Exploiting Arbitrary Paths for the Simulation of Quantum Circuits with Decision Diagrams

Lukas Burgholzer* Alexander Ploier* Robert Wille*†
*Institute for Integrated Circuits, Johannes Kepler University Linz, Austria
†Software Competence Center Hagenberg GmbH (SCCH), Austria
lukas.burgholzer@jku.at alexander.ploier@jku.at robert.wille@jku.at
https://iic.jku.at/eda/research/quantum/

***Abstract*—The classical simulation of quantum circuits is essential in the development and testing of quantum algorithms. Methods based on tensor networks or decision diagrams have proven to alleviate the inevitable exponential growth of the underlying complexity in many cases. But the complexity of these methods is very sensitive to so-called contraction plans or simulation paths, respectively, which define the order in which respective operations are applied. While, for tensor networks, a plethora of strategies has been developed, simulation based on decision diagrams is mostly conducted in a straight-forward fashion thus far. In this work, we envision a flow that allows to translate strategies from the domain of tensor networks to decision diagrams. Preliminary results indicate that a substantial advantage may be gained by employing suitable simulation paths—motivating a thorough consideration.**

I. Introduction

The current costs, reliability, and availability of quantum computers often prohibits the evaluation of possible applications or prototypes thereof on the actual hardware. As a result, classical simulations of quantum computations are essential in the rapid development and testing of quantum algorithms and applications. Such simulations entail computing the evolution of a quantum state under the application of the sequence of operations described by a quantum circuit. Conceptually, this can be translated to the problem of conducting a sequence of matrix-matrix and matrix-vector multiplications. However, since the underlying representations of the state of a quantum system and the operations to be applied grow exponentially with respect to the number of qubits, classical simulation of quantum circuits quickly becomes a challenging task. Besides using powerful supercomputing clusters [1]–[3], clever data structures such as tensor networks [4]–[6], or decision diagrams [7]–[9] have been demonstrated to alleviate this complexity in many practically relevant cases.

The tensor network corresponding to a quantum circuit is formed by representing each gate of a quantum circuit, as well as the initial quantum state, as a tensor which is connected to the other tensors via shared indices. Classically simulating the circuit then translates to contracting the respective tensor network into a single tensor. But, in general, the complexity of tensor network-based simulation is extremely sensitive to the order in which the individual tensors are contracted. Accordingly, a plethora of methods have been proposed to determine corresponding contraction paths [10]–[13]—a task proven to be NP-hard [14].

Decision diagrams, on the other hand, take a more structural approach in the sense that they try exploit redundancies in the representations of quantum states and operations in order to allow for a more compact representation and also efficient manipulation in many cases. To this end, they represent these quantities as directed, acyclic graphs with complex edge weights. Similar to tensor networks, the initial quantum state and each gate of a quantum circuit are first translated to their (typically linearly-sized) decision diagram representation. Simulation then amounts to multiplying the respective decision diagrams until the final state vector representation is obtained.

Therein, the complexity of multiplying decision diagrams scales with the product of their sizes, i.e., their number of nodes. Whenever the respective intermediate decision diagrams remain rather compact, an efficient scheme for classical simulation is obtained. But this compactness and, by this, the complexity of decision diagram-based quantum circuit simulation is very sensitive to the order in which the individual multiplications are performed (called a *simulation path* in the following). In contrast to the contractions of tensor networks, this effect has hardly been studied yet.

In this work, we investigate this issue and envision a flow that allows to employ existing techniques from the tensor network domain also for decision diagrams. Preliminary results demonstrate the potential for simulations based on decision diagrams, showing substantial advantages over the state of the art.

The rest of this paper is structured as follows: Section II introduces the necessary background on quantum circuit simulation and decision diagrams. Then, Section III illustrates the considered problem and reviews the related work. Section IV shows how simulation paths can be handled in general and describes the flow for using existing techniques from the tensor network domain. Afterwards, Section V summarizes initial results, before Section VI concludes the paper.

II. Background

To keep this paper self-contained, this section briefly covers the basics on quantum circuit simulation followed by a brief review of decision diagrams—which provide the basis of the simulation approach considered in the rest of this work.

