Technische Universität München
Max-Planck-Institut für Quantenoptik

Fakultät für Mathematik
Lehrstuhl für Mathematische Physik

Quantum algorithms for quantum many-body systems and small quantum computers

Yi-Min Ge

Vollständiger Abdruck der von der Fakultät für Mathematik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende:
Prof. Dr. Simone Warzel

Prüfende der Dissertation:
1. Prof. Dr. Michael M. Wolf
2. Hon.-Prof. Dr. J. Ignacio Cirac
3. Prof. Dr. Harry M. Buhrman

Die Dissertation wurde am 05.12.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Mathematik am 16.03.2020 angenommen.

Technical University of Munich
Max Planck Institute of Quantum Optics

Department of Mathematics
Chair of Mathematical Physics

Quantum algorithms for quantum many-body systems and small quantum computers

Yi-Min Ge

Full imprint of the dissertation approved by the Department of Mathematics of the Technical University of Munich to obtain the academic degree of

Doctor of Natural Sciences (Dr. rer. nat.)

Chair:
Prof. Dr. Simone Warzel

Examiners of the dissertation:

1. Prof. Dr. Michael M. Wolf
2. Hon.-Prof. Dr. J. Ignacio Cirac
3. Prof. Dr. Harry M. Buhrman

The dissertation was submitted to the Technical University of Munich on 05.12.2019 and was accepted by the Department of Mathematics on 16.03.2020.

Zusammenfassung

Diese Dissertation beschäftigt sich mit Quantencomputing im Kontext von Quantenvielteilchensystemen und größenlimitierten Quantencomputer. Zunächst betrachten wir die wichtigsten Eigenschaften von Grundzuständen lokaler Hamiltonians mit Spektrallücke aus der Sicht des Quantencomputing. Danach behandeln wir mehrere Quantenalgorithmen zur Präparierung relevanter Quantenvielteilchenzustände auf einem Quantencomputer. Abschließend erforschen wir Zugänge zu quanten-klassischen Hybridalgorithmen falls nur Quantencomputer mit einer beschränkten Anzahl von Qubits verfügbar sind.

Abstract

This dissertation studies quantum computing in the context of quantum many-body systems and size-limited quantum computers. First, we discuss the main properties of ground states of local gapped Hamiltonians from a quantum computing perspective. We then consider several quantum algorithms for preparing relevant quantum many-body states on a quantum computer. Finally, we explore approaches to hybrid quantum-classical algorithms when only quantum computers constrained in the number of qubits are available.

Acknowledgements

There is a long list of people whose support and contributions were invaluable to the completion of this dissertation. First of all, I am indebted to Ignacio Cirac for the opportunity to work in the MPQ Theory group as well as his extensive support and guidance throughout my PhD. His enthusiasm for the field has been a great source of inspiration during my time at MPQ.

Second, I am grateful to Michael Wolf for accepting me into M5 as well as his regular help and advice on many different issues over the years. This thesis benefited immensely from the opportunity to interact with him and his group on a regular basis.

I would like to thank Harry Buhrman for agreeing to act as a referee for this thesis. I would also like to thank Simone Warzel for agreeing to chair the committee.

Furthermore, I would like to thank all my collaborators who I have had the pleasure of working with over the course of this PhD. In particular, I would like to thank all my co-authors, Andras Molnar, Ignacio Cirac, Jens Eisert, Jordi Tura, Norbert Schuch, and Vedran Dunjko, of all the articles this thesis is based on.

I would moreover like to thank all my current and former colleagues at the MPQ Theory group for providing an interactive and cooperative research environment. In particular, I would like to thank my office mates, Henrik Dreyer and Julian Roos, for the pleasant working atmosphere. Special thanks also go to Andrea Kluth and Regina Jasny for all their help on administrative matters. I would moreover like to thank former and current members of M5 for numerous interesting discussions and exchanges over the years.

I am especially grateful to all my friends and family who have supported me in various ways during this PhD and have brought balance to my life. Most importantly, I owe my deepest gratitude to my parents without whom this thesis would not have been possible. Finally, I would like to thank Sophia Michael for the endless support, patience and encouragement.

