Quantum State Preparation Using an Exact CNOT Synthesis Formulation

Hanyu Wang
ETH, Zurich, Swizterland
hanywang@student.ethz.ch

Jason Cong
UCLA, California, United States
cong@cs.ucla.edu

Giovanni De Micheli
EPFL, Lausanne, Swizterland
giovanni.demicheli@epfl.ch

***Abstract*—Minimizing the use of CNOT gates in quantum state preparation is a crucial step in quantum compilation, as they introduce coupling constraints and more noise than single-qubit gates. Reducing the number of CNOT gates can lead to more efficient and accurate quantum computations. However, the attainment of optimal solutions is hindered by the complexity of modeling superposition and entanglement. In this paper, we propose an effective state preparation algorithm using an exact CNOT synthesis formulation. Our method represents a milestone as the first design automation algorithm to surpass manual design, reducing the best CNOT count to prepare a Dicke state by 2×. For general states with up to 20 qubits, our method reduces the CNOT count by 9% and 32% for dense and sparse states, respectively, on average, compared to the latest algorithms.**

I. Introduction

Quantum states preparation (QSP) is essential for compiling quantum algorithms [1], implementing quantum communications [2], studying quantum metrology [3], and experimenting with quantum entanglement [4]. For decades, researchers have based QSP circuit designs on mathematical derivation [5]–[8] and created them manually. These methods are effective for certain highly symmetric states, such as GHZ states [9], W states [10], and Dicke states [11]. However, their approach cannot be applied to general states.

Limitations of manual designs point us to the need to develop design automation algorithms. Recent works propose Boolean methods utilizing decision diagrams to prepare general n-qubit states using $O(2^n)$ CNOT gates [12]–[14]. By leveraging sparsity, the latest studies improve the efficiency and develop algorithms to prepare n-qubit with m nonzero amplitudes using $O(mn)$ CNOT gates [15]–[17]. However, these methods sacrifice optimality for efficiency, consuming more CNOT gates than manual designs.

The challenges of solving QSP on classical computers are the complexities of modeling superposition and entanglement [18]. Indeed, while classical computers store binary information, compute Boolean operators, and retrieve a single binary output, quantum computing processes high-dimensional state vectors with complex amplitudes that evolve with matrix multiplications. With superposition, the dimension of state vectors grows exponentially with the number of qubits. Therefore, only certain families of quantum states can be efficiently encoded using classical bits [19], [20]. Besides, because of entanglement, the qubits in the states are inseparable. As a result, developing an effective divide-and-conquer approach for QSP problems is difficult.

In this paper, we propose an exact CNOT synthesis formulation for QSP. Our method encodes quantum states and gates on a graph and formulates QSP as a shortest path problem. This formulation provides full visibility of the solution space. Targeting a library of gates specialized for QSP, our approach is guaranteed to find the optimal circuit to prepare the given state, achieving the lowest CNOT cost. For general real-amplitude states with up to 20 qubits, our method reduces the CNOT cost by 9% and 32% for dense and sparse states, on average, compared to the latest algorithms. Besides, our method represents a milestone as the first design automation algorithm to surpass manual design, reducing the best CNOT count to prepare a Dicke state by 2×.

In the rest of the paper, we present the background in Section II and show an example to motivate our work in Section III. Then, we illustrate our shortest path problem formulation and the specialized library for QSP in Section IV and introduce our specialized shortest path algorithm in Section V. In Section VI, we demonstrate experimental results and evaluate our method.

II. Background

In this section, we provide background for quantum state preparation. For clarity and space constraints, we refer readers to established sources for formal definitions of notations [18] and quantum gates [21].

A. Quantum States and Quantum Gates

We express the n-qubit *quantum state* as a linear combination of 2^n orthonormal basis vectors.

$$|\psi\rangle = \sum_{x\in S(\psi)} c_x |x\rangle, \text{ and } \sum_{x\in S(\psi)} |c_x|^2 = 1,$$

where $|x\rangle \in \{0,1\}^n$ is the *basis state*, $c_x \in \mathbb{C}$ are *amplitudes* whose norm indicates the probability of observing $|x\rangle$, and $S(\psi)$ is the *index set*, which represents the set of basis with nonzero amplitudes. The *cardinality* of a state ψ is the number of elements in its index set, $|S(\psi)|$.

Quantum gates, or *operators*, denoted by U, are unitary matrices representing transitions between quantum states. $\mathcal{U}(2^n)$ represents the set of all n-qubit gates. Y rotations, R_y, are single-qubit unitaries $U \in \mathcal{U}(2)$ with real matrices.

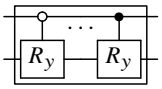
$$U = \begin{pmatrix} a & -b \\ b & a \end{pmatrix} = \begin{pmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix} = \mathrm{R}_y(\theta),$$

where a and b are real numbers satisfying $a^2 + b^2 = 1$ and θ is a rotation angle that satisfies $\theta = 2 \cdot \arctan\frac{b}{a}$.