A. Quantum Circuit Simulation

A quantum state $|\varphi\rangle$ of an n-qubit quantum system can be described as a linear combination of 2^n basis states, i.e.,

$$|\varphi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i\,|i\rangle \text{ with } \alpha_i \in \mathbb{C} \text{ and } \sum_{i\in\{0,1\}^n} |\alpha_i|^2 = 1.$$

This state is commonly represented as a vector $[\alpha_{0\ldots0}, \ldots, \alpha_{1\ldots1}]^\top$, referred to as *state vector*. Measuring this state leads to a collapse of the system's state to one of the basis states $|i\rangle$ each with probability $|\alpha_i|^2$ for $i \in \{0,1\}^n$.

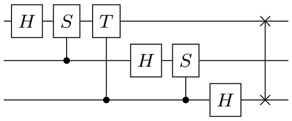


Fig. 1. Circuit for the 3-qubit quantum Fourier transform

In the following, we will always identify $|\varphi\rangle$ with its corresponding state vector, i.e., $|\varphi\rangle \equiv [\alpha_{0\ldots0}, \ldots, \alpha_{1\ldots1}]^\top$.

Example 1. *An important example of a quantum state is the 3-qubit Greenberger–Horne–Zeilinger or GHZ state [15]:*

$$|GHZ\rangle = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) \equiv \frac{1}{\sqrt{2}}[1, 0, 0, 0, 0, 0, 0, 1]^\top$$

The state of any quantum system can be manipulated by quantum operations, also called *quantum gates*. Any such gate g acting on k qubits can be identified by a unitary matrix U of size $2^k \times 2^k$, i.e., $g \equiv U$. Its action on a quantum state corresponds to the matrix-vector product of the matrix with the vector representing the state[1], i.e., $|\varphi'\rangle = U\,|\varphi\rangle$.

A *quantum circuit* is now described as a composition of quantum gates. Consequently, the evolution of an initial quantum state $|\varphi\rangle$ through a quantum circuit $G = g_1 \ldots g_{|G|}$ is described by the subsequent application of the individual gates to this initial state, i.e.,

$$|\varphi\rangle\, G = |\varphi\rangle\, g_1 \ldots g_{|G|} \equiv U_{|G|} * \cdots * U_1 * |\varphi\rangle\,.$$

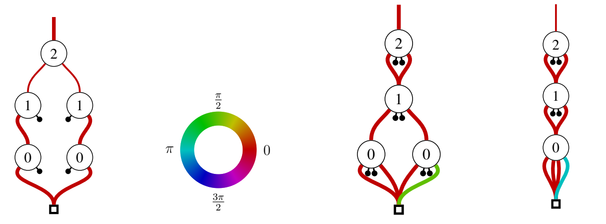
If this task is conducted on a classical computer, it is commonly referred to as *quantum circuit simulation*.

Example 2. *Fig. 1 shows a 3-qubit quantum circuit realizing an important quantum algorithm, namely the quantum analog to the Fourier transform. It consists of three single-qubit Hadamard gates (indicated by boxes labeled H), three two-qubit controlled-phase rotations (indicated by boxes labeled S and T connected to ●), and a two-qubit SWAP gate (indicated by ×). Given an initial state in the computational basis, this circuit outputs the state's representation in the Fourier basis.*

B. Decision Diagrams

The representations of a quantum system state and the operations manipulating it are exponentially large w.r.t. the number of qubits involved. This quickly limits straight-forward approaches for representing (and manipulating) even moderately sized state vectors, such as arrays, without resorting to supercomputing clusters. For example, representing the dense state vector of a 32-qubit system already requires 64 GiB of memory (assuming 128 bit complex numbers).

