

“© 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.”

Quantum State Preparation Based on LimTDD

Xin Hong¹, Chenjian Li¹, Aochu Dai², Sanjiang Li³, Shenggang Ying¹ and Mingsheng Ying³

¹Key Laboratory of System Software (Chinese Academy of Sciences) and State Key Laboratory of Computer Science Institute of Software, Chinese Academy of Sciences, Beijing, China

²Department of Computer Science and Technology, Tsinghua University, Beijing, China

³Centre for Quantum Software and Information, University of Technology Sydney, Sydney, Australia

Emails: {hongxin, licj, yingsg}@ios.ac.cn, dac22@mails.tsinghua.edu.cn, {sanjiang.li, mingsheng.ying}@uts.edu.au

Abstract—Quantum state preparation is a fundamental task in quantum computing and quantum information processing. With the rapid advancement of quantum technologies, efficient quantum state preparation has become increasingly important. This paper proposes a novel approach for quantum state preparation based on the Local Invertible Map Tensor Decision Diagram (LimTDD). LimTDD combines the advantages of tensor networks and decision diagrams, enabling efficient representation and manipulation of quantum states. Compared with the state-of-the-art quantum state preparation method, LimTDD demonstrates substantial improvements in efficiency when dealing with complex quantum states, while also reducing the complexity of quantum circuits. Examples indicate that, in the best-case scenario, our method can achieve exponential efficiency gains over existing methods. This study not only highlights the potential of LimTDD in quantum state preparation but also provides a robust theoretical and practical foundation for the future development of quantum computing technologies.

Index Terms—quantum state preparation, decision diagrams, quantum circuits

I. INTRODUCTION

Quantum computing, as a cutting-edge computational technology, holds extremely significant importance. By leveraging the principles of superposition and entanglement of quantum bits (qubits), quantum computation achieves exponential acceleration in certain computational tasks, thereby demonstrating substantial advantages in solving complex problems. For example, in the field of cryptography, quantum computing can rapidly factor large integers, posing a challenge to traditional encryption methods [1]. In material science and drug development, quantum computing can accurately simulate the behaviour of molecules and chemical reactions, accelerating the discovery of new materials and drugs [2], [3]. Additionally, quantum computing has a broad range of application prospects in fields such as optimisation problems and artificial intelligence [4], [5].

Quantum state preparation (QSP) is one of the fundamental tasks in quantum computing and quantum information processing. In quantum computing, the implementation of many quantum algorithms relies on the ability to precisely prepare specific quantum states, such as entangled states and superposition states. However, as the scale of quantum systems increases, the complexity of quantum states grows exponentially, making the efficient preparation of quantum states a highly challenging problem.

In recent years, with the rapid development of quantum technologies, significant progress has been made in developing (quantum) algorithms for QSP. Many methods have been established to study the state preparation of special quantum states, such as sparse quantum states [6], [7], [8]. Methods based on the gate decomposition [9], [10], [11] and uniformly controlled rotations [12] have been proposed for the preparation of general quantum states. Some works focus

on preparing a quantum state with the minimal number of ancilla qubits [13], and some works focus on preparing a quantum state with the minimal circuit depth [14]. At the same time, theoretical upper and lower bounds have also been identified [15] for QSP. However, most of the methods are established on explicit representations of the quantum state, such as the vector representation, whose size grows exponentially as the number of qubits increases, restricting the size of the quantum state that can be prepared.

Decision diagrams, as an efficient mathematical tool and a compact data structure, have been widely used in classical circuit synthesis [16] and verification [17]. In recent years, researchers have begun to explore the application of decision diagrams in the quantum field. By adapting classical decision diagrams to support quantum operations or developing new decision diagrams, researchers have made progress in efficient simulation and verification of quantum circuits [18], [19]. There are also several works using decision diagrams to conduct QSP, such as [20], [21], [22]. Due to the compactness of decision diagrams, these algorithms demonstrate high efficiency and are capable of supporting the preparation of relatively large-scale quantum states. Mozafari et al. proposed an efficient algorithm for the preparation of uniform quantum states [20]. The algorithm in [21] uses one auxiliary qubit, with complexity related to the number of paths in the decision diagram, while the algorithm in [22] employs multiple auxiliary qubits, with complexity related to the number of nodes in the decision diagram. Although these algorithms are highly efficient, the compression efficiency of the decision diagrams they employ may potentially affect the effectiveness of the final state preparation.

Recently, a new decision diagram, Local Invertible Map Tensor Decision Diagram (LimTDD) [23], has been proposed and shows great compactness compared to other decision diagrams. LimTDD combines the strengths of TDD and LIMDD, can be used to represent any tensors and can identify the isomorphic structure between tensors, achieving an exponential gap between other decision diagrams, such as TDD and LIMDD, in the best-case scenario.

This paper introduces a novel quantum state preparation algorithm based on LimTDD. By leveraging the extreme compression efficiency of LimTDD, our algorithm can handle large-scale quantum states more efficiently, while also reducing the number of quantum gates and circuit depth. This work not only highlights the potential of LimTDD in QSP but also provides a robust foundation for the future development of quantum computing technologies. We have developed an interface that converts state vectors into LimTDDs and then synthesises these representations into executable state preparation circuits. This capability is designed to streamline the workflow for quantum physicists and experimentalists, enabling them to efficiently generate the quantum states required for their research. In future work, we plan to integrate our algorithm into widely used quantum computing frameworks, such as Qiskit, to further accelerate and standardise the QSP process.

Work partially supported by Innovation Program for Quantum Science and Technology under Grant No. 2024ZD0300502, Beijing Nova Program Grant No. 20220484128 and 20240484652, and the National Natural Science Foundation of China Grant No. 12471437.

The structure of this paper is as follows. In section II, we provide basic concepts of quantum computing and QSP. In section III, we introduce how to represent quantum states as LimTDDs. In section IV, we give the basic algorithms for QSP using LimTDD. A detailed example is given in section V. The complexity of the algorithm is analysed in section VI. Then, we will conduct experiments to carefully analyse the performance of our algorithm in section VII. At last, in section VIII, we conclude the paper.

II. BACKGROUND

In this section, we give basic background on quantum computing and quantum state preparation.

A. Quantum Computing

1) *Quantum States*: Quantum computing is a paradigm that leverages the principles of quantum mechanics to perform computations. Unlike classical computing, which operates on classical bits that can be in a state of either 0 or 1, quantum computing utilises quantum bits (qubits) that can exist in superpositions of states. A qubit is described by a two-dimensional complex vector space, with the computational basis states denoted as $|0\rangle$ and $|1\rangle$. These basis states are orthonormal, satisfying the conditions:

$$\langle 0|0\rangle = \langle 1|1\rangle = 1 \quad \text{and} \quad \langle 0|1\rangle = \langle 1|0\rangle = 0.$$

A general state of a qubit can be represented as a linear combination of these basis states:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle.$$

where α and β are complex numbers known as probability amplitudes, satisfying the normalisation condition $|\alpha|^2 + |\beta|^2 = 1$. This superposition property allows a qubit to represent multiple states simultaneously, providing a fundamental advantage over classical bits.