CNOT is a two-qubit gate that transitions $\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$ to $\alpha|00\rangle + \beta|01\rangle + \gamma|11\rangle + \delta|10\rangle$. All operators in

979-8-3503-4859-0/DATE24/© 2024 EDAA

TABLE I: Selected quantum gates with their CNOT costs.

Operators	R_y	CNOT	CR_y	MCR_y
				
CNOT cost	0	1	2	$\mathcal{O}(2^n)$ [23]

$\mathcal{U}(2^n)$ can be decomposed into gates in $\{\text{CNOT}, \mathcal{U}(2)\}$ [21]. We define *CNOT cost* of a quantum operator as the number of CNOTs in the decomposition.

B. Quantum Circuits for State Preparations

Given a state $|\psi\rangle$ and a set of quantum gates $\mathcal{L}$, the *quantum state preparation* finds a quantum circuit comprising l gates $U_1, U_2, ...U_l \in \mathcal{L}$ such that these gates map the ground state $|0\rangle$ to the desired state $|\psi\rangle$, i.e., $|\psi\rangle = U_l ... U_2 U_1 |0\rangle$. For *noisy intermediate-scale quantum* (NISQ) computers, CNOTs introduce more noise than single-qubit gates, and our objective is to minimize the number of CNOTs in the circuit after decomposed into gates in $\{\text{CNOT}, \mathcal{U}(2)\}$.

This paper studies states with **real** amplitudes and state transitions in the **X-Z plane** due to their predominance in quantum algorithms. Additionally, our method adapts through a diagonal unitary phase oracle to prepare arbitrary states with complex amplitudes [22].

Relevant quantum gates and their CNOT costs are listed in Table I, including Y rotations (R_y), CNOT, and multi-controlled Y rotations (MCR_y) gates. Note that an MCR_y gate with n control qubits can implement 2^n rotation angles corresponding to each combination of control qubit values. Its CNOT cost depends on the decomposition algorithm and the number of ancillary qubits [23]–[25]. This paper assumes an MCR_y gate with n control qubits has a CNOT cost of 2^n [23].

III. Motivating Example

Consider the problem of preparing the state ψ, with $|\psi\rangle = \frac{1}{\sqrt{4}}(|000\rangle + |011\rangle + |101\rangle + |110\rangle)$ which comprises three qubits with a cardinality of four. We demonstrate the quantum circuits generated by two categories of existing methods.

The first category of methods use *qubit reduction* [12], [13]. These methods focus on one target qubit at each stage and apply MCR_y gates to separate it from the entanglement. As illustrated in Fig. 1, the gates within the solid box separate q_3, and the gates in the dashed box separate q_2. Each box represents an MCR_y gate, with the hashed having one and the solid having two control qubits, respectively. Therefore, this circuit requires $2^1+2^2=6$ CNOT gates.

Other related works perform *cardinality reduction* [15]. This method iteratively selects two different basis vectors from the index set and merges them using CNOT and controlled rotation gates. As depicted in Fig. 2, the circuit strictly reduces cardinality by one after each "merging" from right to left until $|S| = 1$, which is the ground state, $|000\rangle$. This circuit contains a single-qubit gate, three CNOTs, and two controlled Y-rotation gates. Therefore, the CNOT cost is $3\times1+2\times2=7$.

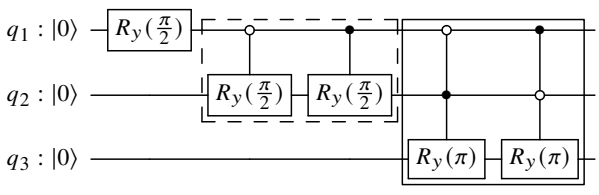


Fig. 1: 6-CNOT circuit using the qubit reduction method for $|\psi\rangle = \frac{1}{\sqrt{4}}(|000\rangle + |011\rangle + |101\rangle + |110\rangle)$.

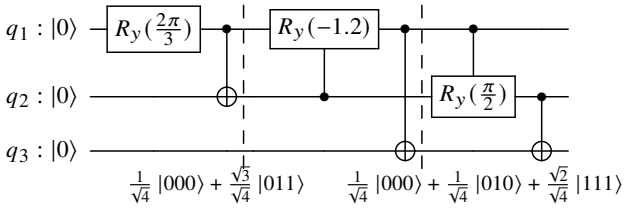


Fig. 2: 7-CNOT circuit using the cardinality reduction method.