List of contributed articles

This thesis is based on the following articles:

Core articles as principal author

I) Yimin Ge, András Molnár, and J. Ignacio Cirac.
Rapid Adiabatic preparation of injective projected entangled pair states and Gibbs states.
Physical Review Letters, 116, 080503, 2016.
(See also article [1] in the bibliography)

II) Yimin Ge and Jens Eisert.
Area laws and efficient descriptions of quantum many-body states.
New Journal of Physics, 18, 083026, 2016.
(See also article [2] in the bibliography)

III) Yimin Ge, Jordi Tura, and J. Ignacio Cirac.
Faster ground state preparation and high-precision ground energy estimation with fewer qubits.
Journal of Mathematical Physics, 60, 022202, 2019.
(See also article [3] in the bibliography)

Further articles as principal author currently under review

IV) Yimin Ge and Vedran Dunjko.
A hybrid algorithm framework for small quantum computers with application to finding Hamiltonian cycles.
arXiv:1907.01258 [quant-ph], 2019
Submitted to *Journal of Mathematical Physics* July 2019.
(See also article [4] in the bibliography)

Further articles as co-author

V) Andras Molnar, Yimin Ge, Norbert Schuch, and J. Ignacio Cirac.
A generalization of the injectivity condition for projected entangled pair states.
Journal of Mathematical Physics, 59, 021902, 2018.
(See also article [5] in the bibliography)

VI) Vedran Dunjko, Yimin Ge, and J. Ignacio Cirac.
Computational speedups using small quantum devices.
Physical Review Letters, 121, 250501, 2018.
(See also article [6] in the bibliography)

I, Yimin Ge, am the principal author of Articles I, II, III, and IV.

Other articles not included in this thesis

VII) Giannicola Scarpa, Andras Molnar, Yimin Ge, Juan Jose Garcia-Ripoll, Norbert Schuch, David Perez-Garcia, Sofyan Iblisdir.
Computational complexity of PEPS zero testing.
arXiv:1802.08214 [quant-ph], 2018
(See also article [7] in the bibliography)

Contents

1 Introduction

Quantum computers are generally expected to revolutionise the way many computational problems can be solved. Their impact, however, critically depends on the performance and resource requirements of the underlying algorithms to be run on such devices, particularly in the foreseeable future. Indeed, although the realisation of fully scalable quantum computers is still remote, small devices that are limited in size may be achievable soon [8]. Finding ways to exploit such constrained devices remains an important task.

One particularly promising application is the simulation of large quantum systems, as originally envisioned by Feynman [9]. Such problems are usually classically intractable, largely due to the exponential scaling of the dimension of the underlying state space. Quantum computers do not inherently suffer from this representability issue, and as such even small devices could in principle be able to solve certain simulation tasks for small yet interesting system sizes which are beyond the reach of classical computers.

The complementary approach to solving small problem sizes on size-limited quantum computers is to expand the applicability of these devices to larger instances. Since established quantum algorithms require more space than is physically available in this scenario, such approaches necessitate the development of so-called “hybrid algorithms”, which combine quantum and classical computational resources.

This dissertation contributes to both of these areas.

1.1 Outline of this dissertation

In the remainder of this chapter, we briefly summarise all articles included in this dissertation. The subsequent chapters provide a short overview of the main concepts in quantum computing, quantum many-body systems, and hybrid algorithms, which are used and referred to in the articles included in this dissertation, and moreover contextualise the results of the contributed articles in the wider field. Specifically, Chapter 2 provides a short introduction to the basic concepts of quantum computing. Chapter 3 discusses important notions in quantum many-body systems, and analyses general criteria of quantum many-body states from the perspective of computational tractability. Chapter 4 covers some important quantum algorithms which are often used as building blocks in other quantum algorithms. Chapter 5 explores several quantum algorithms to prepare certain quantum many-body states on quantum computers. Chapter 6 presents new approaches for designing hybrid quantum-classical algorithms for the scenario when the sizes of the available quantum computers are severely limited.

After this overview, the contributed articles are included. The published Core Articles I, II, and III, of which the author of this thesis is the principal author, are included in Appendix A. Contributed Article IV, which is currently under review and of which the author of this thesis is the principal author, is included in Appendix B. Afterwards, in Appendix C, Contributed Articles V and VI, of which the author of this thesis is a co-author, are included. Every article is preceded by a short summary, a statement of the individual contribution of the author of this thesis to the respective article, as well as the permission to include it in this thesis.