For multi-qubit systems, the state space grows exponentially as the number of qubits increases. For instance, an n -qubit system is described by a 2^n -dimensional complex vector, with the computational basis states denoted as $|k\rangle$, where k is a binary string of length n . For example, the state $|000\rangle$ represents the state where all three qubits are in the $|0\rangle$ state, and can be written as:

$$|000\rangle = |0\rangle \otimes |0\rangle \otimes |0\rangle.$$

Example 1: Consider the 3-qubit quantum state $\frac{1}{\sqrt{6}}(|000\rangle + |001\rangle + \frac{1}{\sqrt{2}}|010\rangle - \frac{1}{\sqrt{2}}|011\rangle - |100\rangle - \frac{1}{\sqrt{2}}|101\rangle + \frac{1}{\sqrt{2}}|110\rangle + |111\rangle)$. This state can be represented as an 8-dimensional vector: $\frac{1}{\sqrt{6}}[1, 1, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, -1, -\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 1]^T$. Here, each component of the vector corresponds to a specific computational basis state.

2) *Quantum Gates*: Quantum computations are performed through the application of quantum gates, which are unitary transformations on the qubits. Common quantum gates include:

- **Hadamard gate (H gate)**: This gate creates a superposition state by transforming the basis states $|0\rangle$ and $|1\rangle$ into equal superpositions:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad \text{and} \quad H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

- **Pauli-Z gate (Z gate)**: This gate introduces a phase flip to the $|1\rangle$ state and remain the state $|0\rangle$ unchanged:

$$Z|0\rangle = |0\rangle \quad \text{and} \quad Z|1\rangle = -|1\rangle.$$

- **Controlled-X gate (CX gate)**: Also known as the CNOT gate, this gate performs a NOT operation on a target qubit conditioned on the state of a control qubit. For example, if the control qubit is in the state $|1\rangle$, the target qubit's state is flipped:

$$CX(|0\rangle \otimes |\psi\rangle) = |0\rangle \otimes |\psi\rangle \quad \text{and} \quad CX(|1\rangle \otimes |\psi\rangle) = |1\rangle \otimes X|\psi\rangle.$$

3) *Quantum Circuits*: Quantum circuits are constructed by combining these basic gates in specific sequences to implement various quantum algorithms. Each gate is applied to one or more qubits, and the sequence of gates determines the overall transformation applied to the qubits. The output of a quantum circuit is typically measured in the computational basis, providing the result of the computation.

Quantum circuits can be represented graphically, with qubits as horizontal lines and gates as symbols applied to these lines. This graphical representation helps visualise the sequence of operations and the flow of information through the circuit.

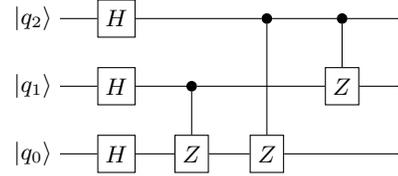


Fig. 1. Example of a quantum circuit with Hadamard gates and CZ gates.

Fig. 1 gives an example of quantum circuits. In this example, the quantum circuit consists of Hadamard gates and CZ gates applied to three qubits. The Hadamard gates create superposition states, and the CZ gates introduce entanglement. The specific sequence of gates determines the transformation applied to the qubits, which can be used to implement various quantum algorithms.

B. Quantum State Preparation

Quantum state preparation is a fundamental task in quantum computing and quantum information processing, aiming to transform a given initial state into a desired target quantum state. This process is crucial for the implementation of various quantum algorithms, as many algorithms require specific quantum states as inputs to achieve their computational advantages.

1) *Definition and Objective*: Formally, quantum state preparation can be defined as follows. Given an initial state, typically $|0\rangle^{\otimes n}$, and a target quantum state $|\psi_v\rangle = \sum_{k=0}^{2^n-1} v_k |k\rangle$, where $v = (v_0, v_1, \dots, v_{2^n-1})^T \in \mathbb{C}^{2^n}$ is a normalized vector ($\|v\|_2 = 1$) representing the amplitudes of the target state in the computational basis, the objective of QSP is to construct a quantum circuit that transforms the initial state into the target state $|\psi_v\rangle$.

2) *Challenges and Importance*: As the number of qubits n increases, the complexity of representing and manipulating quantum states grows exponentially. Specifically, a general n -qubit state requires 2^n complex amplitudes to be fully described, making the explicit representation and preparation of arbitrary quantum states computationally intractable in general cases. Efficient QSP is thus a highly challenging problem, especially for large quantum systems. However, it is also essential for practical quantum computing applications, as the efficiency of QSP directly impacts the feasibility and performance of many quantum algorithms.

3) *Existing Approaches and Limitations*: In [21], Mozafari et al. proposed an efficient algorithm for QSP using decision diagrams. The proposed algorithm starts from the $|0\rangle$ state and prepares the quantum state by traversing the decision diagram path by path. It uses an

ancilla qubit to mark the paths that have already been processed. By using this auxiliary qubit as a control qubit, the subsequent preparation process will not affect the paths that have already been prepared. As a result, its time complexity and the gate complexity of the resulting circuit are proportional to the number of paths in the decision diagram.

However, the decision diagram used in [21] is a multi-terminal Algebraic Decision Diagram (ADD), which has a relatively low compactness representing quantum states. For example, the quantum state given in Example 1 requires seven paths to be represented as an ADD as shown in Fig. 2 (a), while for other decision diagrams such as LimTDD, only two (reduced) paths are needed.

The algorithm presented in this paper is inspired by the work of [21]. But differs significantly in details, especially that we use an ancilla qubit to indicate the open or closed state of a node rather than a path. This difference is determined by the inherent characteristics of the two different decision diagrams.

III. REPRESENTING QUANTUM STATE USING LIMTDD

LimTDD is a highly compact decision diagram designed for representing and operating tensors and tensor networks in an efficient way. In this paper, we use the data structure LimTDD to represent quantum states, which is capable of representing a quantum state with a very high compression ratio. This feature ensures efficient preparation and the preparation of large-scale quantum states.

A. LimTDD

The efficient representation of LimTDD is established on the isomorphism of quantum states.

Definition 1 (LIM, Quantum State Isomorphism): An n -qubit Local Invertible Map (LIM) is an operator O of the form

$$O = \lambda O_n \otimes \cdots \otimes O_1, \quad (1)$$

where $\lambda \in \mathbb{C}$ is a complex number and each O_i is an invertible 2×2 matrix. The set of all such maps is denoted as $\mathcal{M}(n)$, and the set of all LIMs, which is a group, is defined as

$$\mathcal{M} = \bigcup_{n \in \mathbb{N}} \mathcal{M}(n). \quad (2)$$

Two n -qubit quantum states $|\Psi\rangle$ and $|\Phi\rangle$ are said to be *isomorphic* if $|\Phi\rangle = O|\Psi\rangle$ for some $O \in \mathcal{M}(n)$.

