Forward and Backward Constrained Bisimulations for Quantum Circuits Using Decision Diagrams

LUKAS BURGHOLZER, Technical University of Munich, Munchen, Germany
ANTONIO JIMENEZ-PASTOR, Universidad Politécnica de Madrid, Madrid, Spain
KIM GULDSTRAND LARSEN, Aalborg Universitet, Aalborg, Denmark
MIRCO TRIBASTONE, IMT School for Advanced Studies Lucca, Lucca, Italy
MAX TSCHAIKOWSKI, Computer Science, Aalborg Universitet, Aalborg, Denmark
ROBERT WILLE, Technical University of Munich, Munchen, Germany and Software Competence Center Hagenberg GmbH, Hagenberg, Austria

Efficient methods for the simulation of quantum circuits on classical computers are crucial for their analysis due to the exponential growth of the problem size with the number of qubits. Here we study lumping methods based on bisimulation, an established class of techniques that has been proven successful for (classic) stochastic and deterministic systems such as Markov chains and ordinary differential equations. Forward constrained bisimulation yields a lower-dimensional model which exactly preserves quantum measurements projected on a linear subspace of interest. Backward constrained bisimulation gives a reduction that is valid on a subspace containing the circuit input, from which the circuit result can be fully recovered. We provide an algorithm to compute the constraint bisimulations yielding coarsest reductions in both cases, using a duality result relating the two notions. As applications, we provide theoretical bounds on the size of the reduced state space for well-known quantum algorithms for search, optimization, and factorization. Using a prototype implementation, we report significant reductions on a set of benchmarks. In particular, we show that constrained bisimulation can boost decision-diagram-based quantum circuit simulation by several orders of magnitude, allowing thus for substantial synergy effects.

CCS Concepts: • **Software and its engineering → Abstraction, modeling and modularity**; • **Hardware → Quantum computation;**

Additional Key Words and Phrases: Bisimulation, quantum circuits, lumpability, decision diagrams

This work was partially supported by the Poul Due Jensen Foundation Grant 883901, the Villum Investigator Grant S4OS and the project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU. It was also partially funded under the European Union's Horizon 2020 research and innovation programme (DA QC, Grant No. 101001318) and was part of the Munich Quantum Valley, which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus.
Authors' Contact Information: Lukas Burgholzer, Technical University of Munich, Munchen, Germany; e-mail: lukas.burgholzer@tum.de; Antonio Jimenez-Pastor, Universidad Politécnica de Madrid, Madrid, Community of Madrid, Spain; e-mail: antonio.jimenezp@upm.es; Kim Guldstrand Larsen, Aalborg Universitet, Aalborg, Region Nordjylland, Denmark; e-mail: kgl@cs.aau.dk; Mirco Tribastone, IMT School for Advanced Studies Lucca, Lucca, Tuscany, Italy; e-mail: mirco.tribastone@imtlucca.it; Max Tschaikowski, Computer Science, Aalborg Universitet, Aalborg, Denmark; e-mail: tschaikowski@cs.aau.dk; Robert Wille, Technical University of Munich, Munchen, Germany and Software Competence Center Hagenberg GmbH, Hagenberg, Hagenberg, Austria; e-mail: robert.wille@tum.de.

ACM 2643-6817/2025/02-ART13
https://doi.org/10.1145/3712711

ACM Reference Format:
Lukas Burgholzer, Antonio Jimenez-Pastor, Kim Guldstrand Larsen, Mirco Tribastone, Max Tschaikowski, and Robert Wille. 2025. Forward and Backward Constrained Bisimulations for Quantum Circuits Using Decision Diagrams. *ACM Trans. Quantum Comput.* 6, 2, Article 13 (February 2025), 21 pages. https://doi.org/10.1145/3712711

1 Introduction

Quantum computers can solve certain problems more efficiently than classical computers. Earlier instances are Grover's quantum search [30] and Shor's factorization [51]; more recent work addresses the efficient solution of linear equations [32] and the simulation of differential equations [39]. Despite its potential and increasing interest from a commercial point of view, quantum computing is still in its infancy. The number of qubits of current quantum computers is prohibitively small; furthermore, low coherence times and quantum noise lead to high error rates. Further research and improvement of quantum circuits thus hinge on the availability of efficient simulation algorithms on classical computers.

Being described by a unitary complex matrix, any quantum circuit can be simulated by means of array structures and the respective matrix operations [35, 41, 54]. Unfortunately, direct array approaches are subject to the curse of dimensionality [44] because the size of the matrix is exponential in the number of qubits. This motivated the introduction of techniques that try to overcome the exponential growth, resting, for instance, on the stabilizer formalism [1], tensor networks [60, 61], path sum reductions [3], and decision diagrams [31, 44, 63].

Here we study bisimulation relations for quantum circuits. Bisimulation has a long tradition in computer science [50]. For the purpose of this paper, the most relevant branch of research on this topic regards bisimulations for quantitative models such as probabilistic bisimulation [9, 37]. This is closely related to ordinary lumpability for Markov chains [14], also known as *forward* bisimulation [27], which yields an aggregated Markov chain by means of partitioning the original state space, such that the probability of being in each macro-state/block is equal to the sum of the probabilities of each state in that block. Exact lumpability [14], also known as *backward* bisimulation [53], exploits a specific linear invariant induced by a partition of the state space such that states in the same partition block have the same probability at all time points [14]. In an analogous fashion, forward and backward bisimulations have been developed for chemical reaction networks [15, 16, 55], rule-based systems [26, 27], and ordinary differential equations [19, 21].

In all these cases, lumping can be mathematically expressed as a specific linear transformation of the original state space into a reduced space that is induced by a partition. However, in general, lumping allows for arbitrary linear transformations [13, 53]. Since this may introduce loss of information, *constrained lumping* allows one to specify a subspace of interest that should be preserved in the reduction [13, 33, 45, 57]. In partition-based bisimulations, constraints can be specified as suitable user-defined initial partitions of states for which lumping is computed as their (coarsest) refinement [17, 22]. Bisimulation relations for dynamical systems [12, 46] and the notions of constrained linear lumping in [33, 45, 57], instead, can be understood as linear projections (also known as "lumping schemes") into a lower-dimensional system that preserves an arbitrary linear constraint subspace.

The aim of this paper is to boost the simulation of quantum circuits via forward- and backward-type bisimulations that can be constrained to subspaces.[1]

[1]In the following, the simulation of quantum circuits refers to their execution on a classical computer and not to the notion of one-sided bisimulation.

Analogously to the cited literature, with *forward constrained bisimulation (FCB)* the aim is to obtain a lower-dimensional circuit which (exactly) preserves the behavior of the original circuit on the subset of interest. In *backward constrained bisimulation (BCB)*, the reduction is valid on the constraint subspace; in this manner, the original quantum state can be fully recovered from the reduced circuit. Overall, this setting has complementary interpretation with respect to the analysis of a quantum circuit. It is known that a quantum state can only be accessed by means of a quantum measurement, mathematically expressed as a projection onto a given subspace. FCB, in general, preserves any projection onto the constraint subspace. That is, if the constraint subspace contains the measurement subspace, FCB will exactly preserve the quantum measurement, but the full quantum state cannot be recovered in general. Instead, constraining the invariant set of BCB to contain the input of the circuit ensures that the full circuit result can be recovered from the reduced circuit. We show that FCB and BCB are related by a duality property stating that a lumping scheme is an FCB if and only if its complex conjugate transpose is a BCB. Interestingly, this is analogous to the duality established between ordinary (forward) and exact (backward) lumpability for Markov chains [18, 22], although it does not carry over to other models in general [8, 56]. A relevant implication of this result is that one only needs one algorithm to compute both FCB and BCB. As a further contribution of this paper, we present such an algorithm, developed as an adaptation of the CLUE method for the constrained lumping of systems of ordinary differential equations with polynomial right-hand sides [45] of which it inherits the polynomial-time complexity in matrix size.

To show the applicability of our constrained bisimulations, we analyze several case studies for which we report both theoretical and experimental results. Specifically, we first study three classic quantum circuits for search (Grover's algorithm [43, Section 6.2]), optimization [23], and factorization [43, Section 5.3.2], respectively. In Grover's algorithm, the cardinality of the search domain is exponential in the number of qubits; we prove that BCB can always reduce the circuit matrix to a 2×2 matrix while exactly preserving the output of interest. Next, we consider the *quantum approximate optimization (QAOA) algorithm* for solving SAT and MaxCut instances [23]; in this setting, our main theoretical result is an upper bound on the size of the reduced (circuit) matrix by the number of clauses (SAT) or edges (MaxCut). Finally, for quantum factorization we prove that the size of the reduced matrix gives the multiplicative order, that is, it solves the order finding problem to which the factorization problem can be reduced [43].

From an experimental viewpoint, using a prototype based on a publicly available implementation of CLUE, we compare the aforementioned theoretical bounds against the actual reductions on a set of randomly generated instances. Moreover, we conducted a large-scale evaluation on common quantum algorithms collected from the MQT Bench repository [47], showing considerable reductions in all cases. Finally, we demonstrate that constrained bisimulation complements state-of-the-art methods for quantum circuit simulation based on decision diagrams [44], as implemented in the MQT DDSIM tool [63].

Relation to conference version. The present work extends the conference version [34] by computing quantum bisimulations using decision diagrams. Specifically, the exponential time and space complexity of Theorem 3.4 is improved in the new Theorem 3.11. The latter states that the worst-case complexity for computing the quantum bisimulation, the computational bottleneck of [34], is at most polynomial in the size of the largest decision diagram appearing during the computation of the quantum bisimulation. In a similar vein, the Python implementation from [34] has been re-implemented in C++ and uses decision diagrams instead of vectors. Using the improved framework, the quantum benchmarks from [34] have been revisited. Apart from reducing the simulation times from [34] by several orders of magnitude, it is demonstrated that the combination of quantum bisimulation and decision diagrams outperforms each approach in isolation. This speed-up is

attributable to both, the replacement of Python with C++ and the use of decision diagrams instead of vectors.

Further related work. Probabilistic bisimulations [6, 7, 9] have been considered for quantum extensions of process calculi, see [25, 28] and references therein. Similarly to their classical counterparts, these seek to identify concurrent (quantum) processes with similar behavior. The current work, instead, is about boosting the simulation of quantum circuits and is in line with [5, 18, 20, 58, 59]. Specifically, it operates directly over quantum circuits rather than processes and exploits general linear invariants in the (complex) state space. In engineering, invariant-based reductions of linear systems are known under the names of proper orthogonal decomposition [4, 42], Krylov methods [4], and dynamic mode decomposition [29, 36, 48]. Linear invariants also describe safety properties [10] in quantum model checking [64, 65], without being used for reduction though. $\mathcal{L}$-bisimulation [13] and [29, 36] yield the same reductions, with the difference being that the former obtains the smallest reduction up to a given initial constraint, while the latter computes the smallest reduction up to an initial condition. While similarly to us relying on reduction techniques, [29, 36] focus on the reduction of quantum Hamiltonian dynamics, with applications mostly in quantum physics and chemistry. Instead, we study the reduction of quantum circuits which are the prime components of quantum computing. Moreover, we provide a public prototype implementation in C++ using decision diagrams and use it to conduct a large-scale evaluation on quantum circuit benchmarks.

