Decomposição em Produto de Kronecker para Compilação Quântica

Thiago Melo D. Azevedo[1], **Gabriel M. Langeloh**[2], **Samuraí Brito**[2], **Adenilton J. da Silva**[1]

[1]Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Recife – PE – Brasil

[2]Instituto de Ciência e Tecnologia Itaú - ICTi
São Paulo – SP – Brasil

tmda@cin.ufpe.br, gabriel.langeloh@itau-unibanco.com.br,
samurai.brito@itau-unibanco.com.br, ajsilva@cin.ufpe.br

Abstract. *In this work, we introduce a method for Kronecker Product Decomposition of unitary matrices that can decompose a separable unitary matrix in $O(N^2 \log N)$ time without requiring knowledge about the matrix's dimensions or its qubit partitions. The proposal represents a significant improvement over the $O(N^3)$ time of the algorithms in the literature. We apply the method to the compilation of quantum unitaries, resulting in an empirical reduction in the circuit gate count and a significant improvement in compilation times. We validate the results with computational experiments to verify the reduced compilation time and CNOT count when implementing unitary gates.*

Resumo. *Neste trabalho, introduzimos um método para a Decomposição de Produto de Kronecker de matrizes unitárias, capaz de decompor uma matriz unitária separável em tempo $O(N^2 \log N)$, sem exigir conhecimento prévio sobre as dimensões da matriz ou suas partições de qubits. A proposta representa uma melhoria significativa em relação ao tempo $O(N^3)$ dos algoritmos presentes na literatura. Aplicamos o método à compilação de unitárias quânticas, resultando em uma redução empírica na contagem de portas do circuito e em uma melhoria significativa nos tempos de compilação. Validamos os resultados com experimentos computacionais para verificar reduções no tempo de compilação e na contagem de CNOT ao implementar portas unitárias.*

1. Introdução

A computação quântica representa uma mudança de paradigma computacional, com potencial para resolver problemas em áreas como simulação, otimização e aprendizado de máquina. Computadores quânticos utilizam fenômenos quânticos, como sobreposição, interferência e emaranhamento, para proporcionar vantagens computacionais na resolução de problemas específicos. No entanto, o hardware atual de escala intermediária e ruidoso (NISQ, do inglês Noisy Intermediate-Scale Quantum) [Preskill 2018] é limitado pelo baixo número de qubits, pelas altas taxas de erro de portas e pelos curtos tempos de coerência. Para tornar a computação quântica viável, é imperativo desenvolver técnicas mais eficientes para implementar operações e algoritmos quânticos, visando reduzir a complexidade dos circuitos.

Melhorar a eficiência da síntese de portas quânticas [Shende et al. 2006, Barenco et al. 1995, Iten et al. 2016] é, portanto, uma tarefa essencial para desenvolver implementações mais viáveis de operações e algoritmos quânticos. Uma possível linha de pesquisa é o desenvolvimento de métodos mais eficientes para síntese de portas unitárias genéricas. Um operador U é considerado unitário se satisfaz a relação $U^\dagger U = UU^\dagger = I$. Essa classe de portas, é um bloco fundamental na construção de circuitos quânticos e está presente em diversos algoritmos quânticos, como na estimativa de fase quântica e em algoritmos de busca, fatoração e simulação [Nielsen and Chuang 2010]. Neste trabalho, desenvolvemos um passo de otimização para implementar eficientemente a classe de matrizes unitárias em dispositivos quânticos.

Um operador quântico U é separável, quando ele pode ser escrito como um produto de Kronecker de duas matrizes menores, $U = A \otimes B$. As bibliotecas de computação quântica atuais que suportam a compilação de matrizes em circuitos quânticos não são capazes de detectar a separabilidade de portas quânticas. Um operador quântico ser separável entre dois subespaços implica que na sua representação em circuitos, não haverá portas que emaranhem qubits entre esses subespaços. Por exemplo, o resultado esperado da compilação de $H^{\otimes 3}$ é um circuito com três portas Hadamard e profundidade igual a 1, em hardware que tenha suporte nativo a essa operação ou a portas de 1 qubit genéricas. No entanto, quando criamos o circuito para este operador com a versão do Qiskit 2.2.3 [Javadi-Abhari et al. 2024], obtemos um circuito com 30 operações de um qubit, 15 portas CNOT e profundidade igual a 30. A mesma limitação ocorre em outras bibliotecas quânticas, como Pennylane [Bergholm et al. 2018] e Cirq [Developers 2025].

Para uma compilação eficiente de operadores unitários, desejamos decompor uma matriz unitária $N \times N$, onde $N = 2^n$, como o produto de Kronecker de matrizes unitárias $U = A \otimes B$. A unitária A possui dimensões $N_A \times N_A$ e B possui dimensões $N_B \times N_B$, onde $N_A = 2^{n_a}$, $N_B = 2^{n_b}$ e $N = N_A N_B$.

