $\sum_{b=1\ldots\mathcal{B}_1(n)} \mathcal{B}_2(n; b)$ and the fourth level $\sum_{b_1=1\ldots\mathcal{B}_1(n)} \sum_{b_2=1\ldots\mathcal{B}_2(n;b_1)} \mathcal{B}_3(n; b_2)$ up until the n^{th} level, we have,

$$\sum_{b_1=1}^{\mathcal{B}_1(n)} \cdots \sum_{b_{n-1}=1}^{\mathcal{B}_{n-1}(n;b_{n-2})} \mathcal{B}_n(n; b_{n-1}) \tag{6}$$

This is clearly exponential in n. A tree of an example three-qubit state is shown in Fig. 4. With an $l_{\max} = 0.1$ only one bipartition is possible within this error bound, and three CNOT gates are saved.

Albeit the fact that the BAA algorithm uses low-rank approximations with $r = 1$ to fully disentangle the bipar-

titions (Fig. 2 and Fig. 4), it is possible to include low-rank approximations with $r > 1$ when complete disentanglement is no longer within $l_{\max}$. This allows fine-tuning the partial disentanglement of states, as each value of the parameter r may achieve reductions in the number of CNOTs that would not be possible with the original approach, producing additional branches from the tree nodes. Therefore, this advantage comes with a larger search space, and increases the cost of the algorithm.

While the proposed approximate state preparation algorithm reduces the quantum circuit complexity, the BAA has an exponential preprocessing cost. The naive ("brute-force" breadth-first) algorithm is already a solution for those researchers who need to work on small-qubit experiments and currently have no alternatives to an efficient quantum initialization on noisy devices. But a scalable workaround is necessary. One way is to use a greedy approach [38, 39] that seeks to reduce the computational cost, making BAA a useful solution for general problems.

The greedy strategy proposes that branching from a node is limited to a qubit-by-qubit analysis, selecting only one representative of the partitions of size k where $1 \leq k \leq \lfloor n/2 \rfloor$. The increment in the partition size is done by choosing the qubit with the lowest fidelity loss when removed from the remaining entangled subsystem. As an example, on a seven qubit state, the best 1 vs. 6, 2 vs. 5 and 3 vs. 4 bipartitions are attempted, and the best one is selected locally and then propagated in the recursion. This approach implies an exponential reduction in the number of bipartition combinations, of which one is chosen, producing only one path.

The greedy algorithm reduces the search-problem exponentially, yet this will necessarily reduce its ability to find a good CNOT reduction. The advantage over the breadth-first (brute-force) strategy is that it will never exceed the approximation configured, but the downside is that the CNOT complexity to attain this approximation could be far from optimal. We use seven qubit random circuits [40] of linear depth to generate 1000 quantum states and apply different strategies and $l_{\max}$ to them. It is obvious that the greedy strategy performs less well for low $l_{\max}$ (Fig. 5).

Regarding the approximation quality, both the greedy and brute-force approaches find approximations that to each other scatter quite evenly, if $l_{\max}$ is high, this scattering is less pronounced. The search of the best node will always reject a solution that exceeds the $l_{\max}$ value and it is always helpful to reduce the fidelity loss, but the cost of CNOT gates will outgrow the impact on NISQ devices. We can conclude on this survey that the greedy algorithm will miss optimality for low $l_{\max}$.

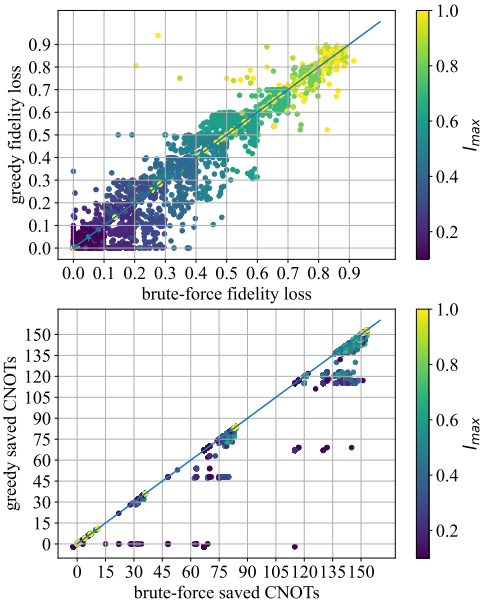


FIG. 5: Difference of greedy strategy saved CNOT gates compared to the brute-force search algorithm (BAA). The mean difference is an indicator how effective an approximation really is, and the lower the difference, the better. The greedy algorithm fares relatively well when the maximum fidelity is large, but introduces large inefficiencies when it is low. The most significant applications are most likely the low loss regimes, so that the greedy algorithm will need improvements. The data was created on a simulation with 1000 randomly initialized states of seven qubits.

A. Product State Approximations

In view of treating quantum state approximation, one could be naturally inclined to ask for the best product state approximation. This is indeed a measure for entanglement, the so-called geometric measure of entanglement. It was first introduced by Shimony [41] and refined by Barnum & Linden [42]. The definition for this measure of entanglement is given by

$$E_g(|\psi\rangle) = \min_{|\phi\rangle} \| \, |\psi\rangle - |\phi\rangle \, \| \tag{7}$$

where the minimization is over all states $|\phi\rangle$ that are product states [35], i.e., $|\phi\rangle = \otimes_{l=1}^{n} |\phi^l\rangle$ with each $|\phi^l\rangle$ being a local state. That this value is a useful measure of entanglement was shown by Wei et al. [43], it is an entanglement monotone.

The geometric measure of entanglement is related to the bounded approximate algorithm (BAA) in the following way. Theoretically, the greatest fidelity loss that any quantum state can experience under the BAA should

never be higher than it. Therefore, the geometric entanglement indicates how far an approximation can get. This shows that low-entanglement states can be encoded with little loss as a product state. However, looking at the BAA algorithm, the sequence of nested SVDs is not optimal, and there may be resulting product states with a higher fidelity loss.

The particular case where the BAA partitions are split in half and only one branch selected is equivalent to the Hierarchical Tucker Format (HTF) with a binary dimension tree (called Canonical Dimension Tree) [44]. The HTF can be used to estimate the maximum possible fidelity loss with an error less or equal to $\sqrt{2n-3}\|A - A^{\text{best}}\|$, where A^{best} is the best approximation of the tensor A, and n is the dimensionality of the problem. The number of bipartitions performed to estimate the maximum fidelity loss is linear in the number of qubits. Since BAA performs the SVD from already truncated tensors, the cost of HTF is given by Lemma 26 of Ref. [44].