Decision diagrams [7]–[9] have been proposed as a complementary approach for efficiently representing and manipulating quantum states by exploiting redundancies in the underlying representation. A decision diagram representing a quantum state (or operation) is a directed, acyclic graph with complex edge weights. To this end, a given state vector with its complex



(a) $|GHZ\rangle$ (b) Color wheel (c) Controlled-S gate (d) H gate

Fig. 2. Decision diagrams for 3-qubit states and gates

amplitudes α_i for $i \in \{0,1\}^n$ is recursively decomposed into sub-vectors according to

$$\begin{array}{cc} \multicolumn{2}{c}{[\alpha_{0\ldots0}, \ldots, \alpha_{1\ldots1}]^\top} \\ [\alpha_{0x}]^\top & [\alpha_{1x}]^\top \\ [\alpha_{00y}]^\top \quad [\alpha_{01y}]^\top & [\alpha_{10y}]^\top \quad [\alpha_{11y}]^\top \end{array}$$

with $x \in \{0,1\}^{n-1}$ and $y \in \{0,1\}^{n-2}$, until only individual amplitudes remain. The resulting graph has n levels of nodes, labelled $n-1$ down to 0. Each node i has exactly two successors indicating whether the path leads to an amplitude where qubit i is in state $|0\rangle$ or $|1\rangle$.

By extracting common factors into edge weights (and employing suitable normalization schemes, see [7], [9]), any two sub-vectors that only differ by a constant factor can be unified and need not be represented by separate nodes in the decision diagram. Exploiting such redundancies frequently allows to obtain rather compact representations (in the best case linear with respect to the number of nodes) for the, in general, exponentially large state vectors.

Example 3. *Fig. 2a shows a graphical representation (proposed in [16]) of the decision diagram for the GHZ state considered previously in Example 1. To this end, the thickness of an edge indicates the weight of the corresponding magnitude, while the color wheel shown in Fig. 2b is used to encode its phase. Furthermore, edges with a weight of 0 are denoted as ●-stubs. In general, the decision diagram for an n-qubit GHZ state requires $2n-1$ nodes for representing the 2^n-dimensional state vector.*

Decision diagram representations for quantum gates are obtained by extending the decomposition scheme for state vectors by a second dimension. This corresponds to recursively splitting the respective matrix into four equally-sized sub-matrices according to the basis $\left\{\left[\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right], \left[\begin{smallmatrix}0&1\\0&0\end{smallmatrix}\right], \left[\begin{smallmatrix}0&0\\1&0\end{smallmatrix}\right], \left[\begin{smallmatrix}0&0\\0&1\end{smallmatrix}\right]\right\}$.

Example 4. *Consider again the circuit shown in Example 2. Then, Fig. 2c and Fig. 2d exemplarily show the decision diagram representations for the $2^3 \times 2^3$ matrices of the controlled-S and the Hadamard gate at the end of the circuit, respectively.*

As described above, applying a gate to a quantum system entails the matrix-vector multiplication of the corresponding matrix with the current state vector. This operation can be recursively broken down according to

$$\begin{bmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{bmatrix} \cdot \begin{bmatrix} \alpha_{0\ldots} \\ \alpha_{1\ldots} \end{bmatrix} = \begin{bmatrix} (U_{00} \cdot \alpha_{0\ldots} + U_{01} \cdot \alpha_{1\ldots}) \\ (U_{10} \cdot \alpha_{0\ldots} + U_{11} \cdot \alpha_{1\ldots}) \end{bmatrix},$$

[1] Technically, the matrix first needs to be extended to the full system size (by forming appropriate tensor products with identity matrices) for the multiplication to make sense.

with $U_{ij} \in \mathbb{C}^{2^{n-1} \times 2^{n-1}}$ and $\alpha_{i\dots} \in \mathbb{C}^{2^{n-1}}$ for $i, j \in \{0, 1\}$. Since the U_{ij} and $\alpha_{i\dots}$ directly correspond to the successors in the respective decision diagrams, matrix-vector (as well as matrix-matrix) multiplication is a native operation on decision diagrams and its complexity scales with the product of the number of nodes of both decision diagrams. Thus, whenever the decision diagrams remain compact throughout the computation, the simulation of quantum circuits can be efficiently conducted using decision diagrams [17]–[22]. While many practical examples lead to compact decision diagram representations [22], their worst case complexity remains exponential.

III. Considered Problem and Related Work

In this section, we describe the considered problem and discuss correspondingly related work including how other types of quantum circuit simulators address this problem.