A Ref. [Van Loan and Pitsianis 1993] apresentou um algoritmo para calcular uma decomposição de produto de Kronecker (KPD) de uma matriz $N \times N$ utilizando a decomposição em valores singulares (*Singular value decomposition, SVD*) com um custo computacional de $O(N^3)$. A SVD é utilizada em diversos trabalhos relacionados à decomposição de produtos de Kronecker [Batselier and Wong 2017, Paleologu et al. 2018, Wang et al. 2021, Fahrbach et al. 2022, Cai et al. 2022, Kamm and Nagy 2000, S. et al. 2024]. Posteriormente, a Ref. [Wu 2023] propôs outro método para determinar a decomposição de produto de Kronecker de uma matriz com complexidade $O(N^2)$.

No entanto, os algoritmos atuais na literatura exigem conhecimento prévio das dimensões de A e B. Essa restrição resulta em um desafio combinatório, pois, para considerar todas as divisões das dimensões de A e B, seria necessário repetir o processo $n = \log N$ vezes. Além disso, para a compilação de operadores quânticos, precisamos considerar as múltiplas possibilidades de partições de qubits (por exemplo, se A atua nos qubits $\{0, 2\}$ e B nos qubits $\{1, 3\}$). Isso exige a busca por $2^n = N$ atribuições de qubits possíveis. Aplicar ingenuamente o método $O(N^2)$ da Ref. [Wu 2023] a todas as N partições resulta em um custo total de $O(N^3)$.

Neste trabalho, utilizamos uma sub-rotina da Ref. [de Carvalho et al. 2024] que

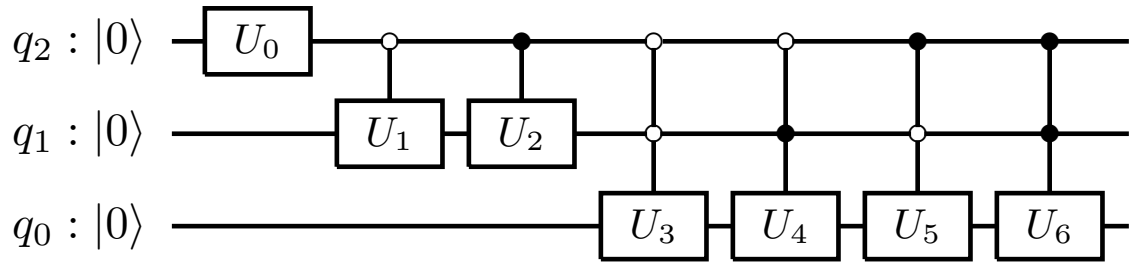

Figura 1. Um circuito para Preparação de Estado Quântico [de Carvalho et al. 2024, Bergholm et al. 2005].

identifica os componentes separáveis de um dado vetor de estado quântico para determinar os componentes separáveis da matriz unitária que representa um operador quântico. Aplicamos a sub-rotina para cada coluna e, em seguida, executamos um algoritmo Union-Find [Tarjan 1975] para obter uma lista de componentes separáveis compatível com toda a matriz.

Uma vez conhecidos os componentes separáveis, torna-se possível obter a decomposição em produto de Kronecker da matriz unitária em $O(N^2)$. Adicionalmente, a execução da identificação de componentes separáveis da Ref. [de Carvalho et al. 2024] para todas as N colunas possui um custo computacional de $O(N^2 \log N)$, e a aplicação do algoritmo Union-Find possui um custo próximo de $O(N \log N)$ [Tarjan 1975, Cormen et al. 2022]. Dessa forma, reduzimos o custo computacional necessário para sua decomposição como produto de Kronecker para $O(N^2 \log N)$, o que é significativamente menor do que o custo $O(N^3)$ da Ref. [Wu 2023], quando se contabiliza a busca combinatória de partições de qubits.

O método proposto é aplicado à compilação de portas quânticas, pois o utilizamos para decompor operadores separáveis como produtos de Kronecker de operadores menores, cada um a ser compilado individualmente. Enquanto a síntese de um operador geral de n qubits U possui uma contagem de portas exponencial de $O(4^n)$ [Shende et al. 2006], nosso método o decompõe em $U = U_1 \otimes \cdots \otimes U_k$. A síntese desses k operadores menores em paralelo possui um custo de portas de $O(4^{\max n_i})$, o que representa uma redução significativa na contagem de portas e no tempo de compilação quando o operador unitário for separável.

O resto deste trabalho é organizado da seguinte forma: Seção 2 revisa um método de simplificação de multiplexadores para preparação de estados, o que nos permite obter as partições de qubit do estado quântico separável; Seção 3 descreve um novo método para calcular uma decomposição em produto de Kronecker, sem necessitar do conhecimento sobre as dimensões das submatrizes ou as suas partições de qubits; Seção 4 aplica o método de decomposição em produto de Kronecker como um passe de otimização na compilação de operadores unitários, apresentando também resultados experimentais.