Both qubit- and cardinality-reduction methods divide and conquer the QSP to decrease complexity. However, their heuristics introduce structural constraints to the circuit, limiting the visibility in the solution space. As a result, neither approach can reach the solution in Fig. 3 that prepares the state $|\psi\rangle$ using fewer CNOT gates. Indeed, qubit reduction methods target each qubit consecutively, while in Fig. 3, an R_y and a CNOT targeting q_1 are separated by a CNOT targeting q_3; similarly, each rotation gate in cardinality reduction methods strictly reduce the cardinality by one, while the two R_y gates shrink the cardinality directly from four to one.

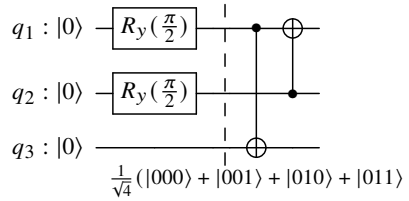


Fig. 3: 2-CNOT circuit using the exact synthesis (ours).

This example points to the need to eliminate unnecessary constraints in the problem formulation, thereby reducing the number of CNOT gates required. More specifically, heuristics that rigidly guide the state transition process often lead to greedy decisions that are locally advantageous, such as reducing entanglement between qubits or decreasing cardinality, but are globally suboptimal. The following sections demonstrate our method that explores various directions for state transition to identify the optimal solution.

IV. Exact CNOT Synthesis Formulation

Based on the necessity to explore state transitions comprehensively, we formulate the QSP as a shortest path problem. In this section, we will first detail the problem formulation and then discuss the selection of the gate library.

A. Shortest Path Problem Formulation

Given a set of quantum gates $\mathcal{L}$ with fixed CNOT costs, we define *state transition graph*, $G_{\mathcal{L}} = (V_G, A_G)$, where vertices are quantum states and arcs represent state transitions implementable by gates in $\mathcal{L}$. An arc $a = (\psi, \varphi)$ in A_G corresponds to an operator U_a such that $|\varphi\rangle = U_a |\psi\rangle$. The distance of a, denoted by $d(a)$, is a known variable and given by the CNOT cost of U_a.

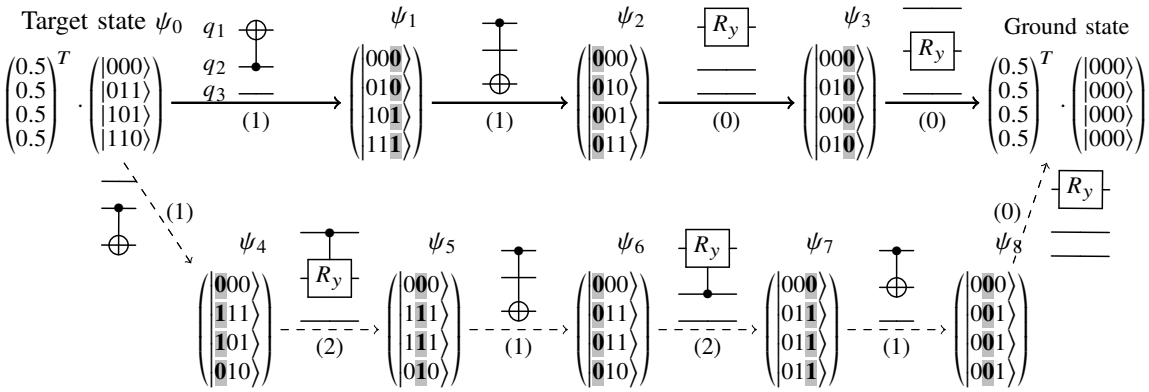


Fig. 4: Example of exploring a (portion of a) state transition graph from the target state $|\psi\rangle = \frac{1}{\sqrt{4}}(|000\rangle + |011\rangle + |101\rangle + |110\rangle)$.

Let $\gamma(\psi, \varphi)$ be the CNOT cost to prepare φ from the state ψ. For a given target state ψ_0, our objective is to find the circuit that minimizes $\gamma(|0\rangle, \psi_0)$. Note that the CNOT cost of a circuit is determined by the sum of the costs of its individual gates. Similarly, the distance of a path is calculated as the cumulative sum of the distances along its arcs. Consequently, the problems of finding the optimal circuit and the shortest path in the state transition graph are equivalent.

As an example, Fig. 4 displays a portion of the state transition from the example in Section III. We label the gates and the corresponding costs on the edges. The solid and dashed paths correspond to the circuits in Fig. 3 and Fig. 2, respectively. On the given state transition graph, the CNOT costs of these circuits are exactly given by the distances: the bold path has a distance of 1+1=2, while the distance of the dashed path is 1+2+1+2+1=7.

B. Specialized Set of Quantum Gates for QSP

The selection of library $\mathcal{L}$ determines the graph size since the vertices V_G include all reachable states using gates in $\mathcal{L}$. Let ϵ be the precision level of amplitudes. A universal set of quantum gates such as $\{\text{CNOT}, \mathcal{U}(2)\}$ can reach all n-qubit states ($\approx \epsilon^{-2^n}$). As ϵ approaches zero, the graph size increases dramatically to infinity, and the shortest path problem is impractical to solve.