IV. THE LOADING PROBLEM ON NISQ DEVICES

The LRSP algorithm (with partitions defined by the Bounded Approximation Error Algorithm) was designed to solve a class of problems on NISQ devices, namely loading data. To solve (or avoid) the loading problem [6], an approximation of the quantum state that uses less entanglement is created. In this Section we present two applications of the LRSP: initialize i) a Probability Distribution Function ii) images to and end-to-end quantum machine learning model. In an experimental evaluation, the approximated circuit created with the LRSP algorithm allows for improvements in both examples.

A. Experiments on Probability Density Functions

The BAA proves to be efficient when working with specific classes of quantum states found in the quantum finance [13, 18, 21]. Indeed, encoding probability distributions on the amplitudes of a 7-qubit quantum state show that even a modest fidelity loss (≤ 0.02) results in a significant reduction in the number of CNOTs.

To encode a continuous probability density function (PDF) into the amplitudes of a quantum state it is necessary to construct its discrete analog. In general, this construction is based on preserving one or more properties of the continuous distribution [45].

There are several methods by which a discrete random variable can be constructed from a continuous one. Here we employ Methodology-V described in Ref. [45] where the cumulative distribution function (CDF) of a discrete random variable Y maintains the form of the CDF of a continuous random variable X. The probability mass function (PMF) of Y is built from the CDF of X $F_X(x) = Pr(X \leq x)$ and is given by

$$P(Y = k) = F_X(k + \delta) - F_X(k - [1 - \delta]) \qquad (8)$$

where $0 < \delta < 1$ and $k = \{0, 1, 2, \dots\}$.

In the experiments, the interval $0 \leq x \leq 20$ is divided into 2^7 discretization points. Therefore, the distance between consecutive points is $d = 20/(2^7 - 1)$. By choosing $\delta = d/2$, the parameters of Eq. (8) are set to

$$P(Y = k) = F_X(k + d/2) - F_X(k - d/2) \qquad (9)$$

for $k = \{0, d, 2d, \dots, (2^7 - 1)d\}$.

Equation (9) is used to generate the discrete probability distributions analogous to the continuous ones characterized by the normal, log-normal, Laplace and semicircular PDFs.

distribution	mean	variance	skewness	kurtosis
normal	10.0	4.0	0.0	0.0
log-normal	3.297	18.68	6.185	110.9
Laplace	10.0	8.0	0.0	3.0
semicircular	10.0	16.0	0.0	-1.0

TABLE II: First four moments of the probability distributions.

Each point of the discretization corresponds to an amplitude of the state vector. Note that, given the conservation of probability, each amplitude must be the square root of the PMF at these points. The result of the discretization is employed to constructs Figure 6 and Figure 7.

The state vectors of the distributions were encoded via BAA and estimated by measurements on actual quantum devices. All experiments make use of the breadth-first search (brute force) strategy and partial disentanglement of states enabled. Each result is an average of 10 runs with 8192 shots each. The experiments leading to Figure 6 were performed at points $l_{\max} = \{0.0, 0.02\}$ using device ibmq_perth, while those leading to Figure 7 were performed at points $l_{\max} = \{0.0, 0.005, 0.01, \dots, 0.055\}$ using devices ibmq_jakarta and ibmq_casablanca.

When preparing the original states ($l_{\max} = 0.0$), the distributions need circuits with 105, 109, 28, and 99 CNOTs, respectively. The difference between the CNOT numbers is due to the LRSP's ability to adjust the complexity of the circuit to the degree of state entanglement. If a maximum fidelity loss of $l_{\max} = 0.02$ is allowed, the effective fidelity loss achieved by the BAA to encode the approximate state for each distribution is the closest possible to $l_{\max}$, i.e., 0.0089, 0.0159, 0.0081, and 0.0158. With such approximation, the BAA can encode the first three distributions using only *six* CNOTs and the semicircular with *three* CNOTs. This relatively low introduced error compared to the extreme reduction of entangling operations leads to better results in terms of mean-absolute-error (MAE). Indeed, when executing

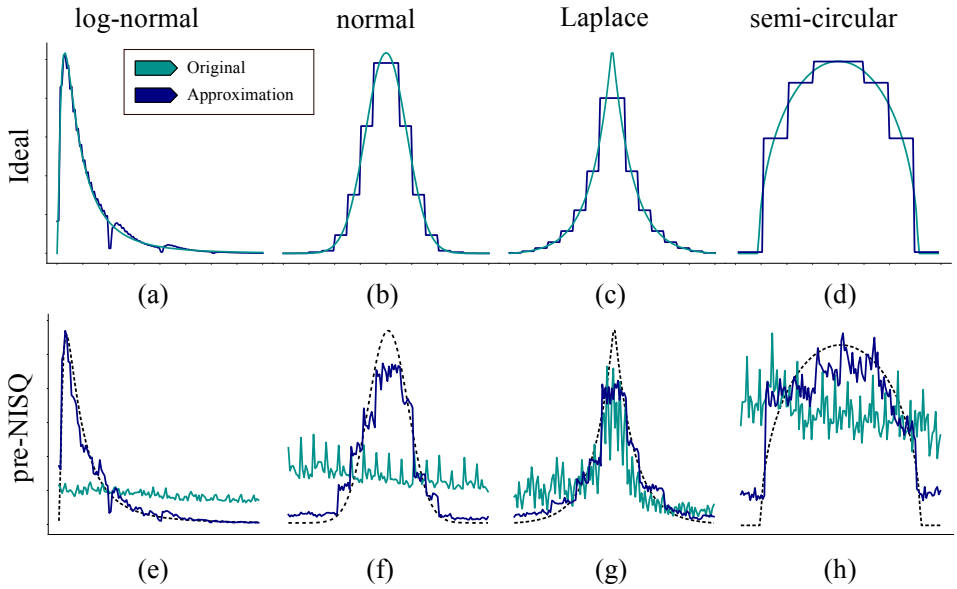


FIG. 6: Visualization of probability distributions encoded on the amplitudes of a 7-qubit quantum state. (a)-(d) Ideal values of the exact (green line) and approximate (blue line) distributions. (e)-(h) Values of the exact and approximate distributions encoded via BAA and estimated by measurements on ibm_perth device. In the actual device, the encoding of the approximations performs closer to the ideal (black dotted line) than the exact distribution encoding. Each result is an average of 10 runs with 8192 shots.