2. Simplificação de multiplexadores para preparação de estados.

A preparação de estados quânticos é um processo que requer a codificação dos dados de um determinado estado $|\psi\rangle$ em um circuito quântico, onde os qubits estão previamente em um estado fixado (por exemplo, o estado $|0\rangle$). O estado $|\psi\rangle$ pode ser escrito como

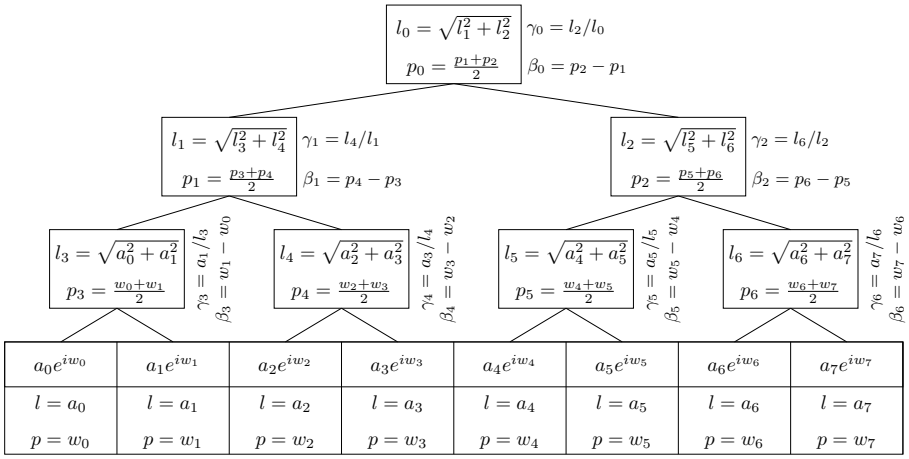

Figura 2. Representação em árvore binária do protocolo de preparação de estado da Ref. [Bergholm et al. 2005].

$$|\psi\rangle = \sum_{k=0}^{2^n-1} a_k e^{iw_k} |k\rangle . \tag{1}$$

Uma das técnicas para a preparação de estados consiste na aplicação de um multiplexador para transformar o estado inicial $|a\rangle_n$ em $|a'\rangle_{n-1} |0\rangle$ [Bergholm et al. 2005], e então aplicar recursivamente uma sequência de multiplexadores para transformar $|a\rangle_n$ em $|0\rangle_n$. Por fim, invertemos o circuito para obter a preparação do estado $|a\rangle_n$ a partir do estado inicial $|0\rangle_n$.

Um circuito para preparação de estado quântico pode ser visualizado na Fig. 1 e utiliza uma sequência de operadores uniformemente controlados $\{U_j\}$ que correspondem a rotações y e z [Bergholm et al. 2005, de Carvalho et al. 2024]. Definindo $\gamma_j = a_{2j+1}/\sqrt{a_{2j}^2 + a_{2j+1}^2}$, os operadores U_j podem ser escritos como:

$$U_j = R_y(-2\arcsin(\gamma_j))R_z(w_{2j} - w_{2j+1}). \tag{2}$$

A preparação de estado pode ser visualizada como uma árvore binária, onde as folhas da árvore representam as amplitudes $a_k e^{iw_k}$ e cada nível da árvore corresponde a uma camada de multiplexadores [de Carvalho et al. 2024], como pode ser visto na Figura 2. Cada nó interno possui um filho à esquerda (*left*) e um à direita (*right*), onde cada ramo no k-ésimo nível da árvore corresponde a uma das portas multicontroladas do multiplexador da camada. Os nós folha possuem os seguintes atributos: uma amplitude complexa *value*; um comprimento $l = |value|$; e uma fase p igual à fase de *value* e definida no intervalo $(-\pi/2, \pi/2]$. Enquanto isso, os nós internos possuem os atributos adicionais γ e β, que são os ângulos para a rotação y e z, respectivamente.

Na Ref. [de Carvalho et al. 2024], os autores propuseram um método para simplificar os multiplexadores envolvidos na preparação de estados, investigando o emaranhamento do estado a ser preparado. Eles representaram os operadores $U_j = R_y(-2\arcsin(\gamma_j))R_z(\beta_k)$ em uma árvore binária, como mostrado na Fig. 3, onde cada nível k da árvore inclui o conjunto de operadores do k-ésimo multiplexador. As setas

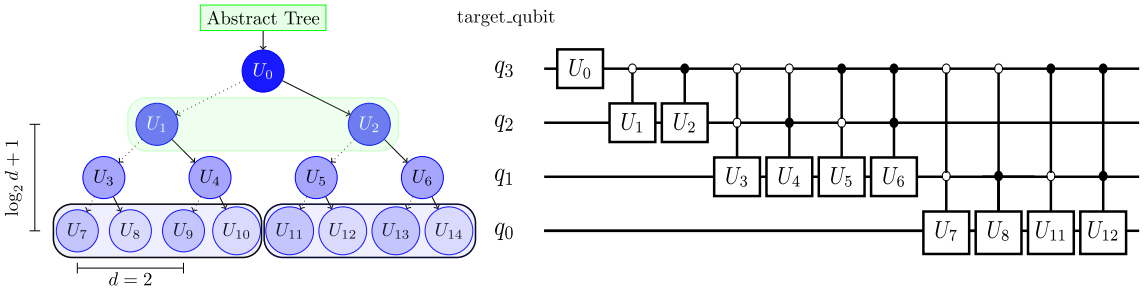

Figura 3. (Esquerda) Representação em árvore dos operadores utilizados nos multiplexadores para preparações de estados quânticos [de Carvalho et al. 2024]. Cada nível da árvore define um multiplexador que é armazenado em um vetor (array). Setas pontilhadas (sólidas) correspondem a controles abertos (fechados) para o operador no multiplexador. (Direita) Circuito de preparação de estado se $[U_7, U_8] = [U_9, U_{10}]$ **e** $[U_{11}, U_{12}] = [U_{13}, U_{14}]$ **[de Carvalho et al. 2024].**