To upper bound the complexity, we introduce a specialized set of quantum gates composed of single-target *amplitude-preserving* transitions, where amplitude values are conserved, but the associated basis vectors, or indices, are changed. We can describe an amplitude-preserving transition between ψ and φ as $\psi = \tilde{\sum}_x c_x |x\rangle$ and $\varphi = \tilde{\sum}_x c_x |f(x)\rangle$, where $|x\rangle$ and $|f(x)\rangle$ are basis vectors and c_x is the amplitude. The symbol $\tilde{\sum}$ denotes a specialized function that computes the root of the quadratic sum of amplitudes corresponding to the same index. It preserves the integrity of a quantum state.

For example, all state transitions in Fig. 4 are amplitude-preserving. The vertices, $\psi_1, \psi_2, ..., \psi_8$, have the amplitude vector of $(\sqrt{0.25}, \sqrt{0.25}, \sqrt{0.25}, \sqrt{0.25})$ and differ only by their basis vectors. ψ_6, for instance, corresponds to the vector $(|000\rangle, |011\rangle, |011\rangle, |010\rangle)$ with duplicated indices $|011\rangle$. It represents $\psi_6 = \sqrt{0.25}|000\rangle + \sqrt{0.25+0.25}|011\rangle + \sqrt{0.25}|010\rangle$ after merging the two amplitudes of $|011\rangle$.

Encoding amplitude-preserving transitions proves efficient on classical computers, as each state is uniquely defined by its index vector, consisting of m n-bit indices, where m represents the cardinality and n is the number of qubits. Consequently, the size of the transition graph is capped at 2^{nm}. Furthermore, as illustrated in Fig. 4, a single-target operator flips only certain bits within the same column. This allows for parallelized bit-wise Boolean operations, which are significantly more efficient than matrix multiplications involving floating-point numbers.

V. Shortest Path Algorithm

The state transition graph provides us full visibility of the solution space. This section introduces our approach to solving the shortest path problem and a state compression method designed to accelerate the algorithm. For a given state, our approach is guaranteed to find the optimal circuit comprising amplitude-preserving transitions that minimize the CNOT cost. Due to the space limit, we refer readers to the appendices for the proof of optimality and complexity [26].

A. Admissible Heuristic Function and A* Algorithm

A* algorithm is the general solution to the shortest path problem, whose pseudocode is shown in Algorithm 1. A `while` loop at line 4 traverses all the reachable quantum states in ascending order of CNOT cost from the target state ψ. We iteratively pop the top element in the queue, enumerate all of its adjacent states, and add those states to the queue if a lower distance is found. After reaching the ground state, our algorithm backtracks the path and returns the quantum circuit by mapping each edge to the corresponding quantum operator.

The key idea of the A* algorithm is to find a *lower bound estimation* for the distance between state ψ and the destination, denoted by $\hat{\delta}(\psi, |0\rangle)$. Then, the queue sorts the states based on the sum of $\gamma(\psi_0, \psi) + \hat{\delta}(\psi, |0\rangle)$. This way, we prioritize the state more likely to be the shortest path that potentially reaches the destination and returns earlier. If the function is *admissible*, i.e., $\hat{\delta}(\psi, |0\rangle)$ always underestimates the true cost between ψ and $|0\rangle$, then the A* heuristic can prune unpromising branches and improve efficiency without loss in optimality.

Algorithm 1: A* algorithm

input : The target quantum state ψ_0.
output: A sequence of quantum operators to prepare ψ_0 from the ground state.

1 q ← PriorityQueue()
2 $\gamma(\psi_0, :) \leftarrow \infty$, $\gamma(\psi_0, \psi_0) \leftarrow 0$
3 q.put(ψ_0)
4 **while** *q is not empty* **do**
5 $\quad \psi \leftarrow$ q.pop()
6 $\quad$ **if** $\psi = |0\rangle$ **then**
7 $\quad\quad$ **break**
8 $\quad$ **for** $(\psi, \varphi) \in A_G$ **do**
9 $\quad\quad \gamma' \leftarrow \gamma(\psi_0, \Pi(\psi)) + d(\psi, \varphi)$.
10 $\quad\quad$ **if** $\gamma' \geq \gamma(\psi_0, \Pi(\varphi))$ **then**
11 $\quad\quad\quad$ **continue**
12 $\quad\quad \gamma(\psi_0, \Pi(\varphi)) \leftarrow \gamma'$
13 $\quad\quad$ q.put(φ)
14 path ← back trace the edges from ψ_0 to ground state.
15 **return** path.