Definition 2 (LimTDD): Let \mathcal{G} be a subgroup of \mathcal{M} . A \mathcal{G} -LimTDD \mathcal{F} over a set of indices S is a rooted, weighted, and directed acyclic graph $\mathcal{F} = (V, E, \text{idx}, \text{low}, \text{high}, \text{wt})$ defined as follows:

- V is a finite set of nodes which consists of non-terminal nodes V_{NT} and a terminal node v_T labelled with integer 1. Denote by $r_{\mathcal{F}}$ the unique root node of \mathcal{F} ;
- $\text{idx} : V_{NT} \rightarrow S$ assigns each non-terminal node an index in S . We call $\text{idx}(r_{\mathcal{F}})$ the top index of \mathcal{F} , if $r_{\mathcal{F}}$ is not the terminal node;
- both low and high are mappings in $V_{NT} \rightarrow V$, which map each non-terminal node to its 0- and 1-successors, respectively;
- $E = \{(v, \text{low}(v)), (v, \text{high}(v)) : v \in V_{NT}\}$ is the set of edges, where $(v, \text{low}(v))$ and $(v, \text{high}(v))$ are called the low- and high-edges of v , respectively. For simplicity, we also assume the root node $r_{\mathcal{F}}$ has a unique incoming edge, denoted e_r , which has no source node;
- $\text{wt} : E \rightarrow \mathcal{G}$ assigns each edge a weight in \mathcal{G} . $\text{wt}(e_r)$ is called the weight of \mathcal{F} , and denoted $w_{\mathcal{F}}$.

When representing a quantum state, the semantics of the terminal node is defined to be $|v_T\rangle = |1\rangle$, the semantics of an edge e , directing to a node v , is defined to be

$$|e\rangle = \text{wt}(e) \cdot |v\rangle,$$

and the semantics of a non-terminal node v is defined to be

$$|v\rangle = |0\rangle \otimes |(v, \text{low}(v))\rangle + |1\rangle \otimes |(v, \text{high}(v))\rangle$$

Note that, when representing quantum states, every index corresponds to a qubit. In this paper, for convenience, we always assume that the top index corresponds to the most significant qubit (that is, q_{n-1} , for an n -qubit quantum state), and index of the bottom non-terminal node corresponds to the least significant qubit (that is, q_0), with the other qubits arranged in sequence. Sometimes, to identify the qubits, we use notations such as $|0\rangle_{n-1}$ and $|0\rangle_0$. The subgroup \mathcal{G} is set to be XP-Operators in [23], however, the algorithm in this paper works for any subgroup of \mathcal{M} .

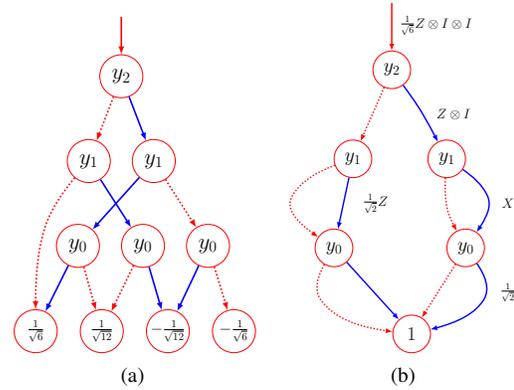


Fig. 2. An example of Multiple-terminal ADD [21] and LimTDD representing the quantum state $\frac{1}{\sqrt{6}}[1, 1, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, -1, -\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 1]^T$. We omit the weight 1 and $1 \cdot I^{\otimes k}$ in the figure of LimTDD.

Example 2: Fig. 2 (b) gives an example of LimTDD, which represents the quantum state shown in example 1. In this figure, we use dotted red lines to represent the low edges and solid blue lines to represent the high edges. We refer to the node with index y_2 as the y_2 node. The node on the left with index y_1 (or y_0) is called the y_1 (or y_0) node. The node on the right with index y_1 (or y_0) is called the y'_1 (or y'_0) node. The indices y_2 , y_1 and y_0 corresponds to the qubits q_2 , q_1 and q_0 . Then, ignoring the normalisation coefficients,

- y_0 node represents the quantum state $|y_0\rangle = |0\rangle + |1\rangle$, y_1 node represents the quantum state $|y_1\rangle = |0\rangle|y_0\rangle + \frac{1}{\sqrt{2}}|1\rangle(Z|y_0\rangle) = |00\rangle + |01\rangle + \frac{1}{\sqrt{2}}|10\rangle - \frac{1}{\sqrt{2}}|11\rangle$.
- Similarly, the y'_0 node represents the quantum state $|y'_0\rangle = |0\rangle + \frac{1}{\sqrt{2}}|1\rangle$, y'_1 node represents the quantum state $|y'_1\rangle = |0\rangle|y'_0\rangle + |1\rangle(X|y'_0\rangle) = |00\rangle + \frac{1}{\sqrt{2}}|01\rangle + \frac{1}{\sqrt{2}}|10\rangle + |11\rangle$.
- Then the y_2 node represent the quantum state $|y_2\rangle = |0\rangle|y_1\rangle + |1\rangle(Z \otimes I|y'_1\rangle) = |000\rangle + |001\rangle + \frac{1}{\sqrt{2}}|010\rangle - \frac{1}{\sqrt{2}}|011\rangle + |100\rangle + \frac{1}{\sqrt{2}}|101\rangle - \frac{1}{\sqrt{2}}|110\rangle - |111\rangle$.
- Finally, the entire LimTDD represents the quantum state $\frac{1}{\sqrt{6}}Z \otimes I \otimes I|y_2\rangle = \frac{1}{\sqrt{6}}(|000\rangle + |001\rangle + \frac{1}{\sqrt{2}}|010\rangle - \frac{1}{\sqrt{2}}|011\rangle - |100\rangle - \frac{1}{\sqrt{2}}|101\rangle + \frac{1}{\sqrt{2}}|110\rangle + |111\rangle)$.

Definition 3 (Reduced paths): Let \mathcal{F} be a decision diagram. The *reduced diagram* of \mathcal{F} is obtained by merging the edges between

any two nodes in \mathcal{F} . We also call paths within this reduced diagram *reduced paths* of \mathcal{F} .

In the example provided, the y_1 node has identical 0-successor and 1-successor nodes, meaning its two outgoing edges are merged when determining the reduced paths. The same applies to the y'_1, y_0 , and y'_0 nodes. Consequently, the LimTDD features only 2 reduced paths, whereas the multiple-terminal ADD comprises 7 reduced paths.

In this paper, we employ a specific graphical representation for LimTDDs. A node in a LimTDD is uniquely determined by its index, its two successors, and the weights on the edges connecting to those successors. We denote this as:

$$\textcircled{v_0} \xleftarrow{w_0} \textcircled{v} \xrightarrow{w_1} \textcircled{v_1}.$$

Similarly, a LimTDD is uniquely determined by its root node and the weight on the incoming edge. This can be represented as:

$$\left(w_{\mathcal{F}}, \textcircled{v_0} \xleftarrow{w_0} \textcircled{v} \xrightarrow{w_1} \textcircled{v_1} \right),$$

or more succinctly as:

$$\xrightarrow{w_{\mathcal{F}}} \textcircled{v}.$$

IV. LIMTDD BASED QUANTUM STATE PREPARATION

In this section, we introduce our method for Quantum State Preparation. Our method is mainly based on the following three observations.