A. Considered Problem

As reviewed above, the simulation of a quantum circuit $G = g_1 \dots g_{|G|}$ given an initial state $|\varphi\rangle$ entails the sequence of computations

$$|\varphi\rangle\, G = |\varphi\rangle\, g_1 \dots g_{|G|} \equiv U_{|G|} * \cdots * U_1 * |\varphi\rangle\,.$$

Since matrix-matrix and matrix-vector multiplication is associative, the order in which the individual multiplications are conducted can, in principle, be chosen arbitrarily. We refer to such an order of computations as a *simulation path*. Since matrix-vector multiplication is, in general, far less complex than matrix-matrix multiplication, the most natural simulation path is to *sequentially* compute the matrix-vector product of the individual (and compact) gate matrices with the current state vector. However, for a circuit G with $|G|$ gates, there are $|G| * (|G| - 1) * \cdots * 1 = |G|!$, i.e., exponentially many, unique simulation paths—raising the question whether the most natural path indeed is always the best path.

B. Related Work

The connection between decision diagrams and tensor networks has already been pointed out in Section I. Both techniques efficiently represent the initial state as well as all the individual operations in the form of a dedicated data structure—decision diagrams on the one hand, and tensors on the other. Then, they choose a certain path to combine these individual description in order to eventually form a representation of the final quantum state—either by multiplying decision diagrams or by contracting tensors. Hence, the problem of determining an optimal simulation path for a decision diagram poses a similar challenge as determining an optimal contraction order for a tensor network, which has been proven to be NP-hard [14]. A plethora of heuristic methods trying to efficiently solve this challenging task for tensor networks has been developed in the past (see, e.g., [10]–[13]).

However, in case of decision diagrams, this question is hardly studied and almost no heuristics exist for determining an efficient simulation path. Initial works related to the problem considered in this work have been conducted in [21]. But there, only a very small subset of the immense space of possibilities has been explored, namely the possibility of performing a couple of matrix-matrix multiplication before conducting the next matrix-vector multiplication.

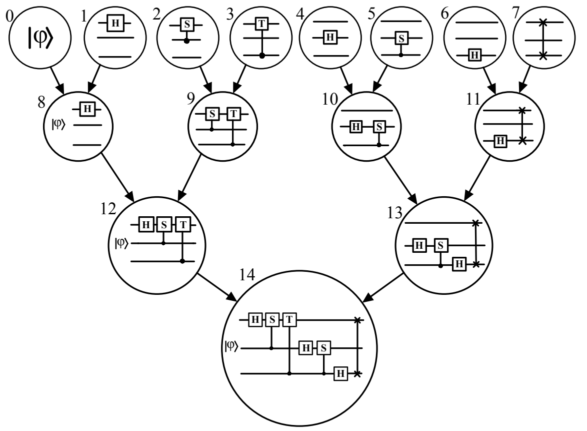


Fig. 3. Task dependency graph for one particular simulation path

IV. Determining Simulation Paths

The simulation of a quantum circuit $G = g_1 \dots g_{|G|}$ with the initial state $|\varphi\rangle$ entails the computation of the expression $U_{|G|} * \cdots * U_1 * |\varphi\rangle$. Initially, this requires the construction of decision diagrams for the initial state and the individual gates. Then, each multiplication in the above expression can be regarded as a *task* that takes two decision diagrams and returns the result of their multiplication. Thus, a path for the simulation of G corresponds to a sequence of (multiplication) tasks that eventually results in the final state vector. It is natural to represent such a sequence as a *task dependency graph*. An example illustrates the idea.

Example 5. *Consider again the* 3*-qubit QFT circuit shown in Fig. 1. Then, the following sequence of tasks describes one particular simulation path of* G*:*

$$[(0, 1), (2, 3), (4, 5), (6, 7), (8, 9), (10, 11), (12, 13)].$$

To this end, index 0 *denotes the initial state, index* 1 *to* $|G|$ *the individual operations, and the result of a task is indexed by the next largest integer not already in use. The corresponding task dependency graph is shown in Fig. 3.*

As a result, arbitrary paths for the simulation of a circuit G using decision diagrams can be specified. The question remains how to determine suitable ones out of the $|G|!$ options. As reviewed in Section III-B, a plethora of methods has been developed for determining efficient contraction paths of tensor networks. In this work, we envision a flow in Python that connects both domains and, as consequence, allows to make use of research conducted towards tensor network contraction.