Given a state ψ, we derive a lower bound on CNOT by checking the number of entangled qubit pairs, which can be acquired by evaluating *mutual information* [27]. Take state ψ_1 in Fig. 4 as an example. The *cofactors* of q_2, the sets of indices with q_2=1 ($\psi_1|_{q_2=1}$) and q_2=0 ($\psi_1|_{q_2=0}$), are identical:

$$S\left(\psi_1|_{q_2=1}\right) = S\left(\psi_1|_{q_2=0}\right) = \left\{|00\rangle_{q_1q_3}, |11\rangle_{q_1q_3}\right\},$$

which indicates q_2 might be separable. Meanwhile, both q_1 and q_3 have different cofactors, which implies that they are entangled with other qubits, and at least one CNOT is required to separate every two entangled qubits.

B. State Compression using Canonicalization

Using the A* algorithm, our solver does not explicitly construct the graph comprising all 2^{nm} states but explores a small portion around the given target. To further decrease the time complexity, we introduce a state compression heuristic that reduces the number of enqueued states.

The idea is based on *canonicalization*. We associate states with their *equivalence class*, denoted by $V_G/\sim$, a set of equivalent states under relation $\sim$, which is defined as follows:

- Single-qubit unitaries, $\mathcal{U}(2)$: Two states ψ and φ are equivalent if a unitary operation $U \in \mathcal{U}(2)$ exists such that $\psi = U\varphi$.
- Qubit permutation, designated by $\mathcal{P}$: Two states are equivalent if they differ only by the swapping of the definition of two qubits.

For example, the state $\varphi = \frac{1}{\sqrt{2}}(|100\rangle + |010\rangle)$ is equivalent to:

- $\psi_1 = \frac{1}{\sqrt{2}}(|000\rangle + |110\rangle)$ because applying a Pauli-X gate on the first or the second qubit transitions φ to ψ_1.
- $\psi_2 = \frac{1}{\sqrt{4}}(|000\rangle + |001\rangle + |110\rangle + |111\rangle)$ because applying a $R_y(\frac{\pi}{2})$ gate on the last qubit further transitions ψ_1 to ψ_2.
- $\psi_3 = \frac{1}{\sqrt{2}}(|100\rangle + |001\rangle)$ because it is equivalent to φ after swapping the definition of the second and the last qubit.

We can utilize equivalent relations to compress states. Since the gates in $\mathcal{U}(2)$ have zero CNOT cost, the states $\psi, \varphi \in V_G/\mathcal{U}(2)$ can be prepared using the same number of CNOT gates. Similarly, if all qubits are interchangeable, which implies a symmetric coupling graph, then states $\psi, \varphi \in V_G/\mathcal{PU}(2)$ should have the same CNOT cost. Therefore, in lines 10 to 13 in Algorithm 1, we select a *representative*, $\Pi(\varphi)$, to store the distance of the equivalence class of state φ. We can skip exploring state φ if another state in the same equivalence class is enqueued with a lower or equal distance.

TABLE II: Number of canonical 4-qubit uniform states.

m	$\lvert V_G\rvert$	$\lvert V_G/\mathcal{U}(2)\rvert$	$\lvert V_G/\mathcal{PU}(2)\rvert$
1	16	1	1
2	120	11	3
3	560	35	6
4	1820	118	16
5	4368	273	27
6	8008	525	47
7	11440	715	56
8	12870	828	68

Table II presents the graph size without an equivalency relation ($|V_G|$), with a layout-variant equivalency $|V_G/\mathcal{U}(2)|$, and with a layout-invariant equivalency $|V_G/\mathcal{PU}(2)|$. The results demonstrate the effectiveness of state compression using canonicalization.

VI. Evaluation

The proposed algorithm is implemented using Python and is open-source.[1] In this section, we illustrate the effectiveness and efficiency of our method compared with manual designs and three recent works: a cardinality reduction method (**"m-flow"**) [15], a qubit reduction method (**"n-flow"**) [13], and the latest QSP solver: a hybrid method combining n- and m-flows (**"hybrid"**) [16]. The hybrid method requires one ancilla, while all others employ no ancilla qubits. We run all the experiments on a classical computer with a 3.7GHz AMD Ryzen 9 5900X processor with 64GB of RAM.

A. Workflow

The workflow is illustrated in Fig. 5, where we integrate our exact CNOT synthesis into a scalable framework. Given the state with n qubits and cardinality m, we choose the divide-and-conquer method based on the sparsity. If the state is sparse ($nm < 2^n$), we iteratively run the cardinality reduction method until the complexity is acceptable for exact CNOT

[1] Available at https://github.com/Nozidoali/quantum-xyz

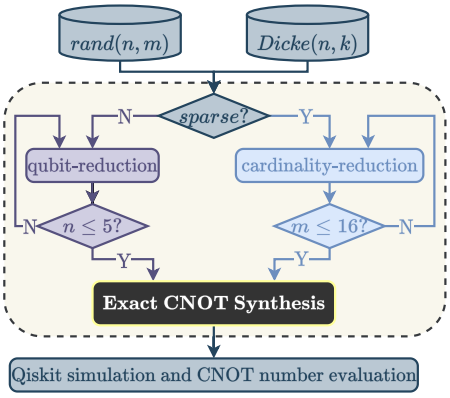


Fig. 5: Evaluation workflow of exact CNOT synthesis.