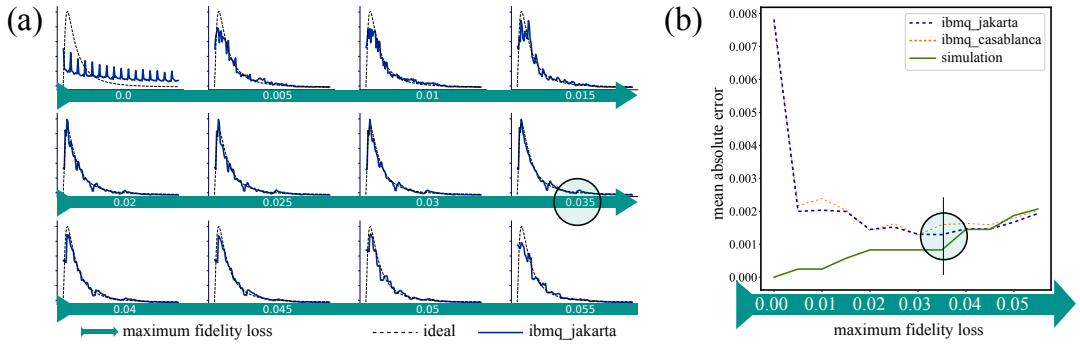


FIG. 7: BAA experimental results with a 7-qubit log-normal state on two IBM quantum devices (ibmq_jakarta & ibmq_casablanca), with a geometric entanglement of ≈ 0.0534. (a) Several approximations in ascending order show that the exact state preparation fails. All subsequent approximations decrease the MAE until $l_{max} = 0.035$. After that, it increases until the product-state (at $l_{max} = 0.055$) is reached. (b) Comparison between two devices and an ideal simulation. The highlighted point indicates the lowest MAE achieved by the devices.

an experiment on the ibm_perth, the high number of CNOTs to initialize the exact state obscure the probabilities whereas the BAA approximation keep distribution specific features closer to the actual distribution, see Figures 6(a)–6(h).

One can also load an approximate probability distribution into a quantum state using Quantum Generative Adversarial Networks (qGANs) [18]. A Generative Adversarial Network [46] is a machine learning model composed of a neural network pair often referred to as generator and discriminator. Discriminator and generator networks are trained competitively. The generator network produces random synthetic data and the discriminator distinguishes real and synthetic data. A preliminary experiment was conducted to load a log-normal PDF into a noisy 7-qubit quantum device. However, the frequencies returned by the generator are not as approximate to the target discretized distribution frequencies as the

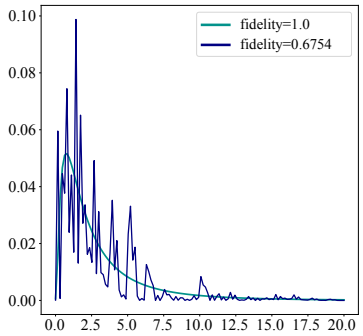


FIG. 8: Comparison between the ideal values of the exact frequencies for the log-normal distribution (green line) and the frequencies produced by qGAN using 21 CNOTs (blue line).

ones produced using the BAA method in terms of state fidelity (See Fig. 8). The qGAN was also applied to other distributions – normal, Laplace, and semicircular. It was found that these approximations are deviating substantially from expectation, noting at this point that each qGAN learning protocol takes about 4.5 hours on a GPU based qiskit aer noiseless simulation. Given this context, the low-rank state preparation method with the BAA search provides an efficient and competitive method to load a PDF into a noisy quantum device.

B. Experiments on VQC

In quantum machine learning, one of the best-known applications is to train a parameterized quantum circuit in order to learn a decision hyperplane [47] and thus be able to classify binary problems. In this setting, the input data is encoded by a feature map into a quantum state. Havlicek et al. [47] have hypothesized that their feature map has quantum advantage properties, yet any feature map, including a regular state preparation [12, 30, 48], is a valid option.

These experiments evaluate the BAA using brute-force strategy and partial disentanglement of states as a quantum feature map [47] to encode data in the amplitudes of an entangled state [12, 30, 48]. The accuracy of a quantum variational classifier (VQC) [47, 49] is used as a metric to evaluate the state preparation.

The classifier model is a hardware efficient parameterized circuit divided into two layers that form a block [47, 50, 51]. The first layer contains parameterized single qubit rotation gates, followed by the second with two-qubit entangling gates. Usually, the block is repeated to achieve better results [49]. The circuit is completed with an additional layer of rotations. Multiple measurements are carried on any qubit to approximate an expectation value. The circuit model TwoLocal from the Qiskit circuit library was chosen for this work. It is a fully entangled model composed of R_y, R_z, and controlled-Z gates with no block repetition.

The Optical Recognition of Handwritten Digits Data Set was used [52]. A binary dataset with 100 samples was extracted from the original dataset. The two classes of the new dataset are 0 and 1, each with 50 samples of 64 attributes. Mean test accuracy is computed on a random initialization for each feature map configuration.

Qiskit's VQC implementation was modified to accept the BAA encoding as a feature map and used to carry on the classical-quantum hybrid approach, in which the optimization procedure is processed on a classical computer to determine a set of parameters for the parameterized quantum circuit [51, 53]. Simulations of the hybrid classification algorithms were performed using Qiskit's Aer simulator with approximately 4500 circuit executions for each experiment configuration – 1024 shots per execution. For the experiments with noisy simulation, the noisy model and the coupling map were imported from ibmq_jakarta device. Ten samples of each class of the binary dataset were used as a test set and the remaining 80 samples (40 for each class) as a training set. As preparation for amplitude encoding, each data vector element of the dataset was standardized for a Gaussian around zero with unit variance, rescaled within the range [0, 1], and normalized. Our simulation employs the Simultaneous Perturbation Stochastic Approximation (SPSA) [50] for the optimization process with a mini-batch size of 1/10 of the training set size.

Ideally, the accuracy should be at a nominal level, which can be established with a noiseless simulation. However, the experiment on a NISQ device introduces noise, reducing the classification accuracy. Still, the BAA produces better results on a noisy device due to the lower circuit complexity. Indeed, the reduction of entangling gates shows that even strong approximations still result into higher accuracy (Figure 9). For a small training data set, one can train a variational quantum circuit with low-rank images, but we suspect that this method will preserve its value for larger training data.