pontilhadas (sólidas) correspondem aos controles abertos (fechados) das portas no multiplexador. Na figura, verificamos se as folhas das subárvores com raízes U_3 e U_4 são iguais, e também verificamos se as folhas das subárvores com raízes U_5 e U_6 são iguais. Em outras palavras, verificamos se $[U_7, U_8] = [U_9, U_{10}]$ e $[U_{11}, U_{12}] = [U_{13}, U_{14}]$ e, em caso afirmativo, podemos remover os controles no qubit q_2 do último multiplexador.

O método itera sobre o parâmetro d de 1 a $\log N$, e então verifica se o operador U_k é igual ao operador U_{k+d}. Se isso for verdadeiro, após as $N/2$ comparações, então os controles associados à altura d da árvore são redundantes, e o algoritmo simplifica o multiplexador ignorando esses controles. Ele então passa para o próximo nível, realizando outras $N/2$ comparações. Portanto, o custo total do algoritmo é $O(N \log N)$ [de Carvalho et al. 2024].

O método verifica se um qubit de controle do multiplexador é redundante sob uma perspectiva de lógica de circuito. Se for o caso, o multiplexador pode ser simplificado. Utilizamos esse método para identificar partições de qubits de estados separáveis: se não houver portas de emaranhamento entre dois subconjuntos de qubits no circuito que prepara o estado, então o estado não pode ter emaranhamento entre esses subconjuntos. Portanto, ao identificar os qubits de controle descartáveis, podemos determinar se o estado quântico dado possui componentes separáveis.

3. Método para decomposição em produto de Kronecker

Nesta seção, apresentamos um novo método para decompor um operador quântico U em um produto de Kronecker ($U = U_1 \otimes U_2 \otimes \cdots \otimes U_k$) sem conhecimento prévio de seus subespaços separáveis. Nossa abordagem utiliza as técnicas da Ref. [de Carvalho et al. 2024] para identificar eficientemente componentes separáveis e calcular a decomposição final. A restrição sobre o conhecimento prévio dos subespaços separáveis é necessária para a compilação quântica, pois o usuário fornece uma matriz como entrada e o compilador deve então detectar quais são os subespaços separáveis. Assim, o método não pode depender da informação sobre as dimensões de cada subespaço, nem de qual é a permutação da ordem dos qubits das partições separáveis. O método desenvolvido neste trabalho possui complexidade computacional de tempo $O(N^2 \log N)$.

Esta é uma redução significativa em relação ao tempo de $O(N^3)$ exigido por abordagens baseadas na Ref. [Wu 2023], considerando todas as combinações possíveis de qubits em cada componente.

Integramos este método como um passo de otimização no processo de compilação, aplicando-o antes de sintetizar o operador unitário em um circuito quântico. Para a síntese, utilizamos a decomposição quântica de Shannon [Shende et al. 2006] (No inglês, *Quantum Shannon Decomposition*, QSD), que aplica recursivamente uma decomposição cosseno-seno; porém, nosso método verifica primeiramente a separabilidade. Se o operador for separável, evitamos a QSD completa e simplificamos o problema em uma compilação de dois ou mais operadores menores e paralelos.

3.1. Determinando as componentes separáveis

Nosso método adapta a subrotina $O(N \log N)$ do algoritmo de preparação de estado UCGE [de Carvalho et al. 2024]. Enquanto a subrotina original identifica componentes emaranhados de um estado quântico, nós a aplicamos para determinar os componentes separáveis do operador unitário $N \times N$, U.

O procedimento começa analisando cada uma das N colunas de U. Tratamos cada coluna j como um vetor de estado, $|\psi_j\rangle = U|j\rangle$, que representa a ação do operador no j-ésimo estado da base computacional. Fornecemos cada um desses N vetores coluna para a subrotina de [de Carvalho et al. 2024]. Este passo resulta em N listas, $\{L_1, \ldots, L_N\}$, onde cada lista L_j descreve os componentes separáveis (isto é, as partições de qubits) do estado correspondente $|\psi_j\rangle$.

Em seguida, devemos agregar essas N listas para encontrar uma única decomposição compatível com todo o operador U. Uma condição necessária para que U seja separável em relação a uma dada partição de qubits é que cada coluna $|\psi_j\rangle$ deve ser separável em relação a essa mesma partição. Portanto, se qualquer lista L_j indicar que os qubits q_1 e q_2 estão emaranhados (isto é, no mesmo componente), eles também devem estar no mesmo componente na decomposição final de U.