Paper outline. The paper is structured as follows. After a review of core concepts, Section 3 introduces forward and backward constrained bisimulation of (quantum) circuits. There, we also provide an algorithm for the computation of constrained bisimulations by extending [38, 45] to circuits. Section 4 then derives bounds on the reduction sizes of quantum search [30], quantum optimization [23], and quantum order finding [43]. Section 5 conducts a large-scale evaluation on published quantum benchmarks [47] and compares constrained bisimulations with MQT DDSIM with respect to the possibility of speeding up circuit simulations. The paper concludes in Section 6.

2 Preliminaries

Notation. We shall denote by n the number of qubits and set $N = 2^n$ for convenience. Column vectors are denoted by the *ket* notation $|z\rangle$, while the complex conjugate transpose of $|z\rangle$ is denoted by $|z\rangle^\dagger = \langle z|$, i.e., $\langle z| = |\bar{z}\rangle^T$ with $\bar{\cdot}$ and $\cdot^T$ denoting complex conjugation and transpose, respectively. In a similar vein, $\langle z| \, |z\rangle = \langle z|z\rangle$, where $\langle \cdot \mid \cdot \rangle$ is the standard scalar product over $\mathbb{C}^N$. Following standard notation, the canonical basis vectors of $\mathbb{C}^N$ are expressed using tensor products and bit strings $x \in \{0,1\}^n$; specifically, writing $\otimes$ for the Kronecker product, we have $|x_n\rangle \otimes |x_{n-1}\rangle \otimes \cdots \otimes |x_1\rangle = |x_n\rangle\,|x_{n-1}\rangle \ldots |x_1\rangle = |x_n x_{n-1} \ldots x_1\rangle = |d\rangle$, where $0 \leq d \leq 2^n - 1$ is a decimal representation of x, see [43] for details. We usually denote by $|x\rangle$ the canonical basis vector with $x \in \{0,1\}^n$, whereas $|u\rangle, |v\rangle, |w\rangle, |z\rangle \in \mathbb{C}^N$ refer to linear combinations in the form $|z\rangle = \sum_{x\in\{0,1\}^n} c_x \, |x\rangle$ with $c_x \in \mathbb{C}$. For any canonical basis vector, we have $|x\rangle = |\bar{x}\rangle$. To avoid confusion, forward constrained bisimulations (FCB) are denoted by row matrices $L \in \mathbb{C}^{d\times N}$ with $d \leq N$, while backward constrained bisimulations (BCB) are denoted by column matrices $L^\dagger \in \mathbb{C}^{N\times d}$.

2.1 Linear Algebra and Dynamical Systems

We begin by introducing core concepts from linear algebra and quantum computing [40, 43].

Definition 2.1 (Core Concepts).
- The column space of a matrix M are all linear combinations of its columns and is denoted by $\langle M\rangle_c$. One says, the columns of M span $\langle M\rangle_c$.
- The row space of a matrix is the set of all linear combinations of its rows and is denoted by $\langle M\rangle_r$. One says, the rows of M span $\langle M\rangle_r$.

— A (quantum) circuit over n qubits is described by a unitary map $U \in \mathbb{C}^{N\times N}$, that is, $U^{-1} = U^\dagger$.
— A (quantum) state $|z\rangle \in \mathbb{C}^N$ is a vector with (Euclidian) norm one.
— A matrix $P \in \mathbb{C}^{N\times N}$ is an orthogonal projection if $P \circ P = P = P^\dagger$.
— Any vector $|z\rangle \in \mathbb{C}^N$ generates the linear subspace $S_{|z\rangle} = \langle |z\rangle \rangle_c$.

Throughout the paper, we do not work at the higher level where quantum circuits are defined by means of quantum gate compositions [43]. Instead, we work directly at the level of the unitary maps that are induced by such compositions. With this in mind, we use the terms "unitary map" and "quantum circuit" interchangeably.

We distinguish between one- and multi-step applications of a quantum circuit [43]. For an input state $|w_0\rangle \in \mathbb{C}^N$, the full quantum state after *one-step* application is $U|w_0\rangle$. Instead, the full quantum state after a *multi-step* application is given by $U^k|w_0\rangle$, where $k > 1$ is the number of steps. These definitions justify interpreting a quantum circuit as a discrete-time dynamical system as follows.

Definition 2.2 (Dynamical System). A circuit $U \in \mathbb{C}^{N\times N}$ with input state $|w_0\rangle$ induces the discrete time *dynamical system (DS)* $|w_{k+1}\rangle = U|w_k\rangle$, with $k \geq 0$. We call $|w_k\rangle$ the *full quantum state* at step k.

Example 2.3. The one-qubit circuit $U = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$ is known as the Pauli X-gate [43]. In the case of $k \geq 1$ steps and input $|w_0\rangle = |\phi\rangle$, where $|\phi\rangle^\dagger = (1,-1)/\sqrt{2}$, the induced DS can be shown to be $|w_k\rangle = (-1)^k|\phi\rangle$.

The result of a quantum computation is not directly accessible and is usually queried using quantum measurements [43]. These can be described by projective measurements [43], formally given by a family of orthogonal projections $\{P_1, \ldots, P_m\}$ that satisfy $P_1 + \cdots + P_m = I$. When a quantum state $|z\rangle \in \mathbb{C}^N$ is measured, the probability of outcome $1 \leq i \leq m$ is $\pi_i = \langle z|P_i|z\rangle$. For outcome i, the quantum state after measurement is $P_i|z\rangle/\sqrt{\pi_i}$. We will be primarily concerned with the case $\{P, I-P\}$ for a given orthogonal projection P.

Often, one is interested in querying states from a specific subspace S. For example, the result of the HHL algorithm [32], considered in Section 5, is stored in a subset of all qubits, i.e., in a subspace. To this end, it suffices to use a projective measurement that identifies S.

Definition 2.4. Given an orthogonal projection P, we call $P|z\rangle$ the P-measurement of $|z\rangle$. A subspace $S \subseteq \mathbb{C}^N$ is identifiable by P if $P|z\rangle = |z\rangle$ for all $|z\rangle \in S$.

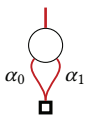
A particularly simple but useful class of projective measurements is the one that measures a single state $|w\rangle$, i.e., that identifies the space $S_{|w\rangle}$ spanned by $|w\rangle$. This is given by the orthogonal projection $P_{|w\rangle} := |w\rangle\langle w|$.

Example 2.5. Assume that we are interested in measuring the result of Example 2.3 using measurement $P_{|\phi\rangle}$ that identifies $S_{|\phi\rangle}$. Then, for $|w_0\rangle = \binom{1}{0}$, it holds that $P_{|\phi\rangle}|w_k\rangle = (-1)^k|\phi\rangle/\sqrt{2}$.

2.2 Linear Algebra with Decision Diagrams

In the following we briefly review decision diagrams and outline how these can be used to conduct linear algebra operations efficiently; see [49] for a detailed discussion. We start by noting that a qubit $|z\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ can be conveniently represented as a decision diagram

$$|z\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle = \begin{pmatrix}\alpha_0 & \alpha_1\end{pmatrix}^\top \equiv$$



.

It consists of a single node with one incoming edge that represents the entry point into the decision diagram as well as two successors that represent the split between the basis states of the single qubit, ending in a terminal node denoted by the black box. The state's amplitudes are annotated to the respective edges. Edges without annotations correspond to an edge weight of 1.

Building on the intuition from a single-qubit state, we can move to larger systems.

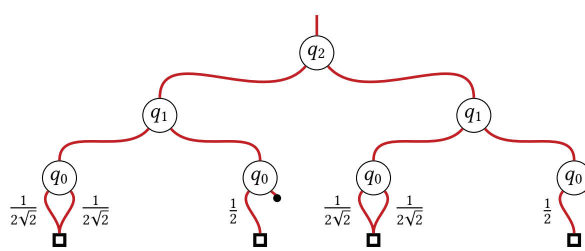
Example 2.6. Consider the following state vector of a three-qubit system:

$$|z\rangle = \begin{pmatrix} \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2} & 0 & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2} & 0 \end{pmatrix}^T \tag{1}$$

Then, $|z\rangle$ can be recursively split into equally-sized parts, essentially making a case distinction over the qubit value $q_i \in \{0, 1\}$, for each $i = 2, 1, 0$:

$$\overbrace{\overbrace{\overbrace{\overset{|000\rangle}{\left(\frac{1}{2\sqrt{2}}\right.} \quad \overset{|001\rangle}{\frac{1}{2\sqrt{2}}}}^{|00q_0\rangle} \quad \overbrace{\overset{|010\rangle}{\frac{1}{2}} \quad \overset{|011\rangle}{0}}^{|01q_0\rangle}}^{|0q_1q_0\rangle} \quad \overbrace{\overbrace{\overset{|100\rangle}{\frac{1}{2\sqrt{2}}} \quad \overset{|101\rangle}{\frac{1}{2\sqrt{2}}}}^{|10q_0\rangle} \quad \overbrace{\overset{|110\rangle}{\frac{1}{2}} \quad \overset{|111\rangle}{\left.0\right)^{\top}}}^{|11q_0\rangle}}^{|1q_1q_0\rangle}}^{|q_2q_1q_0\rangle},$$

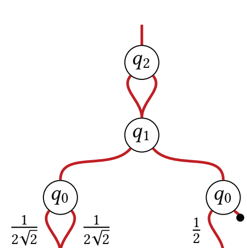
This directly translates to the decision diagram:



Each level of the decision diagram consists of decision nodes with corresponding left and right successor edges. These successors represent the path that leads to an amplitude where the local quantum system (corresponding to the *level* of the node, annotated here with the labels) is in the $|0\rangle$ (left successor) or the $|1\rangle$ state (right successor).

At this point, this has been just a one-to-one translation between the state vector and a graphical representation. The key feature of decision diagrams is that their graph structure allows redundant parts to be merged in the representation instead of being repeated.

Example 2.7. Observe how, in the previous example, the left and right successors of the top-level (q_0) node lead to exactly the same structure. As a result, the whole sub-diagram does not need to be represented twice, i.e.,



From a memory perspective, this reduction compressed the memory required to represent the state by around 43%.