V. CONCLUSION

With the help of Low-Rank State Preparation algorithms, significant improvements in the complexity of two-qubit entangling gates can be attained. The proposed LRSP algorithm exploit this fact by connecting state preparation complexity with the entanglement structure of the quantum state. This work highlights that classical data can be rearranged such that the entanglement structure attains an easier to approximate structure, as can be seen in Figure 3 and Figure 4. This directly bestows the data with the topological structure of qubit systems. In particular, by logically swapping qubits between partitions, one can find more local behavior. This makes it easier to simulate classically, but also, by virtue of the above-mentioned circuit designs, less com-

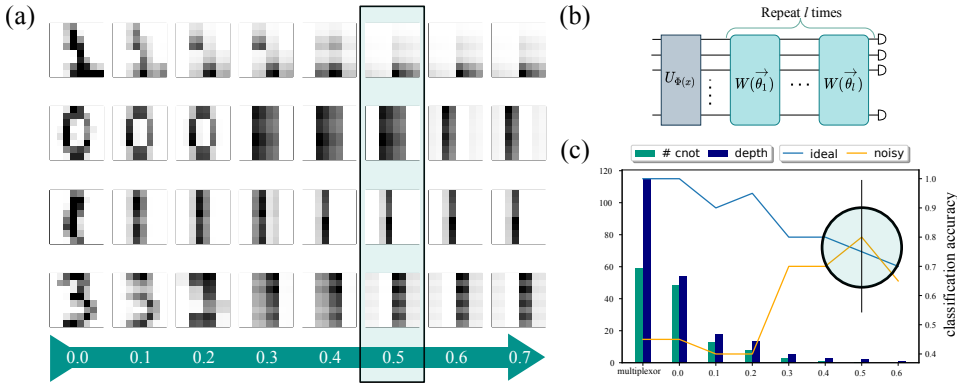


FIG. 9: Application of the BAA to machine learning problems helps to increase classification accuracy. (a) The effect of BAA on the input data (digits) for different maximum fidelity loss parameters. (b) The classifier's feature map U_ϕ encodes the data in the amplitudes of a quantum state using BAA. (c) The bars show the average number of CNOT gates and circuit depth produced by the multiplexor algorithm [48] and the BAA. Blue and yellow lines indicate ideal and noisy simulation accuracy results. The classifier delivers the best accuracy at the maximum fidelity loss of 0.5.

plex to create on a quantum computer. To showcase the immediate advantages of this approach, we present example applications in loading PDF and quantum machine learning. The experiments with VQC show that BAA allows adjustment of the input data according to the device's capability to manage entanglement, improving the classifier's performance. Additionally, VQC applications have the characteristic of loading the same input vector many times, which amortizes the computational time of the BAA preprocessing.

Recent approaches for approximating quantum states use variational quantum circuits in which parameterized single qubit and entangling operations are intertwined and adjusted according to a gradient descent method while minimizing/maximizing a loss function, usually taken from the measurement output of an experiment. In particular noteworthy is the qGAN method [18] and the AAS method [19] which have been proposed as a viable alternative to standard state preparation in order to achieve good results in the NISQ era. These methods try to emulate local-interaction Hamiltonians, and as such they in theory must be able to converge to the desired quantum state. We found in a preliminary comparison that the BAA performs better for several reasons. However, we note that an exhaustive study is necessary, yet is beyond the scope of this work. To summarize, the BAA is based on purely classical algorithms being very efficient in today's hardware with standard methods to apply for approximations (instead of a full search, we have greedy and Monte-Carlo as options) and the method does not suffer from initial conditions failures (such as barren plateaus).

This work also leaves some open technical questions. The BAA algorithm needs to disentangle in the first level of the tree if it is to succeed to the next. This involves the existence of a bipartition that will break the system into two uncorrelated parts. For some quantum states, this may be too much to ask for. Even though some gate-complexity reduction can be attained by the low-rank state preparation, it seems that this approach may not exploit the quantum state's entanglement topology to its maximum.

Apart from the advantages that this view of state preparation for classical data can have for the NISQ-era, and beyond, there is another promising application, which is to enhance quantum adiabatic state preparation [54, 55] by preparing a low-entanglement bounded approximation error state and find its Hamiltonian for which it is an eigenvalue and then apply the adiabatic theorem. The goal would be to potentially skip spectral gap bottlenecks and therefore make either the state preparation possible or reduce the adiabatic time significantly. In fact, a possible way forward could be to reach high-entanglement states by this approach. It remains unclear how this could actually be achieved, but the applications would be obviously beneficial.

ACKNOWLEDGMENTS

This research is supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), and Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco - FACEPE. We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team.

11

DATA AVAILABILITY

The sites `https://github.com/qclib/qclib-papers` and `https://github.com/qclib/qclib` contain all the data and the software generated during the current study.

[1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, *et al.*, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).

[2] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, Physical review letters **103**, 150502 (2009).

[3] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum principal component analysis, Nature Physics **10**, 631 (2014).

[4] A. Montanaro, Quantum speedup of Monte Carlo methods, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **471**, 20150301 (2015), 1504.06987.

[5] P. Rebentrost, B. Gupt, and T. R. Bromley, Quantum computational finance: Monte Carlo pricing of financial derivatives, Physical Review A **98**, 022321 (2018), 1805.00109.

[6] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature **549**, 195 (2017).

[7] M. Schuld and F. Petruccione, *Supervised Learning with Quantum Computers*, 1st ed. (Springer Publishing Company, Incorporated, 2018).

[8] C. Blank, D. K. Park, J.-K. K. Rhee, and F. Petruccione, Quantum classifier with tailored quantum kernel, npj Quantum Information **6**, 41 (2020).

[9] S. Aaronson, Read the fine print, Nature Physics **11**, 291 (2015).

[10] F. Leymann and J. Barzen, The bitter truth about gate-based quantum algorithms in the NISQ era, Quantum Science and Technology **5**, 044007 (2020).

[11] V. Bergholm, J. J. Vartiainen, M. Möttönen, and M. M. Salomaa, Quantum circuits with uniformly controlled one-qubit gates, Physical Review A **71**, 052330 (2005).

[12] M. Plesch and Č. Brukner, Quantum-state preparation with universal gate decompositions, Physical Review A **83**, 032302 (2011).

[13] E. Malvetti, R. Iten, and R. Colbeck, Quantum circuits for sparse isometries, Quantum **5**, 412 (2021).

[14] I. F. Araujo, D. K. Park, F. Petruccione, and A. J. da Silva, A divide-and-conquer algorithm for quantum state preparation, Scientific Reports **11**, 6329 (2021).

[15] I. F. Araujo, D. K. Park, T. B. Ludermir, W. R. Oliveira, F. Petruccione, and A. J. da Silva, Configurable sublinear circuits for quantum state preparation, arXiv preprint arXiv:2108.10182 (2021).