Resolvemos eficientemente este problema de agregação modelando-o como um problema de conectividade de grafos. Começando com um grafo com $n = \log_2 N$ vértices, um para cada qubit, e sem arestas, iteramos por todas as N listas e, para cada lista L_j, adicionamos arestas entre todos os vértices (qubits) que aparecem juntos em uma componente não-separável. Então, tomamos as componentes conexas do grafo final como as componentes separáveis de todo o operador U. Calculamos essas componentes conexas utilizando uma estrutura de dados Union-Find [Cormen et al. 2022, Tarjan 1975].

A complexidade geral é dominada pelo passo de identificação das componentes de cada coluna, que executa a subrotina $O(N \log N)$ para cada uma das N colunas, resultando em um custo de $O(N^2 \log N)$. O passo de agregação, que realiza $O(N \log N)$ operações de união (para N listas com $n = \log N$ qubits), possui um custo de $O(N \log N \cdot \alpha(n))$, onde $\alpha(n)$ é a função de Ackermann inversa e pode ser considerada $O(1)$ [Tarjan 1975]. Isso é absorvido pelo custo de análise, tornando a complexidade de tempo total $O(N^2 \log N)$. Notamos que este algoritmo permite o término antecipado: se qualquer coluna (por exemplo, a primeira) for identificada como um único componente não-separável, podemos concluir imediatamente que U é não-separável e terminar em tempo $O(N \log N)$.

3.2. Computando as matrizes

Uma vez identificadas as componentes separáveis (partições de qubits), calculamos as matrizes correspondentes. Primeiro, decompomos U em $U = A \otimes B$, onde $A \in \mathbb{C}^{N_A \times N_A}$ e $B \in \mathbb{C}^{N_B \times N_B}$ são os operadores unitários nos subespaços previamente identificados.

Para usar uma extração simples, primeiro permutamos U de modo que os $n_A = \log N_A$ qubits correspondentes ao subsistema A sejam os qubits mais significativos. Esta operação requer uma permutação de linhas e colunas, que pode ser feita rearranjando os elementos da matriz, com um custo computacional de $O(N^2)$.

$$U' = A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1N_A}B \\ a_{21}B & a_{22}B & \cdots & a_{2N_A}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{N_A1}B & a_{N_A2}B & \cdots & a_{N_AN_A}B \end{bmatrix} \tag{3}$$

Com U' nesta forma, podemos extrair as matrizes unitárias A e B. O operador A é uma matriz $N_A \times N_A$, em que cada elemento a_{ij} escala a matriz B de dimensões $N_B \times N_B$ no bloco $U'(i,j)$, que é o (i,j)-ésimo bloco com dimensões $N_B \times N_B$ de U'. O Algoritmo 1 implementa a determinação das matrizes A e B. Em linhas gerais, o algoritmo determina B através do primeiro bloco não-nulo (de tamanho $N_b \times N_b$) das primeiras N_B colunas da matriz U, e então utiliza a matriz B encontrada para determinar os elementos a_{ij} de A. Ao determinar cada elemento a_{ij}, o algoritmo verifica se o a_{ij} encontrado satisfaz o produto de Kronecker $A \otimes B = U$, com o B obtido anteriormente. Por fim, o algoritmo normaliza as matrizes obtidas.

Para determinar a matriz B no Algoritmo 1, utilizamos a função $FindNonZeroElement(B, N)$ que procura na primeira coluna da matriz U um elemento não nulo. A função implementa uma busca em blocos de N_B elementos, utilizando um parâmetro $row \leq N$, de forma que, quando encontrar um bloco com elemento não nulo, atribui $B = U[row : row + N_b, 0 : N_B]$, que significa os elementos de U' da linha row até a linha $row + N_B - 1$ e das colunas 0 até $N_B - 1$. Além disso, a função determina k_r, que é a linha do elemento com maior valor absoluto de B.

Em seguida, o algoritmo 1 calcula cada a_{ij} usando esta referência. O algoritmo então realiza uma verificação $O(N^2)$ checando se $U'(i,j)_{kl} = a_{ij} \times B_{kl}$ é válido para todos os elementos. Finalmente, ele normaliza A e B para resolver a ambiguidade de fase inerente. Após obter A e B, a estratégia é aplicada recursivamente a B até que não possa mais ser decomposta.

O algoritmo tem complexidade $O(N_A^2 N_B^2) = O(N^2)$, pois realiza uma verificação de que a matriz possui a estrutura de um produto de Kronecker. Adicionando o custo da permutação, a complexidade computacional para a decomposição de Kronecker é $O(N^2)$. A decomposição adicional de B adicionará um custo computacional de $O(N_B^2)$, menor que o da primeira decomposição. O que significa que o custo de calcular a decomposição do produto de Kronecker de um operador quântico U em k suboperadores $U_j, \{j \in [1, \ldots, k]\}$ também é $O(N^2)$.