The flow (as illustrated in Fig. 4) starts off by importing a circuit (given as *OpenQASM* file [23] or Qiskit *QuantumCircuit* object [24]) into the decision diagram framework. Then, a representation of the underlying tensor network is created and fed into the hyper-optimized tensor network contraction tool CoTenGra [10], which is used as a state-of-the-art representative. Afterwards, a task dependency graph is constructed from the obtained contraction plan and used for the decision diagram simulation.

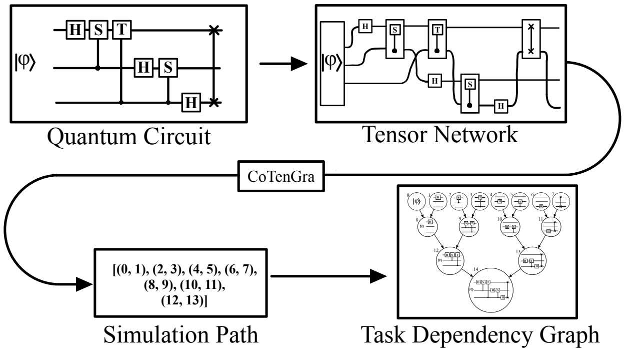


Fig. 4. Automated flow for determining simulation paths

V. Experimental Results

The flow, outlined above, allows to employ strategies developed in the tensor network domain for decision diagram-based simulation. Thus, the potential of different simulation paths can be evaluated. We implemented the proposed flow on top of the publicly available DDSIM tool (github.com/iic-jku/ddsim) and used the resulting implementation for our experimental evaluations[2]. As benchmarks, we considered quantum circuit realizations of a broad range of quantum algorithms, including the Quantum Fourier Transform, Quantum Phase Estimation, GHZ, and Graph State preparation. We used the flow proposed in Section IV to translate the circuit to be simulated to the tensor network domain and employ the contraction path finding tool CoTenGra [10] to determine a corresponding simulation path[3]. The results are summarized in Table I.

As shown by these results, applying methods developed from the domain of tensor networks indeed allows to speed up the simulation of quantum circuits using decision diagrams in many cases. Sometimes even several orders of magnitude compared to the state of the art approach can be observed.

VI. Conclusions

In this work, we studied the importance of the path that is chosen when simulating quantum circuits using decision diagrams connecting it to the domain of tensor networks. We showcased that translating contraction order strategies from the domain of tensor networks does allow for speedups of up to several orders of magnitude compared to the state of the art. This shows the potential impact of choosing a suitable path for the simulation of quantum circuits using decision diagrams, and motivates a thorough investigation of different simulation paths for a broader range of benchmarks.

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 101001318). It has partially been supported by the LIT Secure and Correct Systems Lab funded by the State of Upper Austria as well as by the BMK, BMDW, and the State of Upper Austria in the frame of the COMET program (managed by the FFG).

TABLE I
Experimental Evaluations

Benchmark			Results	
Name	n	$\|G\|$	t_{seq} [s]	t_{cot} [s]
GHZ	64	64	0.00	0.00
GHZ	96	96	0.01	0.01
GHZ	128	128	0.01	0.01
Graph_State	46	115	6.71	5.87
Graph_State	48	120	73.52	13.50
Graph_State	50	125	53.32	5.59
Entangled_QFT	16	160	7.66	0.16
Entangled_QFT	17	178	42.76	0.64
Entangled_QFT	18	198	395.62	0.94
Entangled_QFT	19	218	5 836.17	3.66
QPE	16	158	4.75	0.80
QPE	17	177	27.17	7.05
QPE	18	196	118.11	14.17
QPE	19	217	1 837.00	282.23

n: Number of qubits $|G|$: Gate count of G
t_{seq}: Runtime of sequential simulation path [17]
t_{cot}: Runtime of path determined by CoTenGra [10]

References

[1] G. G. Guerreschi *et al.*, "Intel Quantum Simulator: A cloud-ready high-performance simulator of quantum circuits," *Quantum Sci. Technol.*, 2020.