[16] X.-M. Zhang, M.-H. Yung, and X. Yuan, Low-depth quantum state preparation, Phys. Rev. Research **3**, 043200 (2021).

[17] D. K. Park, F. Petruccione, and J.-K. K. Rhee, Circuit-based quantum random access memory for classical data, Scientific Reports **9**, 3949 (2019).

[18] C. Zoufal, A. Lucchi, and S. Woerner, Quantum generative adversarial networks for learning and loading random distributions, npj Quantum Information **5**, 103 (2019).

[19] K. Nakaji, S. Uno, Y. Suzuki, R. Raymond, T. Onodera, T. Tanaka, H. Tezuka, N. Mitsuda, and N. Yamamoto, Approximate amplitude encoding in shallow parameterized quantum circuits and its application to financial market indicator (2021).

[20] G. Marin-Sanchez, J. Gonzalez-Conde, and M. Sanz, Quantum algorithms for approximate function loading (2021).

[21] F. Mozafari, H. Riener, M. Soeken, and G. De Micheli, Efficient boolean methods for preparing uniform quantum states, IEEE Transactions on Quantum Engineering **2**, 1 (2021).

[22] C. E. Mora and H. J. Briegel, Algorithmic complexity of quantum states, International Journal of Quantum Information **04**, 715 (2006).

[23] C. E. Mora and H. J. Briegel, Algorithmic complexity and entanglement of quantum states, Phys. Rev. Lett. **95**, 200503 (2005).

[24] R. Raussendorf and H. J. Briegel, A one-way quantum computer, Phys. Rev. Lett. **86**, 5188 (2001).

[25] J. T. Barreiro, P. Schindler, O. Gühne, T. Monz, M. Chwalla, C. F. Roos, M. Hennrich, and R. Blatt, Experimental multiparticle entanglement dynamics induced by decoherence, Nature Physics **6**, 943 (2010).

[26] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, Phys. Rev. A **54**, 1098 (1996).

[27] A. Steane, Multiple-particle interference and quantum error correction, Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences **452**, 2551 (1996).

[28] R. Cleve, D. Gottesman, and H.-K. Lo, How to share a quantum secret, Phys. Rev. Lett. **83**, 648 (1999).

[29] W. Dür and J. I. Cirac, Multiparty teleportation, Journal of Modern Optics **47**, 247 (2000).

[30] V. V. Shende, S. S. Bullock, and I. L. Markov, Synthesis of quantum-logic circuits, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **25**, 1000 (2006).

[31] R. Iten, R. Colbeck, I. Kukuljan, J. Home, and M. Christandl, Quantum circuits for isometries, Physical Review A **93**, 032318 (2016).

[32] I. F. Araujo, C. Blank, A. da Silva, I. Cesar, and L. Silva, Quantum computing library (qclib), `https://github.com/qclib/qclib` (2022).

[33] G. Aleksandrowicz, T. Alexander, P. Barkoutsos, L. Bello, *et al.*, Qiskit: An Open-source Framework for Quantum Computing (2019).

[34] C. Blank, D. K. Park, and F. Petruccione, Quantum-enhanced analysis of discrete stochastic processes, npj Quantum Information **7**, 1 (2021).

[35] P. Teng, Accurate calculation of the geometric measure of entanglement for multipartite quantum states, Quantum Information Processing **16**, 181 (2017), 1609.02076.

[36] K. Mehlhorn and P. Sanders, *Algorithms and data structures: the basic toolbox* (Springer, Berlin, 2008).

[37] S. S. Skiena, *The algorithm design manual*, 2nd ed. (Springer, London, 2008) oCLC: ocn228582051.

[38] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, An analysis of approximations for maximizing submodular set functions—i, Mathematical Programming **14**, 265 (1978).

[39] T. H. Cormen, ed., *Introduction to algorithms*, 3rd ed. (MIT Press, Cambridge, Mass, 2009) oCLC: ocn311310321.

[40] A. Bouland, B. Fefferman, C. Nirkhe, and U. Vazirani, On the complexity and verification of quantum random circuit sampling, Nature Physics **15**, 159 (2019), 1803.04402.

[41] A. Shimony, Degree of Entanglement, Annals of the New York Academy of Sciences **755**, 675 (1995).

[42] H. Barnum and N. Linden, Monotones and invariants for multi-particle quantum states, Journal of Physics A: Mathematical and General **34**, 6787 (2001), quant-ph/0103155.

[43] T.-C. Wei and P. M. Goldbart, Geometric measure of entanglement and applications to bipartite and multipartite quantum states, Physical Review A **68**, 042307 (2003), quant-ph/0307219.

[44] L. Grasedyck, Hierarchical singular value decomposition of tensors, SIAM J. Matrix Anal. Appl. **31**, 2029–2054 (2010).

[45] S. Chakraborty, Generating discrete analogues of continuous probability distributions-a survey of methods and constructions, Journal of Statistical Distributions and Applications **2**, 6 (2015).

[46] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial nets, Advances in neural information processing systems **27** (2014).

[47] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature **567**, 209 (2019).

[48] M. Möttönen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa, Transformation of quantum states using uniformly controlled rotations, Quantum Information & Computation **5**, 467 (2005).

[49] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, Circuit-centric quantum classifiers, Phys. Rev. A **101**, 032308 (2020).

[50] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature **549**, 242 (2017).

[51] N. Moll, P. Barkoutsos, L. S. Bishop, J. M. Chow, A. Cross, D. J. Egger, S. Filipp, A. Fuhrer, J. M. Gambetta, M. Ganzhorn, A. Kandala, A. Mezzacapo, P. Müller, W. Riess, G. Salis, J. Smolin, I. Tavernelli, and K. Temme, Quantum optimization using variational algorithms on near-term quantum devices, Quantum Science and Technology **3**, 030503 (2018).

[52] E. Alpaydin and C. Kaynak, Cascading classifiers, Kybernetika **34**, 369 (1998).

[53] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, The theory of variational hybrid quantum-classical algorithms, New Journal of Physics **18**, 023023 (2016).

[54] T. Albash and D. A. Lidar, Adiabatic quantum computation, Reviews of Modern Physics **90**, 015002 (2018), 1611.04471.

[55] L. Veis and J. Pittner, Adiabatic state preparation study of methylene, The Journal of Chemical Physics **140**, 214111 (2014), 1401.3186.