A. Basic constructions

Observation 1: Let $\mathcal{F} = \xrightarrow{\lambda O_n \otimes \dots \otimes O_1} \textcircled{v}$ be a LimTDD representing the quantum state $|\psi\rangle$. Then $|\psi\rangle = \lambda O_n \otimes \dots \otimes O_1 |v\rangle$. Applying the operator $(O_n \otimes \dots \otimes O_1)^\dagger$ to $|\psi\rangle$ can reduce it to $\lambda \cdot |v\rangle$, which can be represented using a LimTDD with the root node v and incoming edge weight λ , that is $\xrightarrow{\lambda} \textcircled{v}$. In other words, applying a $(O_n \otimes \dots \otimes O_1)^\dagger$ to the LimTDD (i.e., contracting each index y_i with an operator O_i) can cancel the operator on the incoming edge of the LimTDD.

Observation 2: Let $\mathcal{F} = \left(w, \textcircled{v_0} \xleftarrow{I} \textcircled{v} \xrightarrow{\lambda O_n \otimes \dots \otimes O_1} \textcircled{v_1} \right)$ be a LimTDD representing the quantum state $|\psi\rangle$. Suppose the weight of \mathcal{F} is a complex number w . Then $|\psi\rangle = w \cdot (|0\rangle |v_0\rangle + |1\rangle \otimes (\lambda O_n \otimes \dots \otimes O_1 |v_1\rangle))$. Applying the operator $(O_n \otimes \dots \otimes O_1)^\dagger$ controlled by the qubit corresponding to the root node of \mathcal{F} to $|\psi\rangle$ can reduce it to $w \cdot (|0\rangle |v_0\rangle + \lambda |1\rangle |v_1\rangle)$, which can be represented using the LimTDD $\left(w, \textcircled{v_0} \xleftarrow{I} \textcircled{v} \xrightarrow{\lambda} \textcircled{v_1} \right)$. In other words, the operator on the high-edge of the node has been cancelled.

Observation 3: Let $\mathcal{F} = \left(w, \textcircled{v_0} \xleftarrow{w_0} \textcircled{v} \xrightarrow{w_1} \textcircled{v_1} \right)$ be a LimTDD representing the quantum state $|\psi\rangle$. Suppose w, w_0, w_1 are complex numbers and $w_0 \neq 0$. Then $|\psi\rangle = w \cdot (w_0 |0\rangle + w_1 |1\rangle) |v_0\rangle$. Denote $c = w_1/w_0$. Applying the unitary operator $\frac{1}{\sqrt{1+|c|^2}} \begin{bmatrix} 1 & c^\dagger \\ -c & 1 \end{bmatrix}$ to $|\psi\rangle$ can reduce it to $w \cdot w_0 \cdot \sqrt{1+|c|^2} \cdot |0\rangle |v_0\rangle$, which can be represented using the LimTDD $\left(w \cdot w_0 \cdot \sqrt{1+|c|^2}, \textcircled{v_0} \xleftarrow{1} \textcircled{v} \xrightarrow{0} \textcircled{v_1} \right)$. In other words, the complex weights on the two outgoing edges of v have been reduced to $[1, 0]$, corresponding to $|0\rangle$.

B. Branch processing

The simplifications introduced above apply primarily to the root node. In this subsection, we extend these techniques to non-terminal nodes by using an ancilla qubit to selectively ‘‘close’’ parts of the decision diagram, thereby exposing and processing other nodes as if they were root nodes.

Definition 4 (Branch Condition): A non-terminal node is termed a branch node if its 0-successor and 1-successor are distinct. The branch condition for a node along a path is defined by the values of all branch nodes preceding it on that path.

Consider the LimTDD given in Fig. 2 (b). Here, node y_2 is a branch node. The branch condition for the y_1 node (the left one) is $|0\rangle_2$, while for the y'_1 node (the right one), it is $|1\rangle_2$.

Definition 5 (Open/Closed Node): Let \mathcal{F} be a LimTDD, representing the quantum state $|\psi_v\rangle = \sum_{k=0}^{2^n-1} v_k |k\rangle$. Introducing an ancilla qubit q_a and marking each computational basis state $|k\rangle$ with $|b_k\rangle_a$ gives the state $\sum_{k=0}^{2^n-1} v_k |b_k\rangle_a |k\rangle$, where $b_k \in \{0, 1\}$. A path, corresponding to the computational basis state $|k\rangle$, is called open (closed, resp.) by q_a , if $b_k = 1$ (0, resp.). A node v on a path is called open (closed, resp.) if all paths that consist of the prefix path up to this node, as well as any suffix path rooted at this node, are open (closed, resp.), i.e., marked as $|1\rangle_a$ ($|0\rangle_a$, resp.). A LimTDD is called open (closed, resp.) if its root node is open (closed, resp.).

With these definitions, we can now use the ancilla qubit to manipulate the LimTDD. Suppose a node v and its successors are the only open nodes. We use q_a as a control qubit to perform operations on v without affecting other parts of the decision diagram.

For instance, consider the LimTDD $\mathcal{F} = \left(w, \textcircled{v_0} \xleftarrow{I} \textcircled{v} \xrightarrow{w_1} \textcircled{v_1} \right)$, representing the quantum state $|\psi\rangle = w \cdot (|0\rangle_b |v_0\rangle + \lambda |1\rangle_b |v_1\rangle)$. Let q_b and q_c be the qubits corresponding to nodes v and v_0 , respectively. By setting the ancilla qubit q_a to $|1\rangle_a$, the entire LimTDD is initially open. Applying a CX gate controlled by $|1\rangle_b$ yields the state $w \cdot (|1\rangle_a |0\rangle_b |v_0\rangle + \lambda |0\rangle_a |1\rangle_b |v_1\rangle)$, effectively closing node v_1 and opening node v_0 .

We can then use $|1\rangle_a$ to control further operations on v_0 . Suppose $\textcircled{v_0} \xleftarrow{I} \textcircled{v_0} \xrightarrow{\lambda O_n \otimes \dots \otimes O_1} \textcircled{v_0}$, such that $|v_0\rangle = |0\rangle_c |v_{00}\rangle + \lambda |1\rangle_c (O_n \otimes \dots \otimes O_1 |v_{01}\rangle)$. Applying an $(O_n \otimes \dots \otimes O_1)^\dagger$ gate controlled by $|1\rangle_a |1\rangle_c$ transforms the state to $w \cdot (|1\rangle_a |0\rangle |v'_0\rangle + \lambda |0\rangle_a |1\rangle |v_1\rangle)$, where $|v'_0\rangle = |0\rangle |v_{00}\rangle + \lambda |1\rangle |v_{01}\rangle$, depicted as $\textcircled{v_0} \xleftarrow{I} \textcircled{v'_0} \xrightarrow{\lambda} \textcircled{v_0}$.

After processing the v_0 branch, applying an X gate to q_a switches the state to $w \cdot (|0\rangle_a |0\rangle_b |v'_0\rangle + \lambda |1\rangle_a |1\rangle_b |v_1\rangle)$, closing node v_0 and opening node v_1 for subsequent processing. To ensure the ancilla qubit q_a is restored to its initial state, we apply a CX gate with the control condition $|0\rangle_b$. This operation changes the state to $w \cdot |1\rangle_a (|0\rangle_b |v'_0\rangle + \lambda |1\rangle_b |v'_1\rangle)$, thus restoring q_a to $|1\rangle_a$. It should be noted that if the nodes v_0, v_1 , and their subsequent nodes have all been fully processed, the state should have the form $w \cdot |1\rangle_a (w_0 |0\rangle_b + w_1 |1\rangle_b) |0\rangle^{\otimes k}$, for some k . At this time, Observation 3 can be applied to reduce the weights w_0 and w_1 and return the qubit q_b to $|0\rangle$.