Portanto, o custo assintótico para a identificação dos componentes e cálculo das matrizes da decomposição em produto de Kronecker é $O(N^2 \log N)$, uma redução em

Algoritmo 1: Decomposição em Produto de Kronecker

```
    // Determinando a matriz B
1   B, k_r ← FindNonZeroElement(B, N)
2   for i = 0 to dim(A) − 1 do
3       for j = 0 to dim(A) − 1 do
            // Determinando cada elemento da matriz A
4           M ← U[i · dim(B) + k_r, j · dim(B)];
5           A[i, j] ← M/B[k_r, 0];
            // Verificando se o elemento encontrado
               satisfaz o produto de Kronecker
6           for k = 0 to dim(B) − 1 do
7               for l = 0 to dim(B) − 1 do
8                   M_ref ← U[i · dim(B) + k, j · dim(B) + l];
9                   M' ← A[i, j] × B[k, l];
10                  if |M_ref − M'| > 0 then
                        // Não satisfaz o produto de Kronecker,
                           a Decomposição falhou
11                      return U, None;
12  a_00 ← 1/‖A[:, 0]‖;
13  A ← a_00 · A;
14  B ← B/a_00;
15  return A, B;
```

relação ao custo de $O(N^3)$ de abordagens da literatura [Wu 2023], quando se levam em conta todas as combinações possíveis de qubits para cada componente. Dado que o custo computacional para implementar um operador unitário é exponencial no número de qubits [Shende et al. 2006], a decomposição da unitária como um produto de Kronecker pode reduzir seu tempo de compilação e contagem de portas, uma vez que implementamos cada suboperador separadamente.

4. Compilação de operadores unitários separáveis

Podemos utilizar o novo método de decomposição em produto de Kronecker como um passo de otimização na compilação de portas unitárias. No caso de uma unitária separável, em vez de realizar a síntese da unitária completa, realizamos a síntese de cada suboperador individualmente. Como a contagem de portas da Decomposição de Shannon Quântica [Shende et al. 2006] escala exponencialmente com o número de qubits, fatorar a unitária de n qubits em operadores menores que atuam em menos qubits pode proporcionar uma redução significativa na contagem de portas e no tempo de compilação para unitárias separáveis.

Nesta seção, realizamos experimentos computacionais para verificar a redução do custo de compilação para matrizes unitárias com componentes separáveis, utilizando o método de decomposição em produto de Kronecker apresentado no trabalho. Analisamos o tempo de compilação (incluindo pré-processamento clássico) e a contagem de CNOTs para matrizes unitárias geradas aleatoriamente, dada uma lista de componentes separáveis. Comparamos nosso método com a compilação unitária padrão do Qiskit

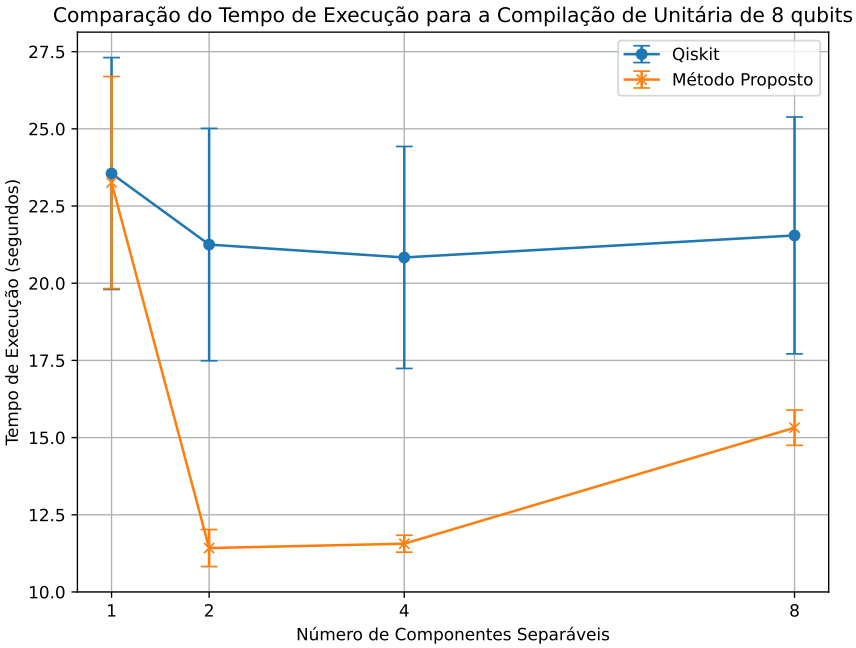

Figura 4. Tempo de compilação para unitárias aleatórias com 8 qubits. A unitária pode ter 1, 2, 4 ou 8 componentes separáveis, em que cada uma possui o mesmo número de qubits. Para cada número de componentes separáveis, foram geradas aleatoriamente 30 matrizes unitárias. Os gráficos mostram os valores médios, enquanto as barras de erro indicam o desvio padrão.

2.2.3 [Javadi-Abhari et al. 2024].

Os experimentos dessa seção foram realizados em Python, utilizando a biblioteca Qiskit 2.2.3 [Javadi-Abhari et al. 2024] para a construção do circuito após aplicarmos o método proposto neste artigo, que inclui a verificação de separabilidade e decomposição em produto de Kronecker do operador unitário. Já para a preparação de estados UCGE utilizamos a implementação disponível na biblioteca qclib [Araujo et al. 2023]. Os experimentos foram executados em um computador com uma CPU Ryzen 7 5700x, 32 GB de RAM e sistema operacional Windows 11.