APÊNDICE D – APPROXIMATED QUANTUM-STATE PREPARATION WITH ENTANGLEMENT DEPENDENT COMPLEXITY – SUPPLEMENTARY INFORMATION

Approximated quantum-state preparation with entanglement dependent complexity – Supplementary Information

Israel F. Araujo*[1], Carsten Blank†[2], Ismael Cesar‡[1], and Adenilton J. da Silva[1]

[1]Centro de Informática, Universidade Federal de Pernambuco, 50740-560, Recife, Pernambuco, Brazil
[2]data cybernetics, 86899, Landsberg am Lech, Germany

June 2022

1 Low-rank state preparation

1.1 Algorithm

To achieve low-rank approximation, the corresponding input parameter r is introduced to Algorithm 1 and can assume values between 1 and 2^{n_A}. The following conditional statement is introduced between lines 2 and 3: [If $r <$ rank then rank $= r$]. Setting $r = 1$ completely disentangles subsystems $\mathcal{H}_A$ and $\mathcal{H}_B$, which implies a quadratic reduction in the number of CNOT gates in the circuit to prepare a state when $n_A = \lfloor n/2 \rfloor$. Additionally, the circuit depth is also reduced.

SUPPLEMENTARY Algorithm 1: Low-rank state preparation algorithm

input : A state vector of 2^n amplitudes.
input : A partition of n_A qubits ($1 \leq n_A \leq \lfloor n/2 \rfloor$).
output: Encoding quantum circuit.

1 Decompose the state vector using *SVD* and partition (store unitaries u and vh, and vector s)
2 Set rank equal to the number of non-zero elements of s (rank = count($s \neq 0$))
3 Set Schmidt measure $m = \lceil \log_2(\text{rank}) \rceil$
4 If $m < n_A$, set the number of columns of u, vh^T, and s to 2^m, otherwise continue
5 Create a quantum circuit with n qubits ($n = \log_2(\text{length(state vector)})$)
6 If $m > 0$, encode the normalized vector s on the first m qubits $q \in$ partition using an amplitude encoding state preparation algorithm. This step is named *Phase 1*
7 Perform m *CNOT* gates between control qubit $q_c \in$ partition and target $q_t \notin$ partition. This step is named *Phase 2*
8 Encode u on qubits $q \in$ partition using an algorithm according to its dimension (vector, isometry, and unitary). This step is named *Phase 3*
9 Encode vh^T on qubits $q \notin$ partition using an algorithm according to its dimension (vector, isometry or unitary). This step is named *Phase 4*
10 Output the encoding quantum circuit

*ifa@cin.ufpe.br
†blank@data-cybernetics.com
‡icsa@cin.ufpe.br

2 Bounded Approximation Algorithm

2.1 Pseudocode

Pseudocode 1 constructs the approximation search tree as described in the main document. Pseudocode 2 employs Pseudocode 1 and expresses the bounded approximation algorithm (BAA). Pseudocode 3 builds quantum circuits using the BAA approach for encoding a complex input vector into the amplitudes of a quantum state.

Lines 3 of Pseudocode 1 generates all combinations of state partitions from size 1 to $\lfloor n/2 \rfloor$. Line 5 separates the indicated partition and creates two new states with a fidelity loss relative to the original state. Line 7 configures the recurrence so that, if the partition's total fidelity loss is less or equal than the maximal acceptable fidelity loss, a new child node is created. Compared to the previous node, the new one has the original state replaced by the two separated states.

Line 2 of Pseudocode 2 employs the geometric measure to estimate the maximum possible fidelity loss. If it is less than the maximal acceptable fidelity loss, the search routine is avoided and the product state used.

Line 3 of Pseudocode 3 initializes each of the state vectors associated to the selected node. Any initialization algorithm can be used, including the LRSP described in Section 1. The qubits used in each initialization must follow the previous BAA partitioning. Line 9 of Pseudocode 2 CNOT count must take into account the algorithm choice.

SUPPLEMENTARY Pseudocode 1: Generate a search tree.

```
1  build_search_tree (node, max_fidelity_loss):
       input : Tree node (node) to apply the next search step.
       input : The maximal acceptable fidelity loss (max_fidelity_loss) for the approximation.
       // Ignore single-qubit states.
2      for s ← 0 to length(node.states) − 1 do
           // Combinations taken k ∈ [1, ⌊n/2⌋] at a time without repetition.
           // Combinations equal to the complement of previous ones can be
           // omitted, as they are redundant.
3          partitions = combinations(node.states[s], k ← 1 to ⌊n/2⌋ )
4          for p ← 0 to length(partitions) − 1 do
               // Computes the two states and the fidelity loss after
               // disentangling partitions[p].
5              subsystem1, subsystem2, fidelity_loss = separate_partition(node.states[s], partitions[p])
6              total_fidelity_loss = 1 - (1 - fidelity_loss) × (node.total_fidelity_loss)
7              if total_fidelity_loss ≤ max_fidelity_loss then
                   // Recursion should not continue for this branch if
                   // total_fidelity_loss has reached max_fidelity_loss.
8                  new_node.total_fidelity_loss = total_fidelity_loss
9                  new_node.states = node.states
10                 new_node.states.remove(s)
11                 new_node.states.add(subsystem1)
12                 new_node.states.add(subsystem2)
13                 node.children.add(new_node)
                   // Call build_search_tree recurrently for each new node.
14                 build_search_tree (new_node, max_fidelity_loss)
```

SUPPLEMENTARY Pseudocode 2: Find the node that will be used to perform the state preparation.

```
baa (state, max_fidelity_loss):
    input  : A state vector (state) to be approximate.
    input  : The maximal acceptable fidelity loss (max_fidelity_loss) for the approximation.
    output: Node with the best approximation.
    // Estimate the maximal fidelity loss.
    entanglement = geometric_measure(state)
    if entanglement ≤ max_fidelity_loss then
        node.vectors = product_state(state)
        return node
    // Start with the complete state.
    root_node.vectors = state
    build_search_tree (root_node, max_fidelity_loss)
    // Leaf nodes are the ones closest to max_fidelity_loss for each branch.
    leaves = list_leaves(root_node)
    // Search the node with the greatest reduction in the number of CNOTs.
    // The CNOT count depends on the algorithm chosen to encode the states.
    best_node = search_best(leaves)
    return best_node
```

SUPPLEMENTARY Pseudocode 3: Construct a circuit that performs an approximate state preparation for the input vector *state*.

```
input  : A state vector (state) to be encoded.
input  : The maximal acceptable fidelity loss (max_fidelity_loss) for the baa approximation.
output: Circuit with the encoded quantum state.
node = baa (state, max_fidelity_loss)
for s ← 0 to length(node.states) − 1 do
    // Use any amplitude encoding algorithm.
    circuit.initialize(node.states[s])
return circuit
```