C. Algorithm

In our algorithm (pseudocode described in Algorithm 1), we first cancel the operator on the incoming edge of the root node. Then, we enter a recursive process. For every node, we first cancel the operator on its high-edge, then process its 0-successor, followed by its 1-successor, and finally adjust the weights on the outgoing edges of the current node.

During this process, we use an auxiliary qubit to mark the current node v being processed, i.e., the open part of the decision diagram. Whenever we encounter a branch node, we first use a multi-controlled X gate to close its 1-successor. The control condition should be set to the branch condition of $\text{high}(v)$. Then we process the 0-successor. After that, we flip the open/close condition of the two successors with the branch condition of v and process the 1-successor. Finally, we

open the 0-successor using the branch condition of $\text{low}(v)$, making the original node open. Then we backtrack upwards. Note that when we finish the process, the root node will be open, meaning that the ancilla qubit has been returned to the $|1\rangle_a$ state. The circuit required to prepare the quantum state is the inverse of the circuit obtained.

Algorithm 1 STATEPRE(v)

Input: A node v of an LimTDD representing an n -qubit quantum state $|\psi\rangle$.

Output: A quantum circuit C , corresponding to an unitary matrix U , such that $U|1\rangle_a|\psi\rangle = |1\rangle_a|0\rangle^{\otimes n}$.

```

1:  $cir \leftarrow \text{QuantumCircuit}(n+1)$   $\triangleright$  An empty quantum circuit
   with  $n+1$  qubits
2: if  $v$  is the terminal node then
3:   return ( $cir, 1$ )
4: end if
5: Suppose  $\text{wt}((v, \text{high}(v))) = \lambda \cdot O$ 
6: Append  $cir$  with a controlled  $O^\dagger$  gate, with the control condition
   set to be  $|1\rangle_a$   $\triangleright$  Reduce the operator on the high-edge of  $v$ 
7: if  $\text{low}(v) = \text{high}(v)$  then
8:    $cir_0, w_0 \leftarrow \text{STATEPRE}(\text{low}(v))$ 
9:   Append  $cir$  with  $cir_0$ 
10:   $w_1 \leftarrow w_0$ 
11: else
12:  Use the branch condition of  $\text{high}(v)$  to close the high-
   branch of the node
13:   $cir_0, w_0 \leftarrow \text{STATEPRE}(\text{low}(v))$ 
14:  Append  $cir$  with  $cir_0$ 
15:  Use the branch condition of  $v$  to close the low-branch and
   open the high-branch
16:   $cir_1, w_1 \leftarrow \text{STATEPRE}(\text{high}(v))$ 
17:  Append  $cir$  with  $cir_1$ 
18:  Use the branch condition of  $\text{low}(v)$  to open the low-branch
19: end if
20:  $w_1 \leftarrow \lambda \cdot w_1$ 
21:  $c \leftarrow w_1/w_0$ 
22: Append a controlled  $\frac{1}{\sqrt{1+|c|^2}} \begin{bmatrix} 1 & c^\dagger \\ -c & 1 \end{bmatrix}$ , with the control
   condition  $|1\rangle_a$  and target  $q_v$ 
23: return ( $cir, w_0 \cdot \sqrt{1+|c|^2}$ )

```

V. AN EXAMPLE

In this section, we provide a detailed example to illustrate the procedure of our algorithm. We give a step by step demonstration of our algorithm on the LimTDD given in Fig. 2, with the quantum state to be prepared being $\frac{1}{\sqrt{6}}(|000\rangle + |001\rangle + \frac{1}{\sqrt{2}}|010\rangle - \frac{1}{\sqrt{2}}|011\rangle - |100\rangle - \frac{1}{\sqrt{2}}|101\rangle + \frac{1}{\sqrt{2}}|110\rangle + |111\rangle)$.

A. Initial Setup

- 1) **Quantum State Representation:** The quantum state represented by the LimTDD is: $\frac{1}{\sqrt{6}}Z \otimes I \otimes I |y_2\rangle$ where: $|y_2\rangle = |0\rangle|y_1\rangle + |1\rangle(Z \otimes I |y_1'\rangle)$ $|y_1\rangle = |0\rangle|y_0\rangle + \frac{1}{\sqrt{2}}|1\rangle(Z |y_0\rangle)$ $|y_1'\rangle = |0\rangle|y_0'\rangle + |1\rangle(X |y_0'\rangle)$ $|y_0\rangle = |0\rangle + |1\rangle|y_0'\rangle = |0\rangle + \frac{1}{\sqrt{2}}|1\rangle$.
- 2) **Ancilla Qubit:** We add an ancilla qubit $|1\rangle_a$ to the system, resulting in the initial state: $\frac{1}{\sqrt{6}}|1\rangle_a(Z \otimes I \otimes I |y_2\rangle)$.

B. Step-by-Step Reduction

- 1) **Cancel the Operator on the Incoming Edge:**

- Apply a Z gate on qubit q_2 to cancel the $Z \otimes I \otimes I$ operator on the incoming edge of the LimTDD. The state becomes: $|1\rangle_a |y_2\rangle$.

- 2) **Process the High-Edge of y_2 Node:**

- Apply a CCZ gate with q_a and q_2 as controls and q_1 as the target to cancel the $Z \otimes I$ operator on the high-edge of the y_2 node. The state becomes: $|1\rangle_a (|0\rangle|y_1\rangle + |1\rangle|y_1'\rangle)$.

- 3) **Process the y_1 Node:**

- Use a CX gate with q_2 as the control qubit and q_a as the target qubit to close the y_1' node. The state becomes: $|1\rangle_a |0\rangle|y_1\rangle + |0\rangle_a |1\rangle|y_1'\rangle$.
- Apply a CCZ gate with q_a and q_1 as controls and q_0 as the target to cancel the Z operator on the high-edge of the y_1 node. The state becomes: $|1\rangle_a |0\rangle(|0\rangle + \frac{1}{\sqrt{2}}|1\rangle)|y_0\rangle + |0\rangle_a |1\rangle|y_1'\rangle$.

- 4) **Process the y_0 Node and Adjust the Weights on Outgoing Edges of the y_1 Node:**

- Since $|y_0\rangle = |0\rangle + |1\rangle$, apply a controlled- U gate with q_a as the control qubit to transform the state to: $|1\rangle_a |0\rangle(\sqrt{2}|0\rangle + |1\rangle)|0\rangle + |0\rangle_a |1\rangle|y_1'\rangle$ where $U = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$.
- Apply a controlled- V gate with q_a as the control qubit to adjust the weights on the outgoing edges of the y_1 node and change the state to: $\sqrt{3}|1\rangle_a |0\rangle|0\rangle|0\rangle + |0\rangle_a |1\rangle|y_1'\rangle$ where $V = \frac{\sqrt{2}}{\sqrt{3}} \begin{bmatrix} 1 & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & 1 \end{bmatrix}$.