Para os experimentos, analisamos casos em que um operador unitário de 8 qubits possui 1, 2, 4 ou 8 componentes separáveis. Geramos aleatoriamente (utilizando a biblioteca SciPy [Virtanen et al. 2020]) 30 matrizes unitárias para cada número possível de componentes e executamos a compilação desses operadores em um circuito quântico, medindo o tempo de compilação e o número de CNOTs do circuito compilado. A Fig. 4 mostra os tempos médios de compilação do circuito para uma matriz de 8 qubits, em função do número de componentes separáveis. O gráfico apresenta o desvio padrão como barras de erro. Observamos que nosso método pode reduzir o tempo de compilação quando o operador possui pelo menos 2 componentes separáveis, além de obter um tempo de compilação comparável ou menor quando o operador não é separável. Além disso, a Fig. 5 compara o custo de CNOT de ambos os métodos, demonstrando uma melhoria expressiva na contagem de portas proporcionada pelo nosso passo de otimização, quando a matriz possui mais de um componente. Essa redução é particularmente relevante dada a elevada taxa de erros das portas de 2 qubits, como a CNOT, nos hardwares quânticos atuais.

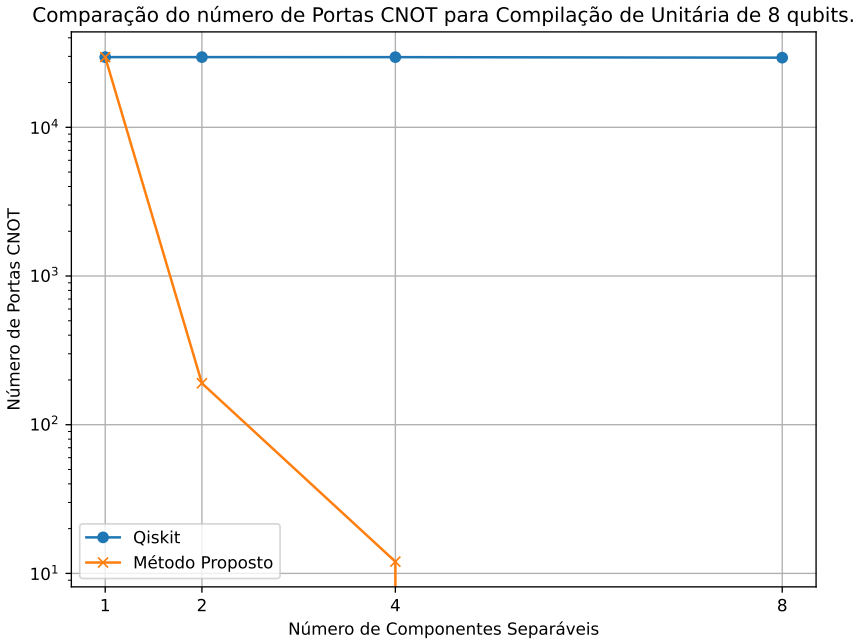

Figura 5. Contagem de CNOT para unitárias aleatórias com 8 qubits. A unitária pode ter 1, 2, 4 ou 8 componentes separáveis, em que cada uma possui o mesmo número de qubits. Para 8 componentes separáveis, o circuito produzido pelo método proposto não possui nenhuma CNOT.

5. Conclusão

Neste trabalho, propusemos um novo método para calcular a decomposição em produto de Kronecker de uma matriz unitária, sem qualquer conhecimento prévio sobre os componentes do operador. O método utiliza um procedimento da Ref. [de Carvalho et al. 2024] para identificar a lista de componentes do estado associado a cada coluna da matriz unitária. Em seguida, utilizamos um algoritmo de Union-Find [Tarjan 1975] para determinar a lista de componentes de toda a matriz unitária. A complexidade de tempo total do método é $O(N^2 \log N)$, composta por uma etapa de custo $O(N^2 \log N)$ para identificar os componentes e outra de custo $O(N^2)$ para calcular as matrizes finais. Essa complexidade é inferior ao tempo $O(N^3)$ exigido por algoritmos da literatura [Wu 2023] ao considerarem todas as partições de qubits possíveis do operador.

Aplicamos nosso algoritmo de decomposição em produto de Kronecker como uma etapa de otimização (passo de compilação) fundamental para a compilação de operadores unitários quânticos. Verificamos se o operador é separável e, em caso afirmativo, decompomos a matriz como produto de Kronecker e compilamos cada submatriz separadamente utilizando a decomposição quântica de Shannon [Shende et al. 2006], o que proporciona um ganho de velocidade (*speedup*) significativo na compilação de operadores separáveis. Confirmamos os resultados por meio de experimentos computacionais, em que comparamos o tempo de compilação e a contagem de portas CNOT nas sínteses de operadores unitários. No entanto, deve-se notar que o tamanho das matrizes é limitado pela quantidade de memória disponível no computador clássico que realiza a compilação do circuito. Dessa forma, é inviável decompor uma matriz unitária arbitrariamente grande. Adicionalmente, o método proposto fundamenta-se em uma condição necessária, mas não suficiente, de separabilidade.