TABLE III: CNOT count comparison results of Dicke state preparation. We calculate the improvements achieved by four design automation algorithms compared to the manual design.

Dicke states preparation $\|D\rangle_n^k$						
n	k	Manual [7]	m-flow [15]	n-flow [13]	Hybrid [16]	**ours**
3	1	**4**	5	6	27	**4**
4	1	**7**	9	14	141	**7**
4	2	12	24	14	141	**6**
5	1	**10**	13	30	237	**10**
5	2	20	67	30	601	**16**
6	1	**13**	17	62	241	**13**
6	2	28	117	62	645	**22**
6	3	33	231	62	779	**25**
geo. mean		13.0	28.5	26.6	251.1	**10.9**
Impr%		-	-119%	-105%	-1832%	**17%**

synthesis; otherwise, if the state is dense ($nm \geq 2^n$), we apply qubit reduction before an exact synthesis. We use the library introduced in Section IV-B. We verify the correctness of the circuits returned by the QSP solver employing Qiskit simulators [28] and evaluate the number of CNOT gates after mapping the circuit to $\{\mathcal{U}(2), \text{CNOT}\}$.

B. Preparation of Dicke States

Dicke states, denoted by $|D\rangle_n^k$, are the family of quantum states of n qubits with nonzero coefficient if k out of the n qubits in the basis state is $|1\rangle$. Due to its importance and wide applications [6], various manually designed techniques have been proposed [5]–[8]. The latest manual design requires $(5nk - 5k^2 - 2n)$ CNOT gates to prepare $|D\rangle_n^k$ [7].

Table III presents the CNOT count of each method and the improvements compared to the manual design [7], which demonstrates the effectiveness of the exact synthesis method on small states: our method achieves the best CNOT cost among four methods on **all** the benchmarks.

Moreover, our method is the first design automation algorithm that outperforms manual designs. Utilizing the exact CNOT synthesis formulation, our algorithm implicitly derives more complicated properties throughout the vast solution space exploration. While the properties developed manually are mostly symmetric or inductive, our solver synthesizes irregular yet effective circuits that are not easily generalizable, as shown in Fig. 6. We reduce the CNOT count from 12 to 6.

C. Preparation of General Quantum States

Then, we evaluate the generality of our method by applying exact CNOT synthesis on asymmetry states with various numbers of qubits with different cardinalities. Our benchmark suites contain dense states with cardinality $m = 2^{n-1}$ and the sparse states with $m = n$. We randomly sample 100 different states for each parameter setup and present the average number of CNOT gates in Table IV.

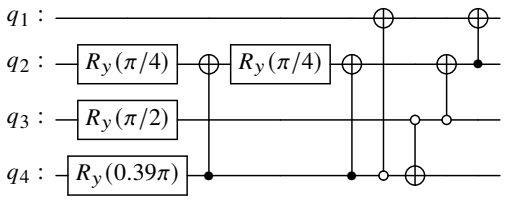


Fig. 6: Circuit to prepare $|D\rangle_4^2$ using 6 CNOTs

TABLE IV: CNOT number comparison results of general state preparation. "TLE" represents that the method fails to prepare the state within a one-hour time limit. We calculate the improvement of our method compared to the most effective baseline in each category: to n-flow for dense states and to m-flow for sparse states.