- 5) **Process the y_1' Node:**

- Use an X gate on q_a to switch the branches and open the y_1' node. The state becomes: $\sqrt{3}|0\rangle_a |0\rangle|0\rangle|0\rangle + |1\rangle_a |1\rangle|y_1'\rangle$.
- Apply a CCX gate controlled by q_a and q_1 to cancel the operator on the high-edge of the y_1' node. The state becomes: $\sqrt{3}|0\rangle_a |0\rangle|0\rangle|0\rangle + \sqrt{3}|1\rangle_a |1\rangle(|0\rangle + |1\rangle)|y_0'\rangle$.

- 6) **Process the y_0' Node and Adjust the Weights on Outgoing Edges of the y_1' Node:**

- Apply a controlled- V gate with q_a as the control qubit, the state will be changed to $\sqrt{3}|0\rangle_a |0\rangle|0\rangle|0\rangle + |1\rangle_a |1\rangle(\frac{\sqrt{3}}{\sqrt{2}}|0\rangle + \frac{\sqrt{3}}{\sqrt{2}}|1\rangle)|0\rangle$.
- Apply a further gate U controlled by $|1\rangle_a$ and change the state to $\sqrt{3}|0\rangle_a |0\rangle|0\rangle|0\rangle + \sqrt{3}|1\rangle_a |1\rangle|0\rangle|0\rangle$.

- 7) **Adjust the Weights on Outgoing Edges of the y_2 Node:**

- Apply a CX gate with the control qubit q_2 set to be $|0\rangle$ and target qubit q_a , to open the y_1 node, thus making the all branches of y_2 node open, and the state becomes: $\sqrt{3}|1\rangle_a (|0\rangle + |1\rangle)|0\rangle|0\rangle$.
- Use a controlled- U gate with q_a as the control qubit to adjust the weights on the outgoing edges of the y_2 node, the state becomes: $\sqrt{6}|1\rangle_a |0\rangle|0\rangle|0\rangle$. Note that the coefficient $\sqrt{6}$ is cancelled with the ignored coefficient $\frac{1}{\sqrt{6}}$.

C. Resulting Quantum Circuit

The resulting quantum circuit for preparing the desired quantum state is shown in Fig. 3, which includes the following gates:

- Z gates to cancel the operators on the incoming edges.
- CCZ and CCX gates to cancel the operators on the high-edges of the nodes.
- Controlled- U and controlled- V gates to adjust the weights on the edges.

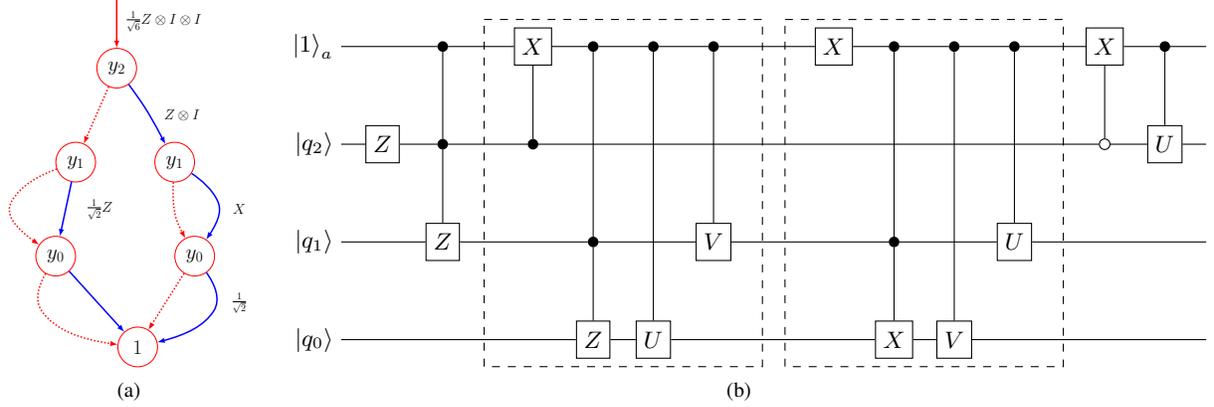


Fig. 3. The quantum circuit that transfers the quantum state represented by the LimTDD shown in Fig. 2 to $|000\rangle$. The two dotted boxes correspond to the processing of the y_1 and y'_1 nodes, respectively. In this circuit $U = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$, $V = \frac{\sqrt{2}}{\sqrt{3}} \begin{bmatrix} 1 & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & 1 \end{bmatrix}$.

- Ancilla qubit q_a used to control the processing of different branches.

VI. COMPLEXITY

A. Time complexity

In our algorithm, we first process the operator on the incoming edge of the root node of the LimTDD, then traverse the decision diagram starting from the root node and subsequently handle the operator on the high-edge of each node. When we encounter a branch node (that is, a node whose 0- and 1-successors are different), we use an auxiliary qubit to mark and close its 1-successor, then proceed to process the 0-successor. After completing the processing of the 0-successor, we return to the branch node, close the 0-successor, and open the 1-successor for processing. If the current node is not a branch node, we simply process its successors without closing half of the decision diagram. This process continues recursively.

Upon reaching the terminal node, we perform a reverse pass, processing the complex weights on each node's edges in a bottom-up manner. After reaching the nearest branch node, we wait for the processing of branch 1 to complete, then process the complex weight on the edge of the branch node, and subsequently backtrack upwards. Eventually, all operators and complex weights on the branches will have been resolved.

To summarise, we traverse the decision graph in a depth-first manner, with each reduced path being traversed exactly once. Note that there are n non-terminal nodes on a path (where n is the number of qubits in the system), each requiring handling of the high-edge operator and outgoing weight. Assuming there are p reduced paths in the LimTDD, the time complexity of our algorithm is $\mathcal{O}(np)$.

B. Gate Complexity

We begin by analysing the operators appearing on the edges of the LimTDD. There are no more than $(n-1) + \dots + 2 + 1 = \frac{n(n-1)}{2}$ local operators on each path, and these operators will be eliminated through 3-qubit controlled gates. Thus, we need $\mathcal{O}(n^2p)$ 3-qubit gates to eliminate all these operators. Note that since the control qubits to cancel the local operators on an edge are all the same, these 3-qubit gates can be reduced to two 3-qubit gates and a series of 2-qubit gates if a further ancilla qubit is given. In addition, we need no more

than n single-qubit gates to eliminate the operator on the incoming edge of the LimTDD.

Next, we examine the weights on the outgoing edges of each node. As each node on each path requires an operator, and each operator requires a control qubit, we need $\mathcal{O}(np)$ 2-qubit gates to eliminate them.

Finally, we examine the quantum gates used to control the opening and closing of branches. For each branch node, we need one controlled gate to close its 1-branch, one controlled gate to flip the open/close status of both branches, and one controlled gate to reopen the 0-branch. Assuming there are k branch nodes preceding the current node on the path, this requires two $(k+1)$ -qubit controlled gates and one k -qubit controlled gate.