[2] T. Häner and D. S. Steiger, "0.5 petabyte simulation of a 45-Qubit quantum circuit," in *Int'l Conf. for High Performance Computing, Networking, Storage and Analysis*, 2017.

[3] T. Jones *et al.*, "QuEST and high performance simulation of quantum computers," in *Scientific Reports*, 2018.

[4] R. Jozsa. "On the simulation of quantum circuits." arXiv: quant-ph/0603163. (2006).

[5] I. L. Markov and Y. Shi, "Simulating quantum computation by contracting tensor networks," *SIAM J. Comput.*, vol. 38, no. 3, pp. 963–981, 2008.

[6] J. D. Biamonte and V. Bergholm, "Tensor networks in a nutshell," 2017. arXiv: 1708.00006.

[7] P. Niemann *et al.*, "QMDDs: Efficient quantum function representation and manipulation," *IEEE Trans. on CAD of Integrated Circuits and Systems*, 2016.

[8] L. Chin-Yung, W. Shiou-An, and K. Sy-Yen, "An extended XQDD representation for multiple-valued quantum logic," *IEEE Trans. Comput.*, pp. 1377–1389, 2011.

[9] A. Zulehner, S. Hillmich, and R. Wille, "How to efficiently handle complex values? implementing decision diagrams for quantum computing," in *Int'l Conf. on CAD*, 2019.

[10] J. Gray and S. Kourtis, "Hyper-optimized tensor network contraction," *Quantum*, vol. 5, p. 410, 2021.

[11] C. Huang *et al.* "Classical Simulation of Quantum Supremacy Circuits." arXiv: 2005.06787 [quant-ph]. (2020).

[12] S. Boixo *et al.* "Simulation of low-depth quantum circuits as complex undirected graphical models." arXiv: 1712.05384. (2018).

[13] D. Lykov *et al.* "Tensor Network Quantum Simulator With Step-Dependent Parallelization." arXiv: 2012.02430 [quant-ph]. (2020).

[14] L. Chi-Chung, P. Sadayappan, and R. Wenger, "On optimizing a class of multi-dimensional loops with reduction for parallel execution," *Parallel Process. Lett.*, 1997.

[15] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, 2010.

[16] R. Wille, L. Burgholzer, and M. Artner, "Visualizing decision diagrams for quantum computing," in *Design, Automation and Test in Europe*, 2021.

[17] A. Zulehner and R. Wille, "Advanced simulation of quantum computations," *IEEE Trans. on CAD of Integrated Circuits and Systems*, 2019.

[18] V. Samoladas, "Improved BDD algorithms for the simulation of quantum circuits," in *Algorithms - ESA*, D. Halperin and K. Mehlhorn, Eds., 2008.

[19] G. F. Viamontes, I. L. Markov, and J. P. Hayes, "High-performance QuIDD-Based simulation of quantum circuits," in *Design, Automation and Test in Europe*, 2004.

[20] S. Hillmich *et al.*, "As Accurate as Needed, as Efficient as Possible: Approximations in DD-based Quantum Circuit Simulation," in *Design, Automation and Test in Europe*, 2020.

[21] A. Zulehner and R. Wille, "Matrix-Vector vs. Matrix-Matrix multiplication: Potential in DD-based simulation of quantum computations," in *Design, Automation and Test in Europe*, 2019.

[22] T. Grurl *et al.*, "Arrays vs. Decision Diagrams: A case study on quantum circuit simulators," in *Int'l Symp. on Multi-Valued Logic*, 2020.

[23] A. W. Cross *et al.* "Open Quantum Assembly Language." arXiv: 1707.03429. (2017).

[24] G. Aleksandrowicz *et al.*, "Qiskit: An open-source framework for quantum computing," *Zenodo*, 2019.

[2] All evaluations have been conducted on a machine equipped with an AMD Ryzen 9 3950X CPU and 128 GiB RAM running Ubuntu 20.04.

[3] A timeout of 1 h and a maximum of 256 trials are used to determine a suitable simulation path.