Pesquisas futuras podem ser direcionadas para uma versão aproximada da decomposição em produto de Kronecker. Também poderiam ser estudadas aplicações em outras implementações de operadores quânticos, como isometrias ou unitárias esparsas.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001. Agradecemos ao ICTi, Instituto de Ciência e Tecnologia Itaú, e ao Instituto Nacional de Computação Quântica Aplicada por financiarem esse trabalho. Os autores agradecem a José Victor Soares Scursulim e Victor Leme Beltran pelas discussões sobre o tema.

Declarações

Uso de ferramentas de IA generativa: O Gemini foi utilizado neste trabalho nas etapas de aprimoramento e revisão do texto, na implementação em código dos algoritmos desenvolvidos e na elaboração dos experimentos.

Quaisquer opiniões, descobertas, conclusões ou recomendações expressas nesse material são aquelas dos autores e não necessariamente refletem a visão do Itaú Unibanco e do Instituto de Ciência e Tecnologia Itaú. Além disso, todos os dados utilizados nesse estudo estão de acordo com a Lei Geral de Proteção de Dados (LGPD).

Referências

Araujo, I. F., Araújo, I. C. S., da Silva, L. D., Blank, C., and da Silva, A. J. (2023). Quantum computing library. `https://github.com/qclib/qclib`. Version 0.0.21.

Barenco, A., Bennett, C. H., Cleve, R., DiVincenzo, D. P., Margolus, N., Shor, P., Sleator, T., Smolin, J. A., and Weinfurter, H. (1995). Elementary gates for quantum computation. *Physical Review A*, 52:3457–3467.

Batselier, K. and Wong, N. (2017). A constructive arbitrary-degree kronecker product decomposition of tensors. *Numerical Linear Algebra with Applications*, 24(5).

Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., et al. (2018). PennyLane: Automatic differentiation of hybrid quantum-classical computations.

Bergholm, V., Vartiainen, J. J., Möttönen, M., and Salomaa, M. M. (2005). Quantum circuits with uniformly controlled one-qubit gates. *Phys. Rev. A*, 71:052330.

Cai, C., Chen, R., and Xiao, H. (2022). Hybrid kronecker product decomposition and approximation. *Journal of Computational and Graphical Statistics*, 32(3):838–852.

Cormen, T., Leiserson, C., Rivest, R., and Stein, C. (2022). *Introduction to Algorithms, fourth edition*. MIT Press.

de Carvalho, J. A., Batista, C. A., de Veras, T. M. L., Araujo, I. F., and da Silva, A. J. (2024). Quantum multiplexer simplification for state preparation.

Developers, C. (2025). Cirq.

Fahrbach, M., Fu, G., and Ghadiri, M. (2022). Subquadratic kronecker regression with applications to tensor decomposition. *Advances in Neural Information Processing Systems*, 35:28776–28789.

Iten, R., Colbeck, R., Kukuljan, I., Home, J., and Christandl, M. (2016). Quantum circuits for isometries. *Physical Review A*, 93(3):032318.

Javadi-Abhari, A., Treinish, M., Krsulich, K., Wood, C. J., Lishman, J., Gacon, J., Martiel, S., Nation, P., Bishop, L. S., Cross, A. W., Johnson, B. R., and Gambetta, J. M. (2024). Quantum computing with Qiskit.

Kamm, J. and Nagy, J. G. (2000). Optimal kronecker product approximation of block toeplitz matrices. *SIAM Journal on Matrix Analysis and Applications*, 22(1):155–172.

Nielsen, M. A. and Chuang, I. L. (2010). *Quantum computation and quantum information*. Cambridge university press.

Paleologu, C., Benesty, J., and Ciochină, S. (2018). Linear system identification based on a kronecker product decomposition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(10):1793–1808.

Preskill, J. (2018). Quantum computing in the nisq era and beyond. *Quantum*, 2:79.

S., J., Yadav, S. K., and George, N. V. (2024). Adaptive low-rank doa estimation using complex kronecker product decomposition. *IEEE Transactions on Vehicular Technology*, 73(7):10726–10731.

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

Tarjan, R. E. (1975). Efficiency of a good but not linear set union algorithm. *J. ACM*, 22(2):215–225.

Van Loan, C. F. and Pitsianis, N. (1993). Approximation with kronecker products. In Moonen, M. S., Golub, G. H., and De Moor, B. L. R., editors, *Linear Algebra for Large Scale and Real-Time Applications*, volume 232 of *NATO ASI Series*, pages 293–314. Springer, Dordrecht.

Virtanen, P. et al. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Wang, X., Huang, G., Benesty, J., Chen, J., and Cohen, I. (2021). Time difference of arrival estimation based on a kronecker product decomposition. *IEEE Signal Processing Letters*, 28:51–55.

Wu, Y. (2023). A new method of kronecker product decomposition. *Journal of Mathematics*, 2023.