1.2 Summary of results

The articles included in this dissertation consider different aspects of quantum computation in the context of quantum many-body systems and early quantum computers, and can loosely be classified into two categories. The first category explores possibilities but also limitations of quantum computation in the context of quantum many-body systems. Specifically, Articles I and III propose new quantum state preparation algorithms, while Article V introduces and analyses a new class of quantum many-body states, which, amongst other things, can also be prepared with the algorithm presented in Article I. Article II on the other hand exhibits some limitations of characterising the computational tractability of quantum many-body states by simply looking at their entanglement properties. The second category proposes new approaches to hybrid quantum-classical algorithms that utilise quantum computers significantly smaller than the problem size to enhance the performance of the purely classical version of the algorithms. Specifically, Article IV introduces general frameworks and tools for designing such hybrid algorithms and applies them to speed up the solving of a graph theory problem, while Article VI achieves this in the context of Boolean satisfiability.

Core articles as principal author

- *Article I [1]: Rapid adiabatic preparation of injective projected entangled pair states and Gibbs states*
  In this article, we propose an adiabatic algorithm to prepare injective projected entangled pair states and purifications of Gibbs states of quantum many-body Hamiltonians with commuting local terms. We prove that the algorithm is very efficient if a so-called "uniform gap" condition is satisfied. This efficiency is achieved by making strong use of the locality of the parent Hamiltonians involved. First, we employ a version of the adiabatic theorem whose runtime scales only polylogarithmically in the allowed distance to the target state. This is achieved through a smooth reparameterisation of the adiabatic path. Second, the polylogarithmic scaling of the runtime in the allowed error allows for a sequence of adiabatic paths in which only few local terms are changed at a time. As the final ingredient of the algorithm, we prove that having only such local changes along the path of frustration-free Hamiltonians implies that their terms can in fact be truncated at a short distance away from the support of the local changes. This result, whose proof utilises Lieb-Robinson bounds, implies a low runtime overhead when the adiabatic path is converted into a quantum circuit using Hamiltonian simulation.

- *Article II [2]: Area laws and efficient descriptions of quantum many-body states*
  In this article, we disprove a common folklore conjecture, namely that quantum many-body states exhibiting so-called "area laws" are, in some sense, easy to describe. We prove the existence of quantum many-body states on two-dimensional square lattices which exhibit an area law in an extremely strong sense, but yet cannot be well-approximated by any state which has a polynomial classical description. The latter class is extremely broad, as it is only defined via a polynomial Kolmogorov complexity of the coefficients of the states, and in particular includes efficient tensor networks and polynomial quantum circuits. We also show that the result remains true even if more "physical" conditions, such as translational and rotational invariance, are imposed on the states, and we moreover prove a variant of this result with decaying correlations. The proof is non-constructive and uses a counting argument of ϵ-nets.

- *Article III [3]: Faster ground state preparation and high-precision ground energy estimation with fewer qubits*
  In this article, we present general-purpose quantum algorithms for extracting the ground state of a quantum Hamiltonian from a given trial state by projecting the latter onto its ground state component. We provide different versions of the algorithm, depending on whether or not the ground energy is known beforehand. In case of an unknown ground energy, our algorithms can also be used to estimate its value to a very high precision. We prove that our algorithms are significantly faster than phase estimation, namely exponentially better in the allowed error and polynomially better in the overlap of the trial state with the ground state. Moreover, we show that compared to other methods with comparable runtimes, our algorithms use significantly fewer qubits, making them more suitable for early quantum devices. We moreover show how to combine this approach with phase estimation to optimise the runtime dependence on the spectral gap of the target Hamiltonian. The algorithms are based on tools developed in the context of solving the quantum linear systems problem, and implement an approximate ground state projector using the "linear combinations of unitaries lemma".

Further articles as principal author currently under review

- *Article IV [4]: A hybrid algorithm framework for small quantum computers with application to finding Hamiltonian cycles*
  In this article, we propose a general framework for developing hybrid quantum-classical algorithms which speed up classical divide-and-conquer algorithms using only quantum computers with significantly fewer qubits than the problem size. This framework, which generalises the approach of Article VI, comprises two parts. First, we develop the so-called "divide-and-conquer hybrid approach", which takes a classical divide-and-conquer algorithm and replaces the recursive call with a faster quantum algorithm once the problem size, measured by a suitable effective problem size metric, is sufficiently small to fit on the number of available qubits. We prove sufficient conditions on the time- and space-efficiency of the replacing quantum algorithm for obtaining a polynomial speedup. The result also establishes a trade-off between the resource requirements of that quantum algorithm, the speedup obtained, and the number of qubits available. The second part is a general procedure to efficiently generate suitably space-efficient encodings of large sets in a way that is compatible with a polynomial speedup in the divide-and-conquer hybrid approach. This procedure bridges the gap between reversibility, space-efficiency, and low computational overhead of the uncomputation operations involved. As an application, all these results are then used to speed up Eppstein's algorithm for finding Hamiltonian cycles on cubic graphs, and the general framework reduces this task to finding time- and space-efficient implementations of only a small number of graph-theoretic operations.