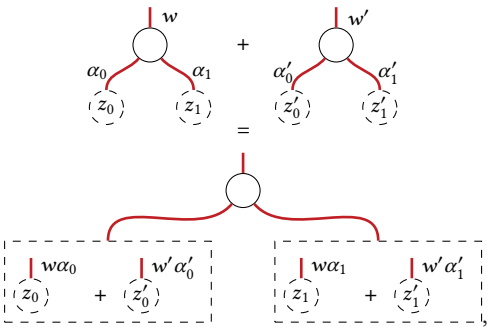
Intuitively, the idea is to represent state vectors compactly as decision diagrams by halving the vector in a recursive fashion until all state vector redundancies have been exploited. Although substantial compression is often possible, we mention that there exist state vectors yielding decision diagrams of exponential size in the number of qubits [66].

Merely defining means for compactly representing states is not sufficient, and it is crucial also to define efficient means to work with or manipulate the resulting representations. In [49] it has been outlined how decision diagrams can be used to perform linear algebra operations efficiently. There, it has been argued that decision diagrams support addition, scalar multiplication, and the scalar product. The main idea behind the efficient implementation is to recursively break the respective operations down into sub-computations.

We shall sketch the addition of decision diagrams next. The idea is to recursively rewrite it by noting that

$$|z\rangle + |z'\rangle = \begin{pmatrix} z_0 \\ z_1 \end{pmatrix} + \begin{pmatrix} z'_0 \\ z'_1 \end{pmatrix} = w \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} + w' \begin{pmatrix} \alpha'_0 \\ \alpha'_1 \end{pmatrix} = \begin{pmatrix} w\alpha_0 + w'\alpha'_0 \\ w\alpha_1 + w'\alpha'_1 \end{pmatrix},$$

where w and w' are common factors of the terms in $|z\rangle$ and $|z'\rangle$, respectively. In the decision-diagram formalism, this corresponds to a simultaneous traversal of both decision diagrams from their roots to the terminal (multiplying edge weights along the way until the individual amplitudes are reached) and back again (accumulating the results of the recursive computations). More precisely,



(2)

where the dashed nodes represent the respective successor decision diagrams. Overall, this results in a complexity that is linear in the size of the larger decision diagram. Scalar multiplication and scalar product of decision diagrams can be treated in a similar fashion and yield the same complexity [49].

3 Constrained Bisimulations for Quantum Circuits

3.1 Forward Constrained Bisimulation

We next introduce FCB.

Definition 3.1 (Forward Constrained Bisimulation, FCB). Fix a circuit defined by $U \in \mathbb{C}^{N\times N}$ with initial state $|w_0\rangle$ and a matrix $L \in \mathbb{C}^{d\times N}$ with orthonormal rows.

(a) The DS reduced by L is given by $|\hat{w}_{k+1}\rangle = \hat{U}\,|\hat{w}_k\rangle$, where $\hat{U} = LUL^\dagger$, and initial state $|\hat{w}_0\rangle = L\,|w_0\rangle$.

(b) L is called forward constrained bisimulation of DS $|w_{k+1}\rangle = U\,|w_k\rangle$ w.r.t. constraint subspace $S \subseteq \mathbb{C}^N$ when $S \subseteq \langle L^\dagger\rangle_c$ and $L\,|w_k\rangle = |\hat{w}_k\rangle$ for all $k \geq 1$.

ALGORITHM 1: Computation of an FCB L w.r.t. subspace S

Require: Unitary map $U \in \mathbb{C}^{N\times N}$ and subspace $S \subseteq \mathbb{C}^N$.
1: **compute** orthonormal basis of S, store it in column matrix $L^\dagger \in \mathbb{C}^{N\times d_0}$
2: **repeat**
3: **for all** columns $|z\rangle$ of $L^\dagger$ **do**
4: **compute** $|\pi\rangle = P_L U |z\rangle$
5: **if** $|\pi\rangle \neq U|z\rangle$ **then**
6: $|w\rangle = U|z\rangle - |\pi\rangle$
7: **append** column $|w\rangle / \langle w||w\rangle$ to $L^\dagger$
8: **end if**
9: **end for**
10: **until** no columns have been appended to $L^\dagger$
11: **return** matrix $L^{\dagger\dagger}$.

Before commenting on the definition, we establish the following.

Lemma 3.2. *The reduced map $\hat{U}$ in Definition 3.1 is unitary.*

Proof. See proof of Theorem 3.9. □

We note that the reduction holds for any choice of initial state $|w_0\rangle$, analogously to the aforementioned forward-type bisimulations [18, 19] for (real-valued) dynamical systems. The assumption of orthonormality of rows of L implies that $d \leq N$, i.e., L is a transformation onto a possibly smaller-dimensional state space. Although it can be dropped without loss of generality [45], it allows a more immediate relation to projective measurements. In fact, matrix L induces the orthogonal projection P_L defined as $P_L = L^\dagger L$. This projective measurement identifies S because $S \subseteq \langle L^\dagger \rangle_c$. Moreover, $P_L|w_k\rangle$ is preserved in the reduced system for any k. To see this, it suffices to multiply $L|w_k\rangle = |\hat{w}_k\rangle$ by $L^\dagger$ from the left and to note that this yields $P_L|w_k\rangle = L^\dagger|\hat{w}_k\rangle$.

Example 3.3. Continuing Example 2.3, it can be shown that the 1×2 matrix $L = |\phi\rangle^\dagger = (1,-1)/\sqrt{2}$ is an FCB w.r.t. $S_{|\phi\rangle}$. In fact, since $\hat{U} = LUL^\dagger = -1$, we obtain $|\hat{w}_{k+1}\rangle = -|\hat{w}_k\rangle$, while a direct calculation confirms that $L|w_0\rangle = |\hat{w}_0\rangle$ implies $L|w_k\rangle = |\hat{w}_k\rangle$ for all $k > 0$. Multiplying both sides by $L^\dagger$ from the left yields $L^\dagger L U^k |w_0\rangle = (-1)^k L^\dagger L|w_0\rangle$. Consequently, the $P_{|\phi\rangle}$-measurement of the original map can be obtained from the $P_{|\phi\rangle}$-measurement of the reduced map.

Algorithm 1 adapts the algorithm for (real-valued) systems of ordinary differential equations with polynomial derivatives developed in [38, 45] to the complex domain and yields the minimal FCB w.r.t. subspace S, i.e., it returns an orthonormal $L \in \mathbb{C}^{d\times N}$ whose dimension d is minimal.

Theorem 3.4 (Minimal FCB). *Algorithm 1 computes a minimal FCB $L \in \mathbb{C}^{d\times N}$ w.r.t. subspace S, that is, the row space of any FCB L' w.r.t. S contains that of L. The complexity of Algorithm 1 is polynomial in N.*

Proof. See proof of Theorem 3.9. □

We briefly comment on Algorithm 1. The idea behind it exploits the fact that L can be shown to be an FCB whenever $L^\dagger$ is an invariant set of the map U, that is, if the column space of $UL^\dagger$ is contained in the column space of $L^\dagger$.

The algorithm begins by initializing $L^\dagger$ with a basis of S in line 1. This ensures that S is contained in the column space of the final result. For every column $|z\rangle$ of $L^\dagger$, the main loop in line 2 checks whether $U|z\rangle$ is in the column space of $L^\dagger$ (line 5) by computing its projection $|\pi\rangle$ onto the column

space of $L^\dagger$ (line 4). If it is not in the column space, the projection will differ from $U|z\rangle$ and the residual $|w\rangle$ must be added to $L^\dagger$. This shows correctness, while the minimality of FCB L follows from the fact that only the necessary residuals are added to $L^\dagger$. The complexity of the algorithm, instead, follows by noting that at most N columns can be added to $L^\dagger$ and that all computations of the main loop require, similarly to the computation in line 1, at most $O(N^3)$ operations.

Remark 3.5. As can be noticed in Algorithm 1, e.g., line 4, the computation of an FCB subsumes the computation of a single step of the circuit. For practical applications to single-step circuits where the modeler is interested in only a single input, FCB may be as expensive as simulating the original circuit directly. Hence, it is obvious that the effectiveness of constrained bisimulations is particularly relevant when simulating the circuit with respect to several inputs, or when considering multi-step applications. Examples of this are provided in Section 4 and a numerical evaluation is carried out in Section 5.

Example 3.6. Consider the FCB $L = |\phi\rangle^\dagger$ w.r.t. subspace $S_{|\phi\rangle}$ from Example 3.3. Then, noting that $(I - P_L)|\phi\rangle = 0$, we infer that Algorithm 1 terminates in line 5. Hence, L is a minimal FCB w.r.t. $S_{|\phi\rangle}$.

3.2 Backward Constrained Bisimulation

BCB yields a reduced system through the identification of an invariant set, i.e., a subspace S such that $U^k|z\rangle \in S$ for any $|z\rangle \in S$ and $k \geq 1$. Whereas in FCB the reduced model can recover projective measurements onto the constraint set *for any* initial set, here one can recover the full quantum state, so long as the initial states belong to the invariant set.

Definition 3.7 (Backward Constrained Bisimulation, BCB). Let U, L and $\hat{U}$ be as in Definition 3.1. Then, $L^\dagger$ is a BCB of the dynamical system $|w_{k+1}\rangle = U|w_k\rangle$ w.r.t. a subspace of inputs $S \subseteq \mathbb{C}^N$ when $S \subseteq \langle L^\dagger\rangle_c$ and whenever $|w_0\rangle = L^\dagger|\hat{w}_0\rangle$ implies $|w_k\rangle = L^\dagger|\hat{w}_k\rangle$ for all $k \geq 1$.

Similarly to FCB, we assume without loss of generality that $L \in \mathbb{C}^{d\times N}$ has orthonormal rows. As anticipated above, FCB and BCB allow for different type of reductions. Indeed, an FCB L does not make any assumptions on the initial condition $|w_0\rangle$, while a BCB $L^\dagger$ does so by requiring $L^\dagger L|w_0\rangle = |w_0\rangle$. Conversely, a BCB $L^\dagger$ allows one to obtain $|w_k\rangle$, while an FCB L allows one to obtain $L|w_k\rangle$ instead of $|w_k\rangle$ itself.

Example 3.8. Fix $|\phi\rangle = (1,-1)^T/\sqrt{2}$ from Example 3.6 and recall that $L = |\phi\rangle^\dagger$, $U|\phi\rangle = -|\phi\rangle$ and $\hat{U} = -1$. Then, $L^\dagger$ is a BCB of U w.r.t. $S_{|\phi\rangle}$. In fact, $L^\dagger L|w_0\rangle = |w_0\rangle$ implies $|w_0\rangle = |\phi\rangle$, while

$$L^\dagger|\hat{w}_k\rangle = (-1)^k L^\dagger|\hat{w}_0\rangle = (-1)^k L^\dagger L|w_0\rangle = (-1)^k|w_0\rangle = U^k|\phi\rangle = |w_k\rangle\,.$$

Example 3.8 anticipates the next result that states FCB and BCB are dual notions. This generalizes the known duality of ordinary and exact lumpability of Markov chains [18, 22].