In summary, the total gate requirements are:

- $\mathcal{O}(p)$ s -qubit gates for $s > 3$,
- $\mathcal{O}(n^2p)$ 3-qubit gates,
- $\mathcal{O}(np)$ 2-qubit gates,
- $\mathcal{O}(n)$ single-qubit gates.

Special Case: For decision diagrams in the tower form (where the 0- and 1-successors of each non-terminal node are the same), there is only one path involving different nodes. In this scenario, the ancilla qubit can be omitted since it remains in the $|1\rangle$ state. Consequently, only $\mathcal{O}(n^2)$ 2-qubit gates and $\mathcal{O}(n)$ single-qubit gates are required to prepare the quantum state.

VII. EXPERIMENTS

In this section, we evaluate the performance of our LimTDD-based QSP algorithm and compare it with existing methods, including the ADD-based method [21], Qiskit [24], and QuICT [25]. The algorithms used for quantum state preparation in Qiskit and QuICT are established in [11] and [12], respectively. The experiments focus on two main metrics: gate complexity and runtime complexity.

A. Experimental Setup

- **Hardware and Software:**

- Our Method, Qiskit, and QuICT: Experiments are conducted on a desktop with an 11th Gen Intel Core(TM) i7-11700F CPU and 16GB RAM. All methods are implemented using the Python programming language.

- ADD-based method: Since the implementation of [21] is in C++ and only supports a Linux environment, these experiments are conducted on a Linux server with a 13th Gen Intel(R) Core(TM) i5-13600KF and 32GB RAM.
- **Quantum States:**
 - We generate random quantum states using Clifford + T circuits, which are one of the most commonly used circuit categories. These states serve as the target states for the QSP task.
 - For each qubit number n , we generate 20 random quantum states and calculate the average performance metrics.
- **Metrics:**
 - Gate Complexity: The number of multi-qubit gates required to prepare the target state. The output circuit of Qiskit and QuICT consists of single-qubit gates and CX gate. For our method, we also compile the generated circuit into single-qubit gates and CX gate using Qiskit and count its CX gate count (denoted as “Transpiled”). For the ADD-based method, we used the method given in [21] (integrated in their tool) to estimate the number of CX gates that needed to implement the multi-qubit gates (denoted as “Expanded”).
 - Runtime Complexity: The execution time required to generate the quantum circuit for state preparation. The runtime of LimTDD (Transpiled) also includes the transpilation time.

B. Comparison with ADD-based Method

We first compare our LimTDD-based method with the ADD-based method proposed in [21], which also focuses on decision-diagram-based efficient state preparation. The runtime and the multi-qubit gates complexity of the two methods are demonstrated in Fig. 4.

- **Gate Complexity:** We compare the number of gates required by our LimTDD-based method and the ADD-based method from [21]. The results show that our method consistently requires fewer gates when the number of qubits is larger than 5. This is due to the more compact representation of quantum states using LimTDD compared to ADD. For example, for $n = 15$ qubits, our method uses approximately 100 and 700 gates before and after the transpilation, while the ADD-based method uses around 3500 and 80000 gates before and after the expansion.
- **Runtime Complexity:** When the number of qubits is small, the ADD-based method is faster than our method. The main reason could be that the ADD-based method is implemented in C++, while our method is implemented in Python. But when the number of qubits grows larger, the runtime of our method becomes significantly shorter. At this point, the advantage of the compactness of LimTDD begins to manifest.

C. Comparison with Qiskit and QuICT

We further compare our LimTDD-based method with two widely-used quantum computing frameworks: Qiskit [24] and QuICT [25]. Fig. 5 illustrates the detailed comparison. The overall result is consistent with that of ADD-based method. We note here that no ancilla qubits are used by Qiskit and QuICT.

- **Gate Complexity:** We compare our method with Qiskit and QuICT in terms of the number of multi-qubit gates required. Our method consistently achieves lower gate counts when the number of qubits is larger than 7, demonstrating its efficiency in QSP. For instance, for $n = 15$ qubits, our method requires

around 100 and 700 gates before and after the transpilation, while both Qiskit and QuICT require around 130000 gates.

- **Runtime Complexity:** The runtime of our method is significantly shorter compared to Qiskit and QuICT, when the number of qubits grows larger than 10. For $n = 12$ qubits, our method completes in an average of 0.5 seconds, while Qiskit and QuICT take approximately 3 and 2 seconds, respectively. In addition, Qiskit and QuICT run out of time (> 600 seconds) when the qubit number is larger than 17 and 21, respectively.

D. Discussion

To highlight the exponential advantage of our method, we provide a simple example that demonstrates the superiority of our LimTDD-based algorithm over the ADD-based method presented in [21].

Consider a quantum circuit extended from the one shown in Fig. 1. Suppose n is the number of qubits, starting from the state $|0\rangle^{\otimes n}$. Apply a Hadamard gate (H) to each qubit, and then apply a series of controlled- Z (CZ) gates between qubit q_k and qubits q_0, \dots, q_{k-1} for $1 \leq k \leq n-1$. Denote the resulting quantum state as $|v_n\rangle$. Then,

$$|v_n\rangle = \frac{1}{\sqrt{2}} |0\rangle |v_{n-1}\rangle + \frac{1}{\sqrt{2}} |1\rangle (Z \otimes \dots \otimes Z |v_{n-1}\rangle).$$

For all $k = n$ down to $k = 2$, the first part (corresponding to $|0\rangle$) and the second part (corresponding to $|1\rangle$) of the state are different. Consequently, the ADD representation of the state has an exponential number of paths, while the LimTDD representation has only $n + 1$ nodes, with all the operators $Z \otimes \dots \otimes Z$ appearing on the high edge of every non-terminal node, and one path.

Using our algorithm to generate a circuit for preparing the state $|v_n\rangle$ results in a similar circuit as shown in Fig. 1, requiring $\mathcal{O}(n^2)$ gates. An ancilla qubit q_a is added, and all gates become controlled versions with q_a as the control qubit. Since the state of q_a remains in $|1\rangle$ and never changes, a simple optimisation can restore the circuit. In contrast, the exponential number of paths in the ADD representation necessitates an exponential number of gates for state preparation using the method of [21]. In fact, an exponential number of quantum gates is also needed for Qiskit and QuICT to prepare this quantum state.

This example underscores the potential of LimTDD to handle complex quantum states more efficiently, providing a significant advantage in certain scenarios.

VIII. CONCLUSION

This paper presents a novel quantum state preparation algorithm using LimTDD, which achieves significant improvements in efficiency and circuit complexity compared to existing methods. The compactness of LimTDD allows for efficient handling of large-scale quantum states, with experimental results demonstrating exponential efficiency gains in the best-case scenario. This work highlights the potential of LimTDD in quantum state preparation and provides a robust foundation for future quantum computing technologies. Future work may focus on further optimising LimTDD and exploring its applications in other quantum computing areas.

REFERENCES

- [1] P. W. Shor, “Algorithms for quantum computation: discrete logarithms and factoring,” in *In Proceedings 35th annual symposium on foundations of computer science*, 1994, pp. 124–134.
- [2] A. Kovyshin, L. Tornberg, J. Crain, S. Mensa, I. Tavernelli, and A. Broo, “Prioritizing quantum computing use cases in the drug discovery and development pipeline,” *Drug Discovery Today*, vol. 30, p. 104323, 2025.
- [3] H. Gharibyan, “Quantum leap in material science,” *Tech Briefs*, March 2024.