Further articles as co-author

- *Article V [5]: A generalization of the injectivity condition for projected entangled pair states*
  In this article, we introduce a new class of projected entangled pair states (PEPS), which we term "semi-injective PEPS". These states are obtained through local invertible operators acting on a torus of plaquette states. This class extends the well-understood notion of "injective PEPS". We show that many important states for which no injective PEPS description exists can naturally be described by a semi-injective PEPS. Similar to

injective PEPS, we also construct frustration-free local parent Hamiltonians for which these states are the unique ground states. We then derive a "fundamental theorem" for semi-injective PEPS by showing that two tensors generate the same class of semi-injective PEPS if and only if they are related by a matrix product operator with certain properties acting on the boundary. This result then allows for a characterisation of symmetries in semi-injective PEPS. Finally, we use these results to rederive the third cohomology classification of symmetry protected topological phases within the framework of semi-injective PEPS.

- *Article VI [6]: Computational speedups using small quantum devices*
  In this article, we propose a quantum enhancement of Schöning's algorithm for 3SAT using only significantly fewer qubits than the number of variables involved in the 3SAT instance. The speedup is polynomial for any arbitrarily small constant ratio of the number of available qubits to the number of variables. We demonstrate that straightforward hybrid quantum-classical algorithms using such devices encounter a "threshhold effect", meaning that the hybrid algorithm becomes slower than the original classical algorithm if that ratio is below some constant value. We then develop a new approach based on a reformulation of Schöning's original algorithm as a derandomised divide-and-conquer algorithm, which reduces 3SAT to solving the so-called "Promise-Ball-SAT" (PBS) problem. We achieve the speedup by replacing the recursive call with a suitably time- and space-efficient quantum algorithm sufficiently deep in the recursion tree. The faster runtime of the replacing quantum subroutine is obtained by rewriting a classical recursive PBS algorithm in non-recursive form, and speeding it up using amplitude amplification. In order to be able to do this sufficiently space-efficiently, we develop efficiently uncomputable data structures to store sets of "flipped" variable indices.

2 Quantum computing

In this chapter, we provide a brief introduction to the main concepts of quantum computing. This material can also be found in introductory textbooks such as Ref. [10].

Throughout this dissertation, the computer science convention of the big-O notation is used. We write $f(n) = O(g(n))$ if there exist $c, n_0 > 0$ such that for all $n > n_0$, $f(n) \leq cg(n)$. We moreover write $f(n) = \Omega(g(n))$ if there exist $c, n_0 > 0$ such that for all $n > n_0$, $f(n) \geq cg(n)$. We also write $f(n) = \Theta(g(n))$ if both $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. The O, Ω, and Θ notation are generalised in the obvious way to multiple variables, as well as the case when some of the variables are considered in the limit of sufficiently small instead of sufficiently large values (it will always be clear from the context which limits are considered for which variables).

Unless stated otherwise, all logarithms in this dissertation are with respect to base 2.

2.1 Quantum states

Associated to any isolated physical quantum system is a complex Hilbert space, known as the *state space*. Throughout this dissertation, only the finite-dimensional case $\mathbb{C}^d$ with $d \in \mathbb{N}$ is relevant. Quantum systems with $d = 2$ are called *qubits*.

A *pure quantum state* of this system is an l_2-normalised vector in the state space, denoted as $|\psi\rangle \in \mathbb{C}^d$. The dual to $|\psi\rangle$ is written as $\langle\psi|$, and the inner product between states $|\psi\rangle$ and $|\phi\rangle$ as $\langle\psi|\phi\rangle$. Often, an orthonormal basis $\{|0\rangle, \ldots, |d-1\rangle\}$, called the *computational basis*, is singled out. In this basis, $|\psi\rangle$ is represented as

$$|\psi\rangle = \sum_{x=0}^{d-1} \psi_x |x\rangle , \tag{2.1}$$

where $\psi_x = \langle x|\psi\rangle \in \mathbb{C}$ with $\sum_x |\psi_x|^2 = 1$.