Theorem 3.9 (Duality). *Fix a unitary map $U \in \mathbb{C}^{N\times N}$ and a subspace $S \subseteq \mathbb{C}^N$. L is an FCB w.r.t. S if and only if $L^\dagger$ is a BCB w.r.t. S.*

Proof. Let $S_0 \subseteq S$ be a basis of some fixed $S \subseteq \mathbb{C}^N$. We first note that the discussion of [33, 38, 45] and [4, 48] can be directly extended to the complex field. With this, we obtain:

(1) $L \in \mathbb{C}^{d\times N}$ is an FCB w.r.t. S if and only if $\langle LU\rangle_r \subseteq \langle L\rangle_r$ with $\langle S_0^\dagger\rangle_r \subseteq \langle L\rangle_r$.
(2) $D \in \mathbb{C}^{N\times d}$ is a BCB w.r.t. S if and only if $\langle UD\rangle_c \subseteq \langle D\rangle_c$ with $\langle S_0\rangle_c \subseteq \langle D\rangle_c$.

Moreover, we observe the following:

$$\begin{aligned}
\langle LU\rangle_r \subseteq \langle L\rangle_r &\Leftrightarrow [U \text{ bijection}]\\
\langle LU\rangle_r = \langle L\rangle_r &\Leftrightarrow [\text{daggering}]\\
\langle U^\dagger L^\dagger\rangle_c = \langle L^\dagger\rangle_c &\Leftrightarrow [U \text{ unitary}]\\
\langle U^{-1} L^\dagger\rangle_c = \langle L^\dagger\rangle_c &\Leftrightarrow [U \text{ bijection}]\\
\langle L^\dagger\rangle_c = \langle UL^\dagger\rangle_c &\Leftrightarrow [U \text{ bijection}]\\
\langle UL^\dagger\rangle_c \subseteq \langle L^\dagger\rangle_c &
\end{aligned}$$

This yields Theorem 3.9, i.e., $L \in \mathbb{C}^{d\times N}$ is an FCB of U w.r.t. constraint S if and only if $L^\dagger \in \mathbb{C}^{N\times d}$ is a BCB of U w.r.t. S (because $S_0^{\dagger\dagger} = S_0$). Moreover, if $L^\dagger$ is computed by Algorithm 1, then $L^\dagger$ is a BCB w.r.t. S, while $L^{\dagger\dagger}$ is an FCB w.r.t. S. This follows by noting that in such a case $L^\dagger \in \mathbb{C}^{N\times d}$ satisfies

$$\begin{aligned}
\langle L^\dagger\rangle_c &= \langle U^k |z\rangle \mid 0 \le k \le N-1, |z\rangle \in S\rangle_c\\
&= \langle U^k |z\rangle \mid 0 \le k \le N-1, |z\rangle \in S_0\rangle_c
\end{aligned}$$

The complexity follows from the discussion after Theorem 3.4. A detailed complexity discussion can be obtained in [45]. Exploiting that an FCB L satisfies $LUL^\dagger L = LU$ by [57], we obtain

$$(LUL^\dagger)^\dagger(LUL^\dagger) = (LU^\dagger L^\dagger)(LUL^\dagger) = LU^\dagger UL^\dagger = LL^\dagger = I_{d\times d},$$

where the first identity is due to daggering, while the second identity follows thanks to $LUL^\dagger L = LU$. Comparing the left-hand side with the right-hand side shows that $\hat{U}$ is unitary. □

In light of the above result, we often speak of a (constrained bisimulation) reduction. Moreover, we note that Theorem 3.9 ensures that a BCB reduction up to input yields an FCB reduction up to output. Here, reduction up to input (output) $|z\rangle$ refers to the fact that BCB (FCB) allows one to recover all quantum states $U^k |z\rangle$ (output measurements $P_{|z\rangle} U |w\rangle$ for all $|w\rangle$).

Remark 3.10. Let $L^\dagger$ be the BCB of U w.r.t. $S_{|w_0\rangle}$, where $|w_0\rangle$ is the input. Then, $L = L^{\dagger\dagger}$ is an FCB w.r.t. $S_{|w_0\rangle}$, implying that $P_L = L^\dagger L$ identifies the column space of $L^\dagger$, see discussion after Definition 3.1. At the same time, the result $U^k |w_0\rangle$ is in the column span of $L^\dagger$ because $U^k |w_0\rangle = |w_k\rangle = L^\dagger |\hat{w}_k\rangle = L^\dagger \hat{U} L |w_0\rangle$.

We end the section by pointing out that, thanks to Theorem 3.9, Algorithm 1 can be used to compute a minimal BCB $L^\dagger$ w.r.t. subspace S. In fact, the only difference is that one should return $L^\dagger$ instead of $L^{\dagger\dagger}$ in the last line of the algorithm. With this change, we notice that Algorithm 1 coincides, in the case of a one-dimensional subspace $S \subseteq \mathbb{C}^N$, with the Krylov subspace [4] that can be obtained by the Arnoldi iteration [48].

3.3 Lumpings with Decision Diagrams

The next result ensures that Algorithm 1 for the computation of the minimal quantum circuit can be implemented by representing vectors as decision diagrams. This builds on the fact that linear algebra operations can be realized using decision diagrams instead of vectors, see Section 3. In particular, the next result states that the minimal quantum circuit can be computed in a number of steps that is polynomial in the size of the largest decision diagram representation appearing during the execution of Algorithm 1.

Theorem 3.11 (Lumpings with Decision Diagrams). *Fix a quantum circuit C that induces the unitary map $U \in \mathbb{C}^{N\times N}$ and let $S_{|z\rangle} \subseteq \mathbb{C}^N$ be the subspace spanned by $|z\rangle \in \mathbb{C}^N$. Assume that the quantum lumping L of U w.r.t. $S_{|z\rangle}$ has dimension d and that $U^1|z\rangle, \ldots, U^d|z\rangle$ can be computed and stored as decision diagrams in time and space $O(s)$ using circuit C.*

(1) Without assuming knowledge of d, the quantum lumping L and the reduced unitary map $\hat{U}$ can be computed in time $O(sd^2)$ and space $O(sd + d^2)$.
(2) Provided that a vector $|v\rangle \in \mathbb{C}^N$ is given in terms of a decision diagram of size $O(s)$, vector $L^\dagger \hat{U}^k L|v\rangle$ can be computed, for any $k \geq 1$, in $O(sd + kd^2)$ steps and in $O(sd)$ space.

Proof. Assuming that two vectors of $\mathbb{C}^N$ are represented by decision diagrams of size $O(s)$, it can be shown [49] that their sum, scalar product and scalar multiplication can be computed in time $O(s)$ and stored as decision diagrams of size $O(s)$. Next, we prove the special case $\kappa = 1$.

To show 1., we set $u_i := U^i|z\rangle$ for $i \geq 0$ and note that, by assumption, u_d is a linear combination of $u_0, \ldots, u_{d-1}$. Hence, if applied to $u_0, \ldots, u_d$, the (numerically stable) *modified Gram-Schmidt (MGS)* method computes a sequence $v_0, \ldots, v_{d-1}, v_d \in \mathbb{C}^N$ with $v_d = 0$ and $v_i \neq 0$ for $0 \leq i < d$, allowing one thus to determine d during execution. Moreover, we can set $L = (v_0, \ldots, v_{d-1})^\dagger$ because it has orthonormal rows and the same row space as $(u_0, \ldots, u_{d-1})^\dagger$. With this, we next assume that $u_0, \ldots, u_{d-1}$ are available as decision diagrams of size $O(s)$ and argue that MGS can be implemented over decision diagrams such that it runs in time $O(sd^2)$ and stores $v_0, \ldots, v_{d-1}$ as decision diagrams of size $O(s)$. To see this, recall that v_k arises from $v_0, \ldots, v_{k-1}$ via $v_k^{(k-1)}/\langle v_k^{(k-1)}, v_k^{(k-1)}\rangle$, where

$$\begin{aligned} v_k^{(1)} &= u_k - \langle u_k, v_1\rangle v_1 \\ v_k^{(2)} &= v_k^{(1)} - \langle v_k^{(1)}, v_2\rangle v_2 \\ &\vdots \\ v_k^{(k-1)} &= v_k^{(k-2)} - \langle v_k^{(k-2)}, v_{k-1}\rangle v_{k-1} \end{aligned}$$

Hence, assuming (as inductive hypothesis) that $v_0, \ldots, v_{k-1}$ can be stored as decision diagrams of size $O(s)$, it follows that $v_k^{(1)}, v_k^{(2)}, \ldots, v_k^{(k-1)}$ can be also stored as decision diagrams of size $O(s)$. This, in turn, implies that v_k can be computed in $O(sd)$ time and space. Since this has to be done for all $k = 0, \ldots, d-1$, we obtain a time complexity of $O(sd^2)$ and a space complexity of $O(sd + d^2)$ because we need to store $\hat{U}$ and the computation of each v_k reuses the previously computed $v_0, \ldots, v_{k-1}$. Once $v_0, \ldots, v_{d-1}$ have been obtained, the reduced matrix $\hat{U}$ can be computed in time $O(sd^2)$ and space $O(sd + d^2)$ because $\hat{U}_{i,j} = \langle u_j, v_i\rangle$.

We next turn to claim 2. To this end, we first note that the vector $L|v\rangle \in \mathbb{C}^d$ can be obtained by computing d scalar products between decision diagrams of size $O(s)$, thus giving rise to a time and space complexity of $O(sd)$. Once the vector $L|v\rangle$ is known, computing $\hat{U}^k L|z\rangle$ can be done in $O(kd^2)$ time and $O(d^2)$ space using common matrix operations over $\mathbb{C}^d$. Finally, for any vector $|\hat{z}\rangle \in \mathbb{C}^d$, vector $L^\dagger|\hat{z}\rangle$ can be represented as a decision diagram of size $O(s)$ by performing d scalar multiplications and additions over decision diagrams of size $O(s)$, yielding a time and space complexity $O(sd)$. □

Provided that the sequence $U^1|z\rangle, U^2|z\rangle, \ldots$ can be computed using decision diagrams whose size is polynomial in the number of qubits, Theorem 3.11 ensures that the quantum lumping of U w.r.t. $S_{|z\rangle}$ can be computed with time and space requirements that are polynomial in the number of qubits.

At the expense of higher complexity, Theorem 3.11 can be extended to quantum lumpings w.r.t. arbitrary subspaces, as stated next.