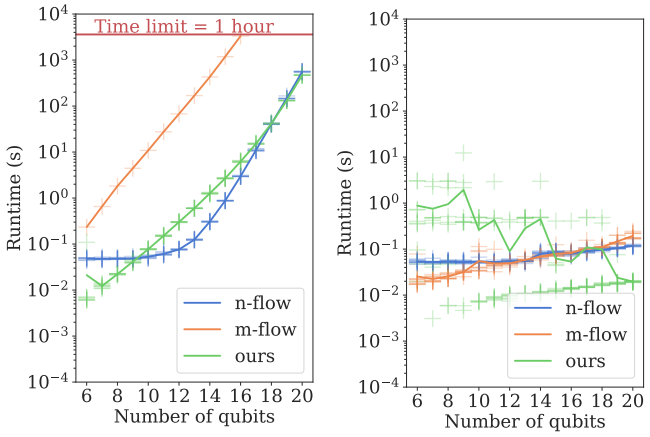
Dense states preparation ($m = 2^{n-1}$)						
n	m	m-flow [15]	n-flow [13]	Hybrid [16]	**ours**	impr%
3	4	8	6	5	5	17%
4	8	40	14	28	9	36%
5	16	123	30	401	29	3%
6	32	478	62	899	56	10%
7	64	1382	126	1268	112	11%
8	128	3954	254	2804	226	11%
9	256	10902	510	6911	484	5%
10	512	28743	1022	15646	962	6%
11	1024	72441	2046	35650	1812	11%
12	2048	178996	4094	82836	3846	6%
13	4096	440843	8190	183556	7746	5%
14	8192	1053633	16382	398602	15630	5%
15	16384	2487775	32766	879236	31254	5%
16	32768	5810670	65534	1915109	63720	3%
17	65536	TLE	131070	4109698	128330	2%
18	131072	TLE	262142	8802090	261684	0%
geo. mean			1399.3	18855.9	**1274.7**	**9%**
Sparse states preparation ($m = n$)						
n	m	m-flow [15]	n-flow [13]	Hybrid [16]	**ours**	impr%
3	3	4	6	7	3	37%
4	4	9	14	38	6	34%
5	5	14	30	83	9	36%
6	6	22	62	181	14	36%
7	7	30	126	253	20	33%
8	8	39	254	398	27	30%
9	9	51	510	472	37	29%
10	10	66	1022	560	44	33%
11	11	80	2046	679	54	33%
12	12	97	4094	763	66	32%
13	13	114	8190	878	78	31%
14	14	130	16382	965	91	30%
15	15	152	32766	1095	106	30%
16	16	172	65534	1187	119	31%
17	17	201	131070	1268	139	31%
18	18	217	262142	1399	155	29%
19	19	238	524286	1491	173	28%
20	20	267	1048574	1605	192	28%
geo. mean		64.3	2809.3	429.8	**44**	**32%**

We observe that the disparity in divide-and-conquer approaches between n-flow and m-flow leads to distinct benefits. When preparing states with n qubits and cardinality of m, the n-flow exhibits a CNOT count bounded by $O(2^n)$, which is well-suited for dense states. On the other hand, the m-flow requires a CNOT count of $O(mn)$ and proves advantageous for sparse states. Meanwhile, the CNOT count of the hybrid method falls between the n-flow and the m-flow.

After employing the exact CNOT synthesis, we improve the results on both dense and sparse states. Compared with the corresponding advantageous baseline, our method reduces the CNOT count by 9% and 32% on dense and sparse states, respectively. Since we set fixed thresholds ($n \leq 4$ and $m \leq 16$) to activate the exact synthesis in our workflow, the room for improving the CNOT count does not scale with the problem size. Therefore, the relative benefit of exact synthesis decreases as n and m increases. We are investigating a more effective integration of the exact synthesis to scale the improvements.

D. CPU Time Analysis

Exact CNOT synthesis searches the shortest path with full visibility of the solution space, which naturally has a higher time complexity than the baselines. We investigate the scalability comparison in Fig. 7, where we plot the CPU



(a) Dense states ($m = 2^{n-1}$) (b) Sparse states ($m = n$)

Fig. 7: CPU time analysis on (a) dense states and (b) sparse states compared with two baselines: the n-flow [13], and the m-flow [15]. For clarity, we omit the CPU time of the hybrid method [16], which falls between n-flow and m-flow.

time with the growth of qubit numbers. We show the results for dense states and sparse states separately in Fig. 7a and Fig. 7b, as their divide-and-conquer procedures are different, as mentioned in Section VI-A.

The slope in Fig. 7a demonstrates the robustness of our state encoding. We use $n \times m$ classical bits to encode a quantum state while the m-flow [15] inherits Qiskit's data structure [28], which stores indices as `string`. The n-flow [13] and the hybrid method [16] use `C++` CUDD[2] library with truth tables of 2^n bits. Moreover, although all three methods are efficient on sparse states, our canonicalization method helps filter out separable qubits, further compressing the states and accelerating the solver. As the number of qubits increases, fixing $m = n$ will likely decrease the degree of entanglement. Our state compression algorithm identifies the separability and reduces the complexity, as shown in Fig. 7b. Therefore, although integrated with exact CNOT synthesis, our flow consumes CPU time comparable to baseline methods and exhibits better scalability as the number of qubits increases.

VII. Conclusion

Quantum state preparation (QSP) initializes quantum superposition which is essential in quantum computing. Synthesizing efficient circuits for QSP improves the accuracy of quantum algorithms. However, the characterization of superposition and entanglement hinders the development of classical algorithms to automate quantum designs. This paper formulates QSP as a shortest path problem, encodes quantum states on a graph, and finds the optimal circuit with the lowest CNOT cost. Equipped with the A* algorithm and state compression heuristics, our method solves QSP efficiently without any loss in optimality. Compared to existing design automation algorithms, our method improves the CNOT count by 9% and 32% for dense and sparse state preparation, on average, using comparable CPU time. On a practical QSP problem, we reduce the best CNOT cost by 2×, which is the first time design automation algorithms surpass manual designs.