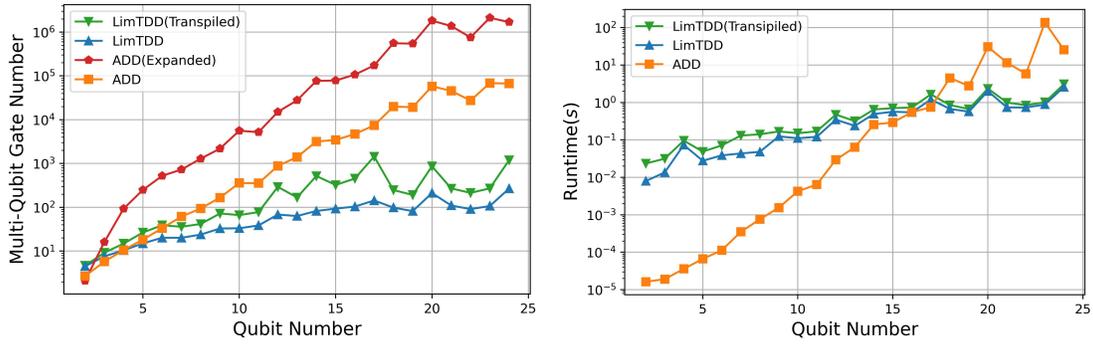


Fig. 4. Experiment results of our method against ADD-based method [21].

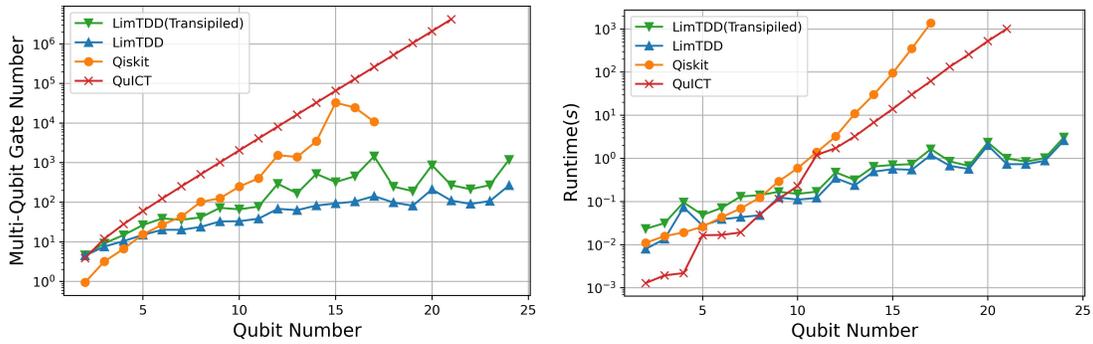


Fig. 5. Experiment results of our method against Qiskit [24] and QuICT [25].

- [4] H. S. M. Gschwendtner and S. Zingg, "Combining quantum and ai for the next superpower," in *Nature Reviews Electrical Engineering*, 1994, pp. 350–351.
- [5] A. Sidford and C. Zhang, "Quantum speedups for stochastic optimization," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NeurIPS, 2024, pp. 35 300–35 330.
- [6] 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.
- [7] R. Mao, G. Tian, and X. Sun, "Toward optimal circuit size for sparse quantum state preparation," *Physical Review A*, vol. 110, no. 3, p. 032439, 2024.
- [8] D. Ramacciotti, A. I. Lefterovici, and A. F. Rotundo, "Simple quantum algorithm to efficiently prepare sparse states," *Phys. Rev. A*, vol. 110, p. 032609, Sep 2024. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevA.110.032609>
- [9] M. Plesch and C. Brukner, "Quantum-state preparation with universal gate decompositions," *Physical Review A*, vol. 83, no. 3, p. 032302, Mar 2011.
- [10] V. Shende, S. S. Bullock, S. S. Bullock, and I. L. Markov, "Synthesis of quantum-logic circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2006.
- [11] R. Iten, R. Colbeck, I. Kukuljan, J. Home, and M. Christandl, "Quantum circuits for isometries," *Physical Review A*, Mar 2016. [Online]. Available: <http://dx.doi.org/10.1103/physreva.93.032318>
- [12] M. Mottonen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa, "Transformation of quantum states using uniformly controlled rotations," *arXiv preprint quant-ph/0407010*, 2004.
- [13] V. Bergholm, J. J. Vartiainen, M. Mottonen, and M. M. Salomaa, "Quantum circuits with uniformly controlled one-qubit gates," *Physical Review A*, vol. 71, no. 5, p. 052330, May 2005.
- [14] X.-M. Zhang, T. Li, and X. Yuan, "Quantum state preparation with optimal circuit depth: Implementations and applications," *Physical Review Letters*, vol. 129, no. 23, p. 230504, 2022.
- [15] X. Sun, G. Tian, S. Yang, P. Yuan, and S. Zhang, "Asymptotically optimal circuit depth for quantum state preparation and general unitary synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 11, pp. 3301–3314, 2023.
- [16] R. Wille and R. Drechsler, "Bdd-based synthesis of reversible logic for large functions," in *2009 46th ACM/IEEE Design Automation Conference*, 2009, pp. 270–275.
- [17] R. Drechsler, "Polynomial circuit verification using bdds," in *2021 5th International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques (ICEECCOT)*, 2021, pp. 49–52.
- [18] L. Burgholzer and R. Wille, "QCEC: A JKQ tool for quantum circuit equivalence checking," *Software Impacts*, vol. 7, p. 100051, 2021.
- [19] Q. Zhang, M. Saligane, H.-S. Kim, D. Blaauw, G. Tzimpragos, and D. Sylvester, "Quantum circuit simulation with fast tensor decision diagram," in *2024 25th International Symposium on Quality Electronic Design (ISQED)*, 2024, pp. 1–8.
- [20] F. Mozafari, M. Soeken, H. Rieneer, and G. De Micheli, "Automatic uniform quantum state preparation using decision diagrams," in *2020 IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL)*. IEEE, 2020.
- [21] F. Mozafari, G. De Micheli, and Y. Yang, "Efficient deterministic preparation of quantum states using decision diagrams," *Phys. Rev. A*, vol. 106, p. 022617, Aug 2022. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevA.106.022617>
- [22] Y. Tanaka, H. Yamasaki, and M. Murao, "Quantum state preparation via free binary decision diagram," 2024. [Online]. Available: <https://arxiv.org/abs/2407.01671>
- [23] X. Hong, A. Dai, D. Gao, S. Li, Z. Ji, and M. Ying, "LimTDD: A Compact Decision Diagram Integrating Tensor and Local Invertible Map Representations," Apr. 2025, arXiv:2504.01168 [cs]. [Online]. Available: <http://arxiv.org/abs/2504.01168>
- [24] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, "Quantum computing with Qiskit," 2024.
- [25] Institute of Computing Technology, Chinese Academy of Sciences, "QuICT: Quantum Computer of Institute of Institute of Computing Technology," <https://quict-docs.readthedocs.io/aa/latest/>, 2023.