Corollary 3.12. *Fix a quantum circuit C that induces the unitary map $U \in \mathbb{C}^{N\times N}$ and let $S \subseteq \mathbb{C}^N$ be a subspace spanned by $|z_1\rangle, \ldots, |z_\kappa\rangle \in \mathbb{C}^N$. Assume that each quantum lumping L_l of U w.r.t. $S_{|z_l\rangle}$ has dimension d_l and assume that all $U^1|z_l\rangle, \ldots, U^{d_l}|z_l\rangle$ can be computed and stored as decision diagrams in time and space $O(s)$ using C. Then, 1. and 2. of Theorem 3.11 carry over for $d = \sum_l d_l$.*

Proof. We begin with statement 1. We apply the procedure from Theorem 3.11 to each $|z_l\rangle$ in isolation, thus computing a quantum lumping L_l of U w.r.t. $S_{|z_l\rangle}$. The overall time and space complexity is $O(\sum_l sd_l^2)$ and $O(\sum_l sd_l)$, respectively. Afterwards, we apply MGS to the union of all columns $L_1^\dagger, \ldots, L_\kappa^\dagger$. This comes with time complexity $O(s(\sum_l d_l)^2)$ and space complexity $O(\sum_l sd_l)$. Statement 2. is shown as in Theorem 3.11. □

4 Applications

In this section, we demonstrate that established quantum algorithms enjoy substantial bisimulation reductions. For each application, we provide a brief description of the quantum algorithm and a theoretical bound on its reduction.

4.1 Quantum Search

Let us assume that we are given a non-zero function $f : \{0,1\}^n \to \{0,1\}$ and that we are asked to find some $x \in \{0,1\}^n$ such that $f(x) = 1$. Grover's seminal algorithm describes how this can be achieved in $O(\sqrt{N})$ steps on a quantum computer [43, Section 6.2], thus yielding a quadratic speed-up over a classic computer. For any $x \in \{0,1\}^n$, the Grover map is given by

$$G|x\rangle = (-1)^{f(x)}(I - 2|\psi\rangle\langle\psi|)|x\rangle, \quad \text{with} \quad |\psi\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle \tag{3}$$

The Grover map yields the following celebrated result.

Theorem 4.1 (Quantum Search [43]). *Map G is unitary. Moreover, if the number of sought solutions $M = |\{x \mid f(x) = 1\}|$ satisfies $M \leq N/2$, then measuring $G^\kappa|\psi\rangle$ for $\kappa = \lceil \frac{\pi}{4}\sqrt{N/M} \rceil$ yields a state $|x\rangle$ satisfying $f(x) = 1$ with probability at least $\frac{1}{2}$.*

The next result allows one to compute the result $G^\kappa|\psi\rangle$ from Theorem 4.1 using a map over a single qubit.

Theorem 4.2 (Reduced Grover). *The BCB $L^\dagger \in \mathbb{C}^{N\times d}$ of G w.r.t. $S_{|\psi\rangle}$ has dimension $d = 2$ and a column space spanned by $|\psi\rangle$ and $G|\psi\rangle$.*

Proof. The claim follows by noting that the column space of an BCB w.r.t. $S_{|\psi\rangle}$ is spanned by $|\psi\rangle, G|\psi\rangle, G^2|\psi\rangle, \ldots, G^{N-1}|\psi\rangle$ and so on. This, in turn, is known to have as basis [43, Section 6.2]

$$|\alpha\rangle = \frac{1}{\sqrt{M}}\sum_{x:f(x)=1}|x\rangle \qquad \text{and} \qquad |\beta\rangle = \frac{1}{\sqrt{N-M}}\sum_{x:f(x)=0}|x\rangle,$$

where M is as above, while $|\alpha\rangle$ is the superposition (i.e., sum) of all solution states and $|\beta\rangle$ is the superposition of all non-solution states. □

Remark 4.3. Although BCB $L^\dagger$ w.r.t. $S_{|\psi\rangle}$ always has dimension 2, its column space depends on the oracle function f. This is because f appears in G, see (3).

4.2 Quantum Optimization

The Quantum Approximation Optimization Algorithm (QAOA) [23] is a computational model that has the same expressive power as the common quantum circuit model [2, 23, 24]. It is described by two matrices. The first is the *problem Hamiltonian* H_P for which we are interested in computing a maximal eigenstate, i.e., an eigenvector for a maximal eigenvalue of H_P. The second is the *begin Hamiltonian* H_B for which a maximal eigenstate $|\psi\rangle$ is known already. With this, a maximal eigenstate of H_P can be obtained by performing the QAOA introduced next.

Definition 4.4 (QAOA [23]). For a problem Hamiltonian H_P and a begin Hamiltonian H_B, fix the unitary matrices

$$U_B(\delta) = \exp(-i\delta H_B) \qquad \text{and} \qquad U_P(\delta) = \exp(-i\delta H_P)$$

where $\delta > 0$ is a sufficiently small time step and $\exp(A)$ is the matrix exponential. For a sequence of natural numbers $(k_i, l_i)_{i=1}^{\kappa}$ of length $\kappa \geq 1$, we define

$$|w_\kappa\rangle = U_B(\delta)^{k_\kappa} U_P(\delta)^{l_\kappa} \cdot \ldots \cdot U_B(\delta)^{k_1} U_P(\delta)^{l_1} |\psi\rangle \tag{4}$$

The QAOA with $\kappa \geq 1$ stages is then given by $\max\{\langle w_\kappa| H_P |w_\kappa\rangle \mid (k_i, l_i)_{i=1}^{\kappa}\}$.

While the problem Hamiltonian H_P depends on the task or problem we are solving, the choice of the begin Hamiltonian H_B is informed by the so-called adiabatic theorem, a result that identifies conditions under which QAOA returns a global optimum. A common heuristic is to pick H_B such that H_B and H_P do not diagonalize over a common basis [23, 24] and to assume without loss of generality that $|\psi\rangle = \sum_x |x\rangle / \sqrt{N}$ is the unique maximal eigenvector of H_B.

We next demonstrate that bisimulation can reduce QAOA when applied to SAT and MaxCut, two NP-complete problems [52]. We start by introducing the problem Hamiltonians H_P for both cases.

Definition 4.5 (SAT and MaxCut Problem Hamiltonians).

— For a Boolean formula $\phi = \bigwedge_{i=1}^{M} C_i$, where C_i is a clause over n Boolean variables, the problem Hamiltonian is given by $H_P = \sum_i H_i$, where

$$H_i |x\rangle = \begin{cases} |x\rangle & , C_i(x) \text{ is true} \\ 0 & , C_i(x) \text{ is false} \end{cases}$$

for any $x \in \{0,1\}^n$ that represents a Boolean assignment.

— For an undirected unweighted graph $G = (V, E)$ with vertices $V = \{1, \ldots, n\}$ and edges $E \subseteq V \times V$, we define the problem Hamiltonian $H_P = \sum_{(i,j)\in E} H_{i,j}$, where

$$H_{i,j} |x\rangle = \begin{cases} |x\rangle & , x_i \neq x_j \\ 0 & , x_i = x_j \end{cases}$$

for any $x \in \{0,1\}^n$ that represents a cut $C \subseteq \{1, \ldots, n\}$ by setting $i \in C$ if and only if $x_{i-1} = 1$.

Following this definition, it can be shown that the QAOA $\langle w_\kappa| H_P |w_\kappa\rangle$ from Definition 4.4 corresponds to a quantum measurement reporting either the expected number of satisfied clauses or the expected size of the cut. It is possible to guarantee that QAOA finds a global optimum for a sufficiently high κ [23, 24].

The next result ensures that H_P has BCB $L^\dagger$ w.r.t. $S_{|\psi\rangle}$ whose reduced map is provably small. Moreover, for any such L, it ensures that there exists a begin Hamiltonian H_B for which $L^\dagger$ is a BCB too, thus ensuring substantial reductions of the entire QAOA calculation (4).

Theorem 4.6 (Reduced QAOA). *Fix H_P as in Definition 4.5, any $\delta > 0$ and let $L^\dagger \in \mathbb{C}^{N\times d}$ be a BCB of $U_P(\delta)$ w.r.t. $S_{|\psi\rangle}$. Then*

(1) The column space of $L^\dagger$ is spanned by

$$\left(|\psi\rangle, U_P(\delta)|\psi\rangle, U_P^2(\delta)|\psi\rangle, \ldots, U_P^{M-1}(\delta)|\psi\rangle \right)^\dagger, \tag{5}$$

where M is the number of clauses (SAT) or edges (MaxCUT). Specifically, the dimension of the BCB d is bounded by M.

(2) Then, for any Hamiltonian $\hat{H}_B \in \mathbb{C}^{d\times d}$ (i.e., Hermitian matrix), there is a Hamiltonian $H_B \in \mathbb{C}^{N\times N}$ such that

- *$L^\dagger$ is a BCB of $U_B(\delta) = \exp(-i\delta H_B)$ w.r.t. $S_{|\psi\rangle}$, while its reduced map is $\hat{U}_B(\delta) = \exp(-i\delta\hat{H}_B)$*
- *The computation (4) satisfies*

$$\begin{aligned} |w_\kappa\rangle &= U_B(\delta)^{k_\kappa} U_P(\delta)^{l_\kappa} \cdot \ldots \cdot U_B(\delta)^{k_1} U_P(\delta)^{l_1} |\psi\rangle \\ &= L^\dagger \hat{U}_B(\delta)^{k_\kappa} \hat{U}_P(\delta)^{l_\kappa} \cdot \ldots \cdot \hat{U}_B(\delta)^{k_1} \hat{U}_P(\delta)^{l_1} L |\psi\rangle \end{aligned} \tag{6}$$

The QAOA in $\mathbb{C}^N$ thus corresponds to a QAOA in the reduced space $\mathbb{C}^d$.

Proof. We begin by proving 1. For SAT, it can be seen that $H_P|x\rangle = \nu|x\rangle$, where $0 \le \nu \le M$ is the number of clauses that are satisfied by assignment x. A similar formula holds for MaxCut, the difference being that ν is the size of the cut x. It should be noted that H_P is in diagonal form for both SAT and MaxCut. If m denotes the number of distinct eigenvalues of H_P, then $m \le M$, where M is in the case of SAT or MaxCUT, respectively, the number of clauses or edges. The same can be said regarding its matrix exponential $U_P(\delta)$ which, being unitary, enjoys an eigendecomposition, allowing us to write $|\psi\rangle = \sum_{i=1}^m c_i |z_i\rangle$, where $|z_i\rangle$ is an eigenvector for eigenvalue λ_i of $U_P(\delta)$. This yields

$$U^k|\psi\rangle = \sum_{i=1}^m c_i \lambda_i^k |z_i\rangle$$

for all $k \ge 0$. Without loss of generality, consider $d \le m$ such that $c_k = 0$ for all $k > d$ and $c_k \ne 0$ otherwise. Writing the vectors $\{U^k|\psi\rangle \mid 0 \le k \le m-1\}$ in the basis $|z_1\rangle, \ldots, |z_d\rangle$ gives rise to a regular Vandermonde matrix [48] in $\mathbb{C}^{d\times d}$. This shows that $\{U^k|\psi\rangle \mid d \le k \le M-1\}$ are linear combinations of $\{U^k|\psi\rangle \mid 0 \le k \le d-1\}$, completing the proof of 1. Instead, 2. follows from the definition of BCB and Lemma 4.7 from below. □

The following auxiliary result is needed in the proof of Theorem 4.6.