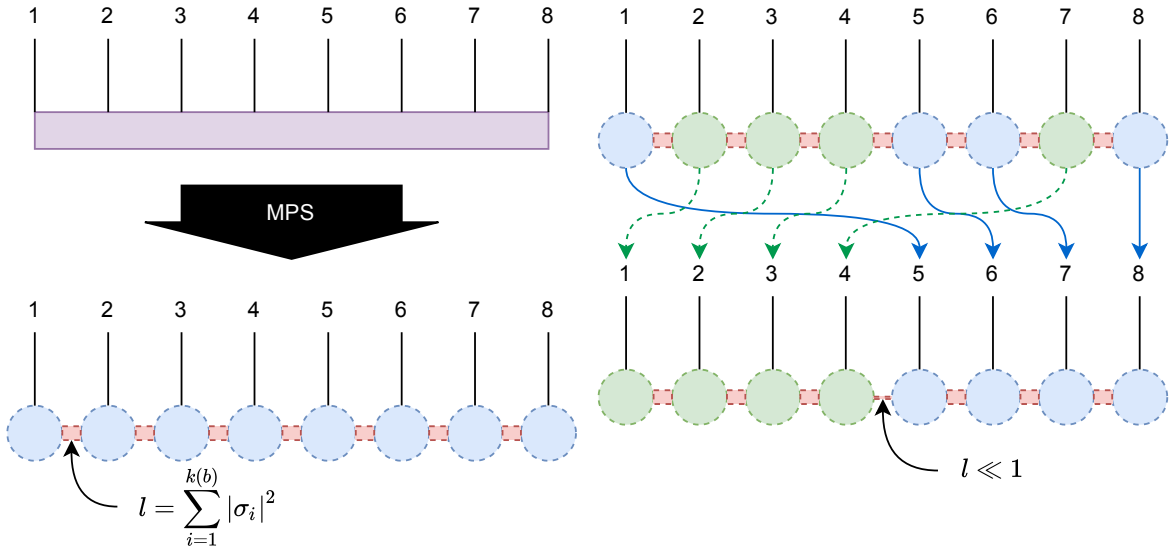
APÊNDICE E – EMARANHAMENTO E A PREPARAÇÃO DE ESTADOS QUÂNTICOS

O emaranhamento é um dos recursos quânticos que permitem o desenvolvimento de algoritmos e protocolos mais eficientes. Sua relação com a complexidade algorítmica de um estado quântico já foi demonstrada em trabalhos anteriores (MORA; BRIEGEL, 2006; MORA; BRIEGEL, 2005) e desempenha um papel fundamental em muitas aplicações quânticas, tais como comunicação quântica, correção de erros quânticos e compartilhamento de segredos quânticos (RAUSSENDORF; BRIEGEL, 2001; BARREIRO et al., 2010; CALDERBANK; SHOR, 1996; STEANE, 1996; CLEVE; GOTTESMAN; LO, 1999; DüR; CIRAC, 2000).

Para entender o significado do emaranhamento na computação quântica, um dos resultados seminais da física da matéria condensada é que sistemas de spin de baixo emaranhamento podem ser computados classicamente de forma eficiente. Como o modelo da computação quântica de circuitos pode ser representado por tais sistemas de spin, não se pode esperar vantagens quânticas quando o baixo emaranhamento está presente. Portanto, deve ser possível inicializar eficientemente um estado quântico com baixo emaranhamento. De fato, para a inicialização de dados clássicos em um estado quântico, deseja-se estar no regime de sistemas de spin de baixo emaranhamento e, após finalizar a preparação do estado, ter um aumento de emaranhamento (exponencial no número de qubits) aplicando uma evolução unitária para alcançar a vantagem quântica. O argumento é que este caminho poderia levar à solução do problema do carregamento de informações clássicas em um sistema quântico.

O trabalho Araujo, Blank e Silva (2022) (Seção 4.3 e Apêndice C) apresenta um algoritmo para preparação de estados cuja complexidade é uma função do emaranhamento, não apenas do número de qubits. Nele argumentamos ser necessário um quadro completo de todos os emaranhamentos bipartidos em sistemas de spin unidimensionais para descrever a estrutura topológica dos dados. De fato, deve-se reverter o problema e encontrar a disposição do sistema 1D de spin de tal forma que a codificação dos dados clássicos exiba o máximo possível de interações locais. Isto levará, então, a um baixo emaranhamento e, consequentemente, a uma redução do custo para a preparação do estado quântico. Como esta reordenação de spin-qubits é um mapeamento um-para-um (uma troca) no espaço Hilbert, essa operação, com algum cuidado, não afeta o algoritmo principal. A Figura 32 mostra a ideia principal em notação tensorial (ORúS, 2014), onde conexões vermelhas espessas significam emaranhamento mais

Figura 32 – Abordagem esquemática. a) Um vetor de estado quântico em notação de rede tensorial é convertido em um MPS através da aplicação de uma decomposição de Schmidt entre qubit 1 e qubits 2, 3, 4, 6, 7, 8, e depois qubit 2 vs. 3, 4, 5, 6, 7, 8 e assim por diante. A chamada dimensão de ligação é escolhida durante esse procedimento. A largura da barra de ligação significa a dimensão efetiva da ligação. b) Reorganizando as qubits é possível encontrar uma bipartição tal que cada partição tenha apenas uma pequena dimensão de ligação, o que significa que estas partições não estão fortemente emaranhadas.



(a) (b)

Fonte: Elaborada pelo autor (2022)

forte, e conexões finas a quase ausência de emaranhamento.

O mecanismo principal do algoritmo em Araujo, Blank e Silva (2022) explora o nível de emaranhamento do vetor de entrada, e eleva a capacidade de preparar uma aproximação de baixo posto (low-rank) (ECKART; YOUNG, 1936) de um estado quântico, usando a decomposição de Schmidt. O algoritmo LRSP é usado como um veículo algorítmico para conduzir a preparação de estado low-rank às operações quânticas concretas. Um método muito similar ao acima, por Gundlapalli e Lee (2021), explora a estrutura de ligação do *Matrix Product State* (MPS) (PEREZ-GARCIA et al., 2006) para projetar um circuito quântico que prepara um estado arbitrário de maneira exata ou aproximada. O método MPS é bem conhecido na comunidade da Física, portanto, queremos destacá-lo neste ponto.