The state space associated to composite quantum systems is given by the tensor product of the individual state spaces. For simplicity of notation, tensor product states $|\psi\rangle \otimes |\phi\rangle$ are often abbreviated as $|\psi\rangle|\phi\rangle$ or $|\psi\phi\rangle$. The computational basis of a composite system is given by the set of all possible tensor products of computational basis states of the individual subsystems.

An important special case in the context of quantum computing is the composite system comprising n qubits, whose state space is $(\mathbb{C}^2)^{\otimes n}$. The computational basis of this system is $\{|x\rangle : x \in \{0,1\}^n\}$.

A *mixed quantum state* of a quantum system is described by a *density operator* ρ, which is a positive operator of unit trace acting on the state space. Often, ρ is given by an ensemble $\{(p_i, |\psi_i\rangle)\}_i$ with $p_i \in [0,1]$ and $\sum_i p_i = 1$, meaning that the quantum system is in the state $|\psi_i\rangle$ with probability p_i. In this case $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$. Density operators can also be obtained as partial states through the partial trace operation from larger systems: if $|\psi_{AB}\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$ is a pure state of a composite system with state spaces $\mathcal{H}_A$ and $\mathcal{H}_B$, respectively, then the density operator $\rho_A = \mathrm{tr}_B(|\psi_{AB}\rangle\langle\psi_{AB}|)$ is called the *reduced state* of $|\psi_{AB}\rangle$ with respect to the subsystem A.

2.2 Quantum circuits

Quantum operations on pure quantum states are unitaries acting on the state space. In the context of quantum computing, when the system is composed of n qubits[1], one usually only considers unitaries that can be obtained as a *quantum circuit*, i.e., a product of so-called *gates*, which are elements of a fixed set of unitaries, acting on any of the n qubits. Moreover, one usually only considers sets of gates which only contain unitaries acting on at most two qubits. The gate set is called *exactly universal* if for all n, any n-qubit unitary can be written as a product of its gates. An example of an exactly universal gate set is the set of all single-qubit unitaries together with the controlled-NOT (or simply CNOT) gate, which in the computational basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ acts as

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2.2}$$

However, it is often preferable to work only with finite gate sets, which clearly cannot be exactly universal. Hence, the notion of universality is usually relaxed to only require good approximations of the target unitaries. In other words, a gate set is called *universal* if for all n, any n-qubit unitary can be approximated to arbitrarily small distance (e.g. in operator norm) by products of its gates.

One of the most commonly used gate sets is

$$\left\{ \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \begin{pmatrix} e^{-i\pi/8} & 0 \\ 0 & e^{i\pi/8} \end{pmatrix}, \mathrm{CNOT} \right\}, \tag{2.3}$$

which can be shown to be universal. The two single-qubit gates in this set are known as the *Hadamard gate* and the *T gate*, respectively. However, many other sets of gates are also known to be universal. Moreover, the *Solovay-Kitaev Theorem* implies that any single qubit unitary can be approximated to distance ϵ in operator norm using $O(\text{polylog}(1/\epsilon))$ Hadamard or T gates. This means that all universal gate sets are essentially equivalent up to polylogarithmic factors.

In quantum computation, one is usually tasked with trying to either approximate a target unitary with a quantum circuit, or alternatively generate an approximation of a target state by applying a quantum circuit to a given initial state (often simply assumed to be $|0\rangle^{\otimes n}$). Moreover, the quantum circuit should be the result of a *quantum algorithm*, which provides descriptions of quantum circuits performing the task for varying problem sizes. As such, we require the circuits to be *uniform*, which means that circuits for different problem sizes should be related in a simple way[2]. This is an important technicality, as the ability to choose arbitrary circuits for each problem size would allow one to solve uncomputable problems. There are several important figures of merit of such quantum algorithms.

The *runtime* or *size* of the quantum circuit is simply the number of its elementary gates. This is the most commonly used metric to measure the efficiency of a quantum algorithm. In particular, one often demands that the runtime scales at most polynomially with the input size.

[1] Sometimes, quantum computations on larger-dimensional individual systems are considered, but these individual systems can be embedded into a system of multiple qubits each.

[2] Formally, uniformity means that there exists a fixed Turing machine which, given a tape containing '1' n times, outputs a description of the n^{th} circuit in time $O(\text{poly}(n))$.

Sometimes, we care not just about the runtime of a circuit, but also about its *depth*, which is given by the maximum number of elementary gates amongst all paths from an input to an output. Roughly speaking, the depth of a circuit measures the runtime if one is allowed to perform gates acting on disjoint subsystems in parallel.