Lemma 4.7. *Pick any $L \in \mathbb{C}^{d\times N}$ and $Q \in \mathbb{C}^{(N-d)\times N}$ so that the rows of L and Q comprise an orthonormal basis of $\mathbb{C}^N$, and define*

$$U_B = L^\dagger \hat{U}_B L + Q^\dagger \tilde{U}_B Q, \qquad \hat{U}_B = \exp\left(-i\delta\hat{H}_B\right), \qquad \tilde{U}_B = \exp\left(-i\delta\tilde{H}_B\right)$$

for any Hamiltonian $\hat{H}_B \in \mathbb{C}^{d\times d}$ and $\tilde{H}_B \in \mathbb{C}^{(N-d)\times(N-d)}$. Then, U_B is unitary, L is an FCB of it w.r.t. $S_{|\psi\rangle}$, and $\hat{U}_B$ is its reduced map. In addition, there exists a Hamiltonian $H_B \in \mathbb{C}^{N\times N}$ that satisfies $U_B = \exp(-i\delta H_B)$.

Proof. We first show that $LU_B = LU_B L^\dagger L$ as this implies that L is an FCB of U by [57]. To see this, we note that

$$\begin{aligned} LU_B L^\dagger L &= L\left(L^\dagger \hat{U}_B L + Q^\dagger \tilde{U}_B Q\right) L^\dagger L = LL^\dagger \hat{U}_B LL^\dagger L + LQ^\dagger \tilde{U}_B QL^\dagger L = \hat{U}_B L \\ LU_B &= L\left(L^\dagger \hat{U}_B L + Q^\dagger \tilde{U}_B Q\right) = LL^\dagger \hat{U}_B L + LQ^\dagger \tilde{U}_B Q = \hat{U}_B L \end{aligned}$$

where we have used that $LL^\dagger = 0$ and $LQ^\dagger = 0$, which follows from the choice of Q. From the calculation, we can also infer that $\hat{U}_B = LU_BL^\dagger$, i.e., $\hat{U}_B$ is indeed the reduced map. In a similar fashion, one can note that Q is also an FCB of U_B and that $\tilde{U}_B$ is the respective reduced map. Since both $\hat{U}_B$ and $\tilde{U}_B$ are unitary, we infer that also U_B is unitary (alternatively, a direct calculation yields $I = U_BU_B^\dagger$). Since any unitary matrix can be written as a matrix exponential of a Hamiltonian, there exists a Hamiltonian H_B satisfying $U_B = \exp(-i\delta H_B)$. □

4.3 Quantum Factorization and Order Finding

Let us assume that we are given a composite number N which we seek to factorize. As argued in [43, Section 5.3.2], this problem can be solved in randomized polynomial time, provided that the same holds for the order finding problem. Given some randomly chosen $x \in \{2, 3 \ldots, N-1\}$, the latter asks to compute the multiplicative order of x modulo N, i.e., the smallest $r \geq 1$ satisfying $x^r \mod N = 1$. Following [43, Section 5.3.1], we consider the quantum algorithm defined by the unitary map

$$U|y\rangle = \begin{cases} |xy \mod N\rangle & , 0 \leq y < N \\ |y\rangle & , N \leq y < 2^l \end{cases}$$

Here, $l \geq 1$ is the smallest number satisfying $N \leq 2^l$.

The next result allows us to relate the order of x to the dimension of the BCB w.r.t. $S_{|1\rangle}$. This fact is exploited in Shor's factorization algorithm [43].

Theorem 4.8 (Reduced Order Finding). *The dimension of the BCB of U w.r.t. $S_{|1\rangle}$ is equal to the order of x modulo N.*

Proof. If can be shown [43] that the U from above is unitary and that $U|u_s\rangle = e^{2\pi is/r}|u_s\rangle$ for all $0 \leq s \leq r-1$, where

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left[\frac{-2\pi isk}{r}\right] \left|x^k \mod N\right\rangle \qquad \text{and} \qquad \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle .$$

With this, $U^k|1\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} (e^{2\pi is/r})^k u_s$ for any $p \geq 0$. Hence, the minimal BCB with respect to $S_{|1\rangle}$ is contained in the span of $u_0, \ldots, u_{r-1}$. To see that the dimension is exactly r, we note that the vectors $\{U^k|\psi\rangle \mid 0 \leq k \leq r-1\}$, written in the basis $u_0, \ldots, u_{r-1}$, constitute a regular Vandermonde matrix [48] in $\mathbb{C}^{r\times r}$. □

5 Numerical Experiments

We evaluated our approach on the applications from Section 4 and the MQT Bench quantum circuit benchmark suite [47]. To this end, we extended the publicly available implementation of the CLUE algorithm from [38, 45] which used Python, Gaussian elimination and sparse vectors and matrices. The new version was re-implemented in C++ and uses the state-of-the-art decision diagram package for quantum computing provided as part of the Munich Quantum Toolkit [62] in the MQT Core library (available at https://github.com/cda-tum/mqt-core). A version that uses sparse vectors and matrices for representing state vectors and quantum circuits, respectively, is offered for completeness. The respective prototype is publicly accessible at the GitHub repository `CLUE`, branch `quantum+cpp` (available at https://github.com/Antonio-JP/CLUE/tree/quantum%2Bcpp), all results reported were executed on a machine with an i7-8665U CPU, 32GB RAM and a 1024GB SSD.

As anticipated, the evaluation demonstrates that a combination of quantum bisimulation with decision diagrams outperforms each individual approach.

Table 1. Simulation Times of Quantum Applications from Section 4 in Case of Three Different Regimes

	Grover			SAT			MAXCUT		
qubits	CLUE	DDSIM+CLUE	DDSIM	CLUE	DDSIM+CLUE	DDSIM	CLUE	DDSIM+CLUE	DDSIM
5	0.0028	0.0020	**0.0019**	**0.0035**	0.0111	0.0134	**0.0022**	0.0038	0.0070
6	0.0112	**0.0023**	0.0025	**0.0080**	0.0160	0.0758	**0.0069**	0.0086	0.0339
7	0.0463	**0.0026**	0.0032	**0.0186**	0.0326	0.4515	0.0225	**0.0179**	0.2372
8	0.1752	**0.0032**	0.0051	**0.0460**	0.0746	1.8876	0.0593	**0.0421**	1.9653
9	0.6328	**0.0037**	0.0072	**0.1150**	0.1493	13.5664	0.1606	**0.1173**	17.1287
10	1.8453	**0.0049**	0.0115	**0.1789**	0.2560	46.2972	0.3117	**0.1675**	52.3542
11	7.1148	**0.0069**	0.0174	0.4323	**0.3356**	208.8560	0.8154	**0.3567**	313.6538
12	29.8714	**0.0114**	0.0274	1.0144	**0.6830**	271.8651	2.4235	**0.7538**	418.2120
13	128.0926	**0.0196**	0.0431	2.6211	**1.3712**	>500	7.3718	**2.3766**	>500
14	>500	**0.0349**	0.0707	5.6021	**2.4126**	>500	17.3324	**4.2761**	>500
15	>500	**0.0670**	0.1077	13.2476	**4.9066**	>500	43.4601	**9.9412**	>500
16	>500	**0.1231**	0.1801	30.1186	**10.0833**	>500	111.2968	**48.0762**	>500
17	>500	**0.2470**	0.2976	64.4899	**26.5241**	>500	279.0988	**135.6231**	>500
18	>500	**0.4723**	0.4993	157.1778	**89.1793**	>500	>500	**334.3610**	>500
19	>500	0.9447	**0.9167**	298.1822	**113.3666**	>500	>500	>500	>500
20	>500	1.8267	**1.7992**	>500	**125.5865**	>500	>500	>500	>500
21	>500	3.5860	**3.4293**	>500	>500	>500	>500	>500	>500
22	>500	**7.2422**	7.7943	>500	>500	>500	>500	>500	>500
23	>500	**14.4505**	17.1884	>500	>500	>500	>500	>500	>500
24	>500	**28.4858**	45.6023	>500	>500	>500	>500	>500	>500
25	>500	**58.6463**	142.5514	>500	>500	>500	>500	>500	>500
26	>500	**112.1946**	331.1369	>500	>500	>500	>500	>500	>500
27	>500	**225.3220**	>500	>500	>500	>500	>500	>500	>500
28	>500	**455.1170**	>500	>500	>500	>500	>500	>500	>500

CLUE: the circuit is reduced using vectors and simulated; DDSIM+CLUE: the circuit is reduced using decision diagrams and then simulated; DDSIM: the original circuit is simulated using decision diagrams without any reduction.

5.1 Applications from Section 4

Here we report the results of numerical experiments on the applications discussed in Section 4. Starting with five qubits, we increased the number until a timeout of 500 seconds has been reached. To allow for a representative evaluation, we averaged runtimes over 50 independent runs for each case study. Specifically, the instances were generated as follows:

- *Grover's algorithm* (Section 4.1): for a fixed numbers n of qubits, we pick randomly an element $m \in \{0, \ldots, 2^n - 1\}$. We create the corresponding circuit for Grover's algorithms with n qubits (one of them is ancillary) that looks for the element m.
- *Quantum Optimization for SAT* (Section 4.2): for each number of qubits n, we generate a random formula with m clauses (m is randomly selected between n and $3n$), where each clause has three variables at most. We guarantee that every formula contains all n variables.
- *Quantum Optimization for MaxCut* (Section 4.2): for each number of qubits n, we generate an Erdős-Rényi [11] graph with n nodes and edge probability $\frac{1}{3}$.

We considered three different regimes:

- CLUE: quantum bisimulation is computed using CLUE that relies on vector matrix representations; the simulation is conducted using the reduced system;
- DDSIM+CLUE: quantum bisimulation is computed using CLUE that relies on decision diagrams; the simulation is conducted using the reduced system, represented as a matrix;
- DDSIM: using decision diagrams, we simulate the full circuit.

The number of steps κ was set to the smallest integer greater than or equal to $\sqrt{N}$. The choice of κ is motivated by Theorem 4.1 and the discussion around the so-called adiabatic theorem in [23, 24].