O MPS também é chamado de *train-tensor formulation* que tem esse nome dada a representação

$$|\psi\rangle = \sum_{i_1,\ldots,i_n} A^{[1]}_{i_1} A^{[2]}_{i_2} \cdots A^{[n-1]}_{i_{n-1}} A^{[n]}_{i_n} |i_1 i_2 \cdots i_n\rangle$$

onde temos os n sites em uma grade 1D de spin-quits, e $A^{[m]}_{i_m} \in \mathbb{C}^{D_m \times D_{m+1}}$ com $D_1 = D_n = 1$ na chamada formulação *open boundary condition*. O D_m é chamado de dimensão da ligação no site m ao $m+1$. Uma aproximação do estado pode ser feita ao permitir apenas uma

dada dimensão de ligação máxima no site específico m. O MPS pode ser facilmente calculado através da aplicação da decomposição de Schmidt entre a bipartição $\{1\}/\{2,\ldots,n\}$, e depois $\{2\}/\{3,\ldots,n\}$, até que todos os qubits/sites tenham sido processados.

Usando os algoritmos de Araujo, Blank e Silva (2022) ou Gundlapalli e Lee (2021), é possível desemaranhar completamente subconjuntos de qubits enquanto o erro de fidelidade introduzido escala com os coeficientes de Schmidt que são descartados. Entretanto, uma desvantagem da abordagem de Gundlapalli e Lee (2021) reside no fato de que a abordagem MPS padrão, por convenção, utiliza uma direção pré-definida na descomposição de Schmidt. Isto por si só não é um inconveniente, mas ao fixar a base computacional do estado, a possibilidade de permitir bipartições arbitrárias – com qubits não necessariamente conectados – não é prontamente óbvia. O BAA, em conjunto com o LRSP, consegue reorganizar os sites e encontrar o sistema 1D de spin que exibe uma baixa dimensão de ligação em alguma fronteira. É então possível aproximar com um erro menor, potencialmente desemaranhando as bipartições. Pode-se então aplicar recursivamente esta abordagem nas duas partes separadamente; um algoritmo de busca com um dado erro de aproximação máxima, como o BAA, pode então encontrar a aproximação ótima de um estado quântico arbitrário.

Tendo em vista o tratamento para a aproximação do estado quântico, pode-se estar naturalmente inclinado a buscar a melhor aproximação de estado completamente separado. Esta é de fato uma medida para emaranhamento, a chamada medida geométrica de emaranhamento. Foi introduzida pela primeira vez por Shimony (SHIMONY, 1995) e refinada por Barnum & Linden (BARNUM; LINDEN, 2001). A definição para esta medida de emaranhamento é dada por

$$E_g(|\psi\rangle) = \min_{|\phi\rangle} \| \, |\psi\rangle - |\phi\rangle \, \| \tag{E.1}$$

onde a minimização está sobre todos os estados $|\phi\rangle$ que são estados completamente separáveis (TENG, 2017), ou seja, $|\phi\rangle = \otimes_{l=1}^{n} \left|\phi^l\right\rangle$ com cada $\left|\phi^l\right\rangle$ sendo um estado local. Wei e Goldbart (2003) mostrou que este valor é uma medida útil de emaranhamento.

A medida geométrica do emaranhamento está relacionada ao algoritmo BAA da seguinte forma. Teoricamente, a maior perda de fidelidade que qualquer estado quântico pode experimentar sob o BAA nunca deve ser superior a ela. Portanto, a medida geométrica indica quão ruim uma aproximação pode ficar. Isto mostra que os estados de baixo emaranhamento podem ser aproximados como um estado completamente separável com pouca perda de fidelidade.

Naturalmente, estamos interessados em calcular o estado separável que maximiza a fideli-

dade na tentativa de calcular a medida geométrica do emaranhamento. A primeira tentativa desse tipo publicada na literatura científica calcula a medida com a ajuda de um problema de valor próprio não linear (WEI; GOLDBART, 2003). Algoritmos mais eficientes foram desenvolvidos nesse ínterim. Como foi apontado por Teng (TENG, 2017), pode-se usar a decomposição de posto tensorial e a decomposição de Tucker (ENRíQUEZ; PUCHAłA; ŻYCZKOWSKI, 2015).

O cálculo da medida geométrica do emaranhamento pode ser feito de forma eficiente e com alta probabilidade aplicando a decomposição de Tucker de um estado quântico $|\psi\rangle$, interpretando-o como um tensor com cada modo sendo bidimensional e de rank um. O chamado *core* (KOLDA; BADER, 2009) é então simplesmente um número complexo λ e a medida geométrica de emaranhamento é $E(\psi) = 1 - |\lambda|^2$ (TENG, 2017). Cada fator é um vetor, e eles constituem os estados de um qubit que podem ser preparados juntos como um estado separável. O estado criado de tal forma é então chamado de $|\psi'\rangle$, e seu *overlap* com o estado original é $\langle\psi|\psi'\rangle = \lambda$.

O caso particular onde as partições do BAA são divididas ao meio e apenas um ramo selecionado é equivalente ao *Hierarchical Tucker Format* (HTF) com uma árvore de dimensão binária (chamada de Árvore de Dimensão Canônica) (GRASEDYCK, 2010). O HTF pode ser usado para estimar a máxima perda de fidelidade possível com um erro menor ou igual a $\sqrt{2n-3}\|A - A^{\text{best}}\|$, onde A^{best} é a melhor aproximação do tensor A, e n é a dimensionalidade do problema. O número de bipartições realizadas para estimar a perda máxima de fidelidade é linear no número de qubits. Como BAA realiza o SVD a partir de tensores já truncados, o custo do HTF é dado pelo *Lemma* 26 de Grasedyck (2010).

O objetivo desta seção foi conectar a física da matéria condensada com a computação quântica e também conectar a representação hamiltoniana de baixo emaranhamento/interação local com a preparação de estados de baixa complexidade, enquanto argumenta que os dados clássicos possuem um grau de liberdade na ordenação dos spin-quits que a física da matéria condensada não tem. Para ir ainda mais longe, a codificação de dados clássicos que devem ser usados no processamento de informações quânticas podem ser mapeados nas propriedades de um hamiltoniano, cujas interações dominantes podem ser locais ou não-locais.

Esta seção deixa como trabalho futuro verificar se a física da matéria condensada possui ferramentas para encontrar a melhor ordenação dos spin-quits na grade 1D, de modo que a localidade seja aumentada. Atualmente, a melhor estratégia é uma busca exaustiva, com custo exponencial no número de qubits.