[2]CUDD: CU Decision Diagram package, https://github.com/ivmai/cudd.

Acknowledgement

The authors would like to thank Bochen Tan for constructive comments on the methodology. Dr. Mathias Soeken for the helpful discussion, and Marci Baun for proofreading the paper.

References

[1] A. M. Childs *et al.*, "Finding cliques by quantum adiabatic evolution," *arXiv preprint quant-ph/0012104*, 2000.
[2] M. Ben-Or and A. Hassidim, "Fast quantum byzantine agreement," in *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, 2005, pp. 481–485.
[3] G. Tóth, "Multipartite entanglement and high-precision metrology," *Physical Review A*, vol. 85, no. 2, p. 022322, 2012.
[4] R. Horodecki *et al.*, "Quantum entanglement," *Reviews of Modern Physics*, vol. 81, no. 2, p. 865, 2009.
[5] D. Cruz *et al.*, "Efficient quantum algorithms for GHZ and W states, and implementation on the IBM quantum computer," *Advanced Quantum Technologies*, vol. 2, no. 5-6, p. 1900015, 2019.
[6] A. Bärtschi and S. Eidenbenz, "Deterministic preparation of Dicke states," in *International Symposium on Fundamentals of Computation Theory*. Springer, 2019, pp. 126–139.
[7] C. S. Mukherjee *et al.*, "Preparing Dicke states on a quantum computer," *IEEE Transactions on Quantum Engineering*, vol. 1, pp. 1–17, 2020.
[8] S. Aktar *et al.*, "A divide-and-conquer approach to Dicke state preparation," *IEEE Transactions on Quantum Engineering*, vol. 3, pp. 1–16, 2022.
[9] D. M. Greenberger *et al.*, "Going beyond Bell's theorem," in *Bell's Theorem, Quantum Theory and Conceptions of the Universe*. Springer, 1989, pp. 69–72.
[10] W. Dür *et al.*, "Three qubits can be entangled in two inequivalent ways," *Physical Review A*, vol. 62, no. 6, p. 062314, 2000.
[11] R. H. Dicke, "Coherence in spontaneous radiation processes," *Physical Review*, vol. 93, no. 1, p. 99, 1954.
[12] I. F. Araujo *et al.*, "A divide-and-conquer algorithm for quantum state preparation," *Scientific Reports*, vol. 11, no. 1, p. 6329, 2021.
[13] F. Mozafari, M. Soeken, and G. De Micheli, "Preparation of uniform quantum states utilizing boolean functions," in *28th International Workshop on Logic Synthesis (IWLS)*, 2019.
[14] P. Niemann, R. Datta, and R. Wille, "Logic synthesis for quantum state generation," in *2016 IEEE 46th International Symposium on Multiple-Valued Logic (ISMVL)*. IEEE, 2016, pp. 247–252.
[15] N. Gleinig and T. Hoefler, "An efficient algorithm for sparse quantum state preparation," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 433–438.
[16] F. Mozafari *et al.*, "Efficient deterministic preparation of quantum states using decision diagrams," *Physical Review A*, vol. 106, no. 2, p. 022617, 2022.
[17] E. Malvetti *et al.*, "Quantum circuits for sparse isometries," *Quantum*, vol. 5, p. 412, 2021.
[18] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge University Press, 2010.
[19] S. Aaronson, "Multilinear formulas and skepticism of quantum computing," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, 2004, pp. 118–127.
[20] R. Raz, "Multi-linear formulas for permanent and determinant are of super-polynomial size," *Journal of the ACM (JACM)*, vol. 56, no. 2, pp. 1–17, 2009.
[21] A. Barenco *et al.*, "Elementary gates for quantum computation," *Physical Review A*, vol. 52, no. 5, p. 3457, 1995.
[22] M. Amy *et al.*, "On the controlled-not complexity of controlled-not–phase circuits," *Quantum Science and Technology*, vol. 4, no. 1, p. 015002, Sep. 2018. [Online]. Available: http://dx.doi.org/10.1088/2058-9565/aad8ca
[23] M. Möttönen *et al.*, "Quantum circuits for general multiqubit gates," *Physical Review Letters*, vol. 93, no. 13, p. 130502, 2004.
[24] D. Maslov, "Advantages of using relative-phase Toffoli gates with an application to multiple control Toffoli optimization," *Physical Review A*, vol. 93, no. 2, p. 022311, 2016.
[25] M. Saeedi and M. Pedram, "Linear-depth quantum circuits for n-qubit Toffoli gates with no Ancilla," *Physical Review A*, vol. 87, no. 6, p. 062318, 2013.
[26] H. Wang, B. Tan, J. Cong, and G. D. Micheli, "Quantum state preparation using an exact cnot synthesis formulation," 2024.
[27] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.
[28] Qiskit contributors, "Qiskit: An open-source framework for quantum computing," 2023.