Overall, Table 1 demonstrates that a combination of CLUE with DDSIM outperforms each individual approach in isolation. More specifically, we explain the flip from DDSIM+CLUE to DDSIM

Table 2. Reduction Ratio (RR) and Time (s) Comparison between CLUE and DDSIM+CLUE for Computing a Lumping w.r.t. $|0\rangle$

model	qubits	CLUE		DDSIM+CLUE	
		RR for $S_{\|0\rangle}$	Time (s)	RR for $S_{\|0\rangle}$	Time (s)
ae	7	100.00%	4.0022	100.00%	**3.4999**
	8	100.00%	**31.7723**	100.00%	32.5920
	9	100.00%	**264.9636**	-	>500
dj	15	0.01%	0.1223	0.01%	**0.0252**
	16	-	>500	<0.01%	**0.0499**
	⋮	⋮	⋮	⋮	⋮
	27	-	>500	<0.01%	**65.7853**
	28	-	>500	<0.01%	**125.6654**
	29	-	>500	<0.01%	**249.1532**
ghz	11	1.56%	2.5651	1.56%	**2.3672**
	12	0.78%	5.0972	0.78%	**3.5408**
	13	0.39%	**10.2004**	0.39%	11.8914
	14	0.20%	**14.9587**	0.20%	40.8600
	15	0.10%	**30.5938**	-	>500
graphstate	8	9.38%	1.6081	9.38%	**0.1426**
	9	3.52%	6.2612	3.52%	**0.2846**
	10	0.98%	13.0928	0.98%	**0.1867**
	11	2.34%	202.4504	2.34%	**2.5678**
	12	0.29%	210.7506	0.29%	**0.9886**
	13	-	>500	0.15%	**0.8902**
	14	-	>500	0.13%	**33.3943**
hhl	8	87.50%	**12.4598**	39.30%	18.8924
portfolioqaoa	6	100.00%	**0.5800**	100.00%	1.1288
	7	100.00%	**4.1406**	100.00%	5.8476
	8	100.00%	**31.2608**	100.00%	54.4699
	9	100.00%	**263.1970**	-	>500
portfoliovqe	7	100.00%	**3.9064**	100.00%	4.0044
	8	100.00%	31.6000	100.00%	**30.6770**
	9	100.00%	**266.4854**	100.00%	490.4663
pricingcall	11	3.17%	8.6745	3.17%	**1.9269**
	13	1.57%	171.3546	1.57%	**16.7369**
	15	-	>500	0.79%	**142.1170**
pricingput	11	3.17%	7.3885	3.17%	**1.4735**
	13	1.57%	129.0688	1.57%	**10.8187**
	15	-	>500	0.79%	**130.6928**
qft	10	0.20%	3.0963	0.20%	**0.0041**
	11	0.10%	11.0858	0.10%	**0.0061**
	12	0.05%	45.4641	0.05%	**0.0086**
	13	-	>500	0.02%	**0.0136**
	⋮	⋮	⋮	⋮	⋮
	25	-	>500	<0.01%	**33.9910**
	26	-	>500	<0.01%	**69.0587**
	27	-	>500	<0.01%	**171.3884**
qnn	6	100.00%	**0.5825**	100.00%	1.6813
	7	100.00%	**4.0485**	100.00%	6.6028
	8	100.00%	**31.4335**	100.00%	52.0715
	9	100.00%	**263.1350**	-	>500
qpeexact	7	100.00%	1.3285	50.00%	**0.3797**
	8	100.00%	10.8579	50.00%	**2.3920**
	9	100.00%	88.0831	50.00%	**28.7468**
	10	-	>500	50.00%	**472.6555**
qpeinexact	7	100.00%	1.7021	50.00%	**0.5057**
	8	100.00%	13.1596	50.00%	**2.9213**
	9	100.00%	107.9556	50.00%	**30.2104**
qwalk	6	50.00%	**0.0466**	50.00%	1.1475
	7	62.50%	**0.3807**	50.00%	4.5801
	8	100.00%	**6.9250**	50.00%	30.7021
	9	100.00%	**56.6530**	-	>500
	10	100.00%	**465.2573**	-	>500
tsp	9	41.41%	74.2882	41.41%	**72.8202**
vqe	6	100.00%	**0.5768**	100.00%	4.5901
	7	100.00%	**3.8829**	100.00%	8.2355
	8	100.00%	**31.1499**	100.00%	457.2644
	9	100.00%	**264.7780**	100.00%	414.3664
wstate	7	100.00%	**2.3740**	100.00%	2.5267
	8	100.00%	**18.2656**	100.00%	23.3446
	9	100.00%	**151.7882**	100.00%	391.4060

for qubits 19, 20, 21 in case of Grover by numerical noise. Indeed, up to qubit number 20, all running times are below one second. Instead, the difference in DDSIM running times for Grover and quantum optimization can be explained with the fact that the computations of the former require smaller decision diagrams than the computations of the latter. It has to be noted that the performance of DDSIM may be improved by using different internal DDSIM settings (e.g., variable order), or by considering a different quantum circuit that realizes quantum optimization. This should not come as surprise because a DDSIM setting which would allow for efficient simulation of all quantum circuits would disprove quantum supremacy.

5.2 Benchmark Circuits

In Table 2, we compare the scalability of vector-matrix operations versus that of decision diagrams, this time on quantum benchmarks from [47], available at https://www.cda.cit.tum.de/mqtbench/. To this end, we computed the constrained bisimulation w.r.t. $|0\rangle$ for all circuits in each benchmark family until a timeout of 500 seconds has been reached. We show the *reduction ratio (RR)*, given by $\frac{d}{N}$, w.r.t. input $|0\rangle$ and the times (in seconds) spent under regimes CLUE and DDSIM+CLUE, averaged over five executions. We note that some circuits were only available for a specific number of qubits (e.g., HHL and price calls). For the benefit of presentation, we omit qubit numbers whose simulation times were below one second for both CLUE and DDSIM+CLUE.

In general, it can be observed that substantial reductions could be obtained for a number of benchmark families. Concerning scalability, we note that DDSIM+CLUE outperforms CLUE in

most cases. An exception to this rule is when a circuit cannot be reduced substantially. In such a case, an extension of CLUE by decision diagrams tends to perform worse. Exceptions to this rule are however possible, e.g., the ghz family. This enjoys a very sparse matrix representation (with only two nonzero entries per row), whereas the respective decision diagrams are bigger.

6 Conclusion

We introduced forward and backward constrained bisimulations for quantum circuits which allow by means of reduction to preserve an invariant subspace of interest. The applicability of the approach was demonstrated by obtaining substantial reductions of common quantum algorithms, including, in particular, quantum search, quantum approximate optimization algorithms for SAT and MaxCut, as well as a number of benchmark circuits. Overall, the results suggest that constrained bisimulation can be used as a tool for speeding up the simulation of quantum circuits on classic computers, especially when the circuit is to be simulated under several initial conditions or for multi-step applications.

In line with the relevant literature on bisimulations for dynamical systems, constrained bisimulations introduce loss of information due to their underlying projection onto a smaller-dimensional state space; the information that is preserved, however, is exact. A relevant issue for future work is to consider approximate variants of bisimulation for quantum circuits, in order to find more aggressive reductions or to capture quantum-specific phenomena such as quantum noise.

References

[1] Scott Aaronson and Daniel Gottesman. 2004. Improved simulation of stabilizer circuits. *Physical Review A* 70, 5 (2004), 052328.

[2] D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev. 2004. Adiabatic quantum computation is equivalent to standard quantum computation. In *45th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, Rome, Italy, 10. https://doi.org/10.1109/focs.2004.8

[3] Matthew Amy. 2018. Towards large-scale functional verification of universal quantum circuits. In *QPL*, Peter Selinger and Giulio Chiribella (Eds.), Vol. 287. 1–21.

[4] A. Antoulas. 2005. *Approximation of Large-Scale Dynamical Systems*. SIAM.

[5] Giorgio Bacci, Giovanni Bacci, Kim Guldstrand Larsen, and Radu Mardare. 2013. The BisimDist library: Efficient computation of bisimilarity distances for Markovian models. In *QEST*, Kaustubh R. Joshi, Markus Siegle, Mariëlle Stoelinga, and Pedro R. D'Argenio (Eds.). 278–281.

[6] Giorgio Bacci, Giovanni Bacci, Kim G. Larsen, and Radu Mardare. 2016. Complete axiomatization for the bisimilarity distance on Markov chains. In *CONCUR (LIPIcs)*, Josée Desharnais and Radha Jagadeesan (Eds.), Vol. 59. 21:1–21:14.

[7] Giorgio Bacci, Giovanni Bacci, Kim G. Larsen, and Radu Mardare. 2018. A complete quantitative deduction system for the bisimilarity distance on Markov chains. *Log. Methods Comput. Sci.* 14, 4 (2018).

[8] Giorgio Bacci, Giovanni Bacci, Kim G. Larsen, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. 2021. Efficient local computation of differential bisimulations via coupling and up-to methods. In *Symposium on Logic in Computer Science, LICS*. 1–14.

[9] Christel Baier and Holger Hermanns. 1997. Weak bisimulation for fully probabilistic processes. In *CAV*. 119–130.

[10] Christel Baier and Joost-Pieter Katoen. 2008. *Principles of Model Checking*. MIT Press.

[11] Béla Bollobás. 2001. *Random Graphs* (2 ed.). Cambridge University Press.

[12] Michele Boreale. 2019. Algebra, coalgebra, and minimization in polynomial differential equations. *Log. Methods Comput. Sci.* 15, 1 (2019).

[13] Michele Boreale. 2020. Complete algorithms for algebraic strongest postconditions and weakest preconditions in polynomial odes. *Sci. Comput. Program.* 193 (2020), 102441.

[14] Peter Buchholz. 1994. Exact and ordinary lumpability in finite Markov chains. *Journal of Applied Probability* 31, 1 (1994), 59–75.

[15] Luca Cardelli, Isabel Cristina Pérez-Verona, Mirco Tribastone, Max Tschaikowski, Andrea Vandin, and Tabea Waizmann. 2021. Exact maximal reduction of stochastic reaction networks by species lumping. *Bioinform.* 37, 15 (2021), 2175–2182.

[16] Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. 2015. Forward and backward bisimulations for chemical reaction networks. In *CONCUR*. 226–239.
[17] Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. 2016. Comparing chemical reaction networks: A categorical and algorithmic perspective. In *Symposium on Logic in Computer Science, LICS*. 485–494.
[18] Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. 2016. Symbolic computation of differential equivalences. In *POPL*. Elsevier BV, 137–150. https://doi.org/10.1016/j.tcs.2019.03.018
[19] L. Cardelli, M. Tribastone, Max Tschaikowski, and A. Vandin. 2017. Maximal aggregation of polynomial dynamical systems. *Proceedings of the National Academy of Sciences* 114, 38 (2017), 10029 – 10034.
[20] Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. 2017. Syntactic Markovian bisimulation for chemical reaction networks. In *Models, Algorithms, Logics and Tools - Essays Dedicated to Kim Guldstrand Larsen on the Occasion of His 60th Birthday*, Luca Aceto, Giorgio Bacci, Giovanni Bacci, Anna Ingólfsdóttir, Axel Legay, and Radu Mardare (Eds.), Vol. 10460. 466–483.
[21] Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. 2018. Guaranteed error bounds on approximate model abstractions through reachability analysis. In *Quantitative Evaluation of Systems*, Annabelle McIver and Andras Horvath (Eds.). Springer International Publishing, Cham, 104–121. https://doi.org/10.1007/978-3-319-99154-2_7
[22] Salem Derisavi, Holger Hermanns, and William H. Sanders. 2003. Optimal state-space lumping in Markov chains. *Inform. Process. Lett.* 87, 6 (2003), 309 – 315.
[23] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A Quantum approximate optimization algorithm. (2014), 16 pages. https://doi.org/10.48550/ARXIV.1411.4028
[24] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. 2000. Quantum computation by adiabatic evolution. (2000), 24 pages. https://doi.org/10.48550/ARXIV.QUANT-PH/0001106
[25] Yuan Feng, Runyao Duan, Zhengfeng Ji, and Mingsheng Ying. 2007. Probabilistic bisimulations for quantum processes. *Information and Computation* 205, 11 (2007), 1608–1639.
[26] Jérôme Feret, Vincent Danos, Jean Krivine, Russ Harmer, and Walter Fontana. 2009. Internal coarse-graining of molecular systems. *Proceedings of the National Academy of Sciences* 106, 16 (2009), 6453–6458.
[27] Jerome Feret, Thomas Henzinger, Heinz Koeppl, and Tatjana Petrov. 2012. Lumpability abstractions of rule-based systems. *Theoretical Computer Science* 431, 0 (2012), 137-164.
[28] Simon J. Gay and Rajagopal Nagarajan. 2005. Communicating quantum processes. In *Proceedings of the 32nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL05)*. ACM, Long Beach, CA, USA, 145–157. https://doi.org/10.1145/1040305.1040318
[29] Andy Goldschmidt, E. Kaiser, J. L. DuBois, S. L. Brunton, and J. N. Kutz. 2021. Bilinear dynamic mode decomposition for quantum control. *New Journal of Physics* 23, 3 (2021), 033035.
[30] Lov K. Grover. 1996. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing - STOC '96 (STOC '96)*. ACM Press, Philadelphia, PA, USA, 212—-219. https://doi.org/10.1145/237814.237866
[31] Thomas Grurl, Jurgen Fus, Stefan Hillmich, Lukas Burgholzer, and Robert Wille. 2020. Arrays vs. decision diagrams: A case study on quantum circuit simulators. In *2020 IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL'20)*. IEEE, Miyazaki, Japan, 6. https://doi.org/10.1109/ismvl49045.2020.000-9
[32] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. 2009. Quantum algorithm for linear systems of equations. *Physical Review Letters* 103, 15 (2009), 150502.
[33] Antonio Jiménez-Pastor, Joshua Paul Jacob, and Gleb Pogudin. 2022. Exact linear reduction for rational dynamical systems. In *Computational Methods in Systems Biology*, Ion Petre and Andrei Păun (Eds.). Springer International Publishing, Cham, 198–216.
[34] Antonio Jiménez-Pastor, Kim G. Larsen, Mirco Tribastone, and Max Tschaikowski. 2024. Forward and backward constrained bisimulations for quantum circuits. In *Tools and Algorithms for the Construction and Analysis of Systems - 30th International Conference, TACAS 2024*, Bernd Finkbeiner and Laura Kovács (Eds.). 343–362.
[35] N. Khammassi, I. Ashraf, X. Fu, C. G. Almudever, and K. Bertels. 2017. QX: A high-performance quantum computer simulation platform. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. IEEE, Lausanne, Switzerland, 464–469. https://doi.org/10.23919/DATE.2017.7927034
[36] Akshat Kumar and Mohan Sarovar. 2014. On model reduction for quantum dynamics: symmetries and invariant subspaces. *Journal of Physics A: Mathematical and Theoretical* 48, 1 (2014), 015301.
[37] Kim G. Larsen and Arne Skou. 1991. Bisimulation through probabilistic testing. *Inf. Comput.* 94, 1 (1991), 1–28.
[38] Alexander Leguizamon-Robayo, Antonio Jiménez-Pastor, Micro Tribastone, Max Tschaikowski, and Andrea Vandin. 2023. Approximate constrained lumping of polynomial differential equations. In *Computational Methods in Systems Biology*, Jun Pang and Joachim Niehren (Eds.). Springer Nature Switzerland, Cham, 106–123.

[39] Jin-Peng Liu, Herman Øie Kolden, Hari K. Krovi, Nuno F. Loureiro, Konstantina Trivisa, and Andrew M. Childs. 2021. Efficient quantum algorithm for dissipative nonlinear differential equations. *Proceedings of the National Academy of Sciences* 118, 35 (Aug. 2021), 6. https://doi.org/10.1073/pnas.2026805118

[40] C. D. Meyer. 2000. *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics, USA. https://books.google.dk/books?id=HoNgdpJWnWMC

[41] Prakash Murali, David C. Mckay, Margaret Martonosi, and Ali Javadi-Abhari. 2020. Software mitigation of crosstalk on noisy intermediate-scale quantum computers. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'20)*. ACM, Lausanne, Switzerland, 1001–1016. https://doi.org/10.1145/3373376.3378477

[42] Anne E. B. Nielsen, Asa S. Hopkins, and Hideo Mabuchi. 2009. Quantum filter reduction for measurement-feedback control via unsupervised manifold learning. *New Journal of Physics* 11, 10 (2009), 105043.

[43] Michael A. Nielsen and Isaac L. Chuang. 2012. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, Cambridge. https://doi.org/10.1017/cbo9780511976667

[44] Philipp Niemann, Robert Wille, D. Michael Miller, Mitchell A. Thornton, and Rolf Drechsler. 2016. QMDDs: Efficient quantum function representation and manipulation. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 35, 1 (2016), 86–99.

[45] Alexey Ovchinnikov, Isabel Pérez Verona, Gleb Pogudin, and Mirco Tribastone. 2021. CLUE: Exact maximal reduction of kinetic models by constrained lumping of differential equations. *Bioinformatics* 37, 19 (08 2021), 3385–3385.

[46] George J. Pappas, Gerardo Lafferriere, and Shankar Sastry. 2000. Hierarchically consistent control systems. *IEEE Trans. Automat. Contr.* 45, 6 (2000), 1144–1160.

[47] Nils Quetschlich, Lukas Burgholzer, and Robert Wille. 2022. MQT bench: Benchmarking software and design automation tools for quantum computing. (2022). arXiv:2204.13719 MQT Bench is available at https://www.cda.cit.tum.de/mqtbench/

[48] Clarence W. Rowley, Igor Mezič, Shervin Bagheri, Philipp Schlatter, and Dan S. Henningson. 2009. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics* 641 (2009), 115–127.

[49] Aaron Sander, Lukas Burgholzer, and Robert Wille. 2023. Towards Hamiltonian simulation with decision diagrams. (2023). https://doi.org/10.48550/ARXIV.2305.02337

[50] Davide Sangiorgi. 2011. *Introduction to Bisimulation and Coinduction*. Cambridge University Press, Cambridge. https://doi.org/10.1017/cbo9780511777110

[51] Peter W. Shor. 1999. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* 41, 2 (1999), 303–332.

[52] Michael Sipser. 1996. Introduction to the theory of computation. *ACM SIGACT News* 27, 1 (1996), 27–29.

[53] J. Sproston and S. Donatelli. 2006. Backward bisimulation in Markov chain model checking. *Software Engineering, IEEE Transactions on* 32, 8 (Aug. 2006), 531–546.

[54] Damian S. Steiger, Thomas Häner, and Matthias Troyer. 2018. ProjectQ: An open source software framework for quantum computing. *Quantum* 2 (2018), 49.

[55] Stefano Tognazzi, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. 2017. EGAC: A genetic algorithm to compare chemical reaction networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '17)*. ACM, Berlin, Germany, 8. https://doi.org/10.1145/3071178.3071265

[56] Stefano Tognazzi, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin. 2018. Backward invariance for linear differential algebraic equations. In *2018 IEEE Conference on Decision and Control (CDC'18)*. IEEE, Miami, FL, USA, 3771–3776. https://doi.org/10.1109/CDC.2018.8619710

[57] Alison S. Tomlin, Genyuan Li, Herschel Rabitz, and János Tóth. 1997. The effect of lumping and expanding on kinetic differential equations. *SIAM J. Appl. Math.* 57, 6 (1997), 1531–1556.

[58] Max Tschaikowski and Mirco Tribastone. 2014. Tackling continuous state-space explosion in a Markovian process algebra. *Theor. Comput. Sci.* 517 (2014), 1–33.

[59] Max Tschaikowski and Mirco Tribastone. 2017. Spatial fluid limits for stochastic mobile networks. *Perform. Evaluation* 109 (2017), 52–76.

[60] Guifré Vidal. 2003. Efficient classical simulation of slightly entangled quantum computations. *Phys. Rev. Lett.* 91 (Oct. 2003), 147902. Issue 14.

[61] Benjamin Villalonga, Sergio Boixo, Bron Nelson, Christopher Henze, Eleanor Rieffel, Rupak Biswas, and Salvatore Mandrà. 2019. A flexible high-performance simulator for verifying and benchmarking quantum circuits implemented on real hardware. *npj Quantum Information* 5, 1 (2019), 86.

[62] Robert Wille, Lucas Berent, Tobias Forster, Jagatheesan Kunasaikaran, Kevin Mato, Tom Peham, Nils Quetschlich, Damian Rovara, Aaron Sander, Ludwig Schmid, et al. The MQT Handbook: A summary of design automation tools and software for quantum computing. In *Int'l. Conf. on Quantum Software* (2024). arXiv:2405.17543

[63] Robert Wille, Stefan Hillmich, and Lukas Burgholzer. 2022. Tools for quantum computing based on decision diagrams. *ACM Transactions on Quantum Computing* 3, 3, Article 13 (June 2022), 17 pages.
[64] Mingsheng Ying and Yuan Feng. 2018. Model checking quantum systems — A survey. (2018). https://doi.org/10.48550/ARXIV.1807.09466
[65] Mingsheng Ying, Yangjia Li, Nengkun Yu, and Yuan Feng. 2014. Model-checking linear-time properties of quantum systems. *ACM Transactions on Computational Logic* 15, 3 (July 2014), 1–31. https://doi.org/10.1145/2629680
[66] Alwin Zulehner and Robert Wille. 2019. Advanced simulation of quantum computations. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 38, 5 (2019), 848–859.

Received 9 May 2024; revised 14 October 2024; accepted 25 December 2024