In many cases, quantum algorithms require additional qubits, called *ancillas*. Indeed, the number of gates required to map $|0\rangle^{\otimes n}$ to some $|\psi\rangle \in (\mathbb{C}^2)^{\otimes n}$ might be much higher than the number of gates required to map $|0\rangle^{\otimes(n+m)}$ to $|\psi\rangle|0\rangle^{\otimes m}$ for a given $m \in \mathbb{N}$. Particularly in the context of early quantum computers with a limited number of qubits, the number of ancillas m required by a quantum algorithm may be an important consideration.

2.3 Measurements

Although more general measurement operations are commonly considered in quantum mechanics, in the context of quantum algorithms, one usually only allows projective measurements in the computational basis. If such a measurement is performed on an n-qubit state $|\psi\rangle$ given by

$$|\psi\rangle = \sum_{x\in\{0,1\}^n} \psi_x \, |x\rangle \tag{2.4}$$

with $\sum_x |\psi_x|^2 = 1$, then the outcome $x \in \{0,1\}^n$ is obtained with probability $|\psi_x|^2$ and the state after the measurement will be in $|x\rangle$.

Alternatively, one can choose to perform a measurement on only m of the n qubits. If $|\psi\rangle$ is written as

$$|\psi\rangle = \sum_{x\in\{0,1\}^m} \alpha_x \, |x\rangle|R_x\rangle\,, \tag{2.5}$$

where the $|R_x\rangle$ are $(n-m)$-qubit states and $\sum_x |\alpha_x|^2 = 1$, then a measurement on the first m qubits will with probability $|\alpha_x|^2$ yield the outcome $x \in \{0,1\}^m$ and leave the state in $|x\rangle|R_x\rangle$.

Quantum circuits that utilise measurements will generally only perform a required task with a certain probability. For most applications, this is good enough as long as the success probability is sufficiently large. Especially in the context of decision problems, this has termed the notion of *bounded-error algorithms*, which solve the problem at hand with a probability of at least $2/3$. The value $2/3$ is arbitrary here and could be replaced with any constant in $(1/2, 1)$ because any algorithm with such a probability of success can be lifted to succeeding with probability at least $1-\epsilon$ with $O(\log(1/\epsilon))$ repetitions and a majority vote.

2.4 Reversibility

Since unitary matrices are invertible, any quantum circuit implements a reversible operation. Conversely, for any classical reversible gate performing $x \mapsto f(x)$, there is a quantum gate performing $|x\rangle \mapsto |f(x)\rangle$. However, classical cicuits are often written using non-reversible gates, such as AND and OR. Such computations can nevertheless be rewritten in reversible form, by replacing any irreversible gate computing $x \mapsto f(x)$ with the reversible gate computing $(x, y) \mapsto (x, y \oplus f(x))$, and applying it to $(x, 0)$. In this way, any non-reversible classical circuit computing $x \mapsto F(x)$ can be rewritten as a reversible circuit $\mathcal{C}$ which implements $(x, 0, 0) \mapsto (x, F(x), G(x))$, where $G(x)$ is some unwanted extra output (e.g., the results of all intermediate computational steps). The presence of this "garbage" register is problematic for quantum computation, since it prevents the desired interference properties when we lift this circuit to a quantum circuit. This can be fixed with the following trick, known as *uncomputation*.

We introduce a fourth register onto which we first reversibly "copy" the register containing $F(x)$ using controlled-NOT gates, i.e. $(x, F(x), G(x), y) \mapsto (x, F(x), G(x), y \oplus F(x))$. We then apply the inverse of $\mathcal{C}$. Since $\mathcal{C}$ does not involve the last register, we obtain the state $(x, 0, 0, y \oplus F(x))$, yielding a reversible circuit which implements $(x, 0, 0, y) \mapsto (x, 0, 0, y \oplus F(x))$. In particular, by replacing each reversible elementary gate with the corresponding elementary quantum gate, one obtains a quantum circuit which performs $|x\rangle|0\rangle|0\rangle|0\rangle \mapsto |x\rangle|0\rangle|0\rangle|F(x)\rangle$.

This simple way of turning any non-reversible classical circuit into a reversible and further into a quantum one requires additional ancillas, in the worst case roughly the number of elementary gates in the original non-reversible circuit. More general uncomputation schemes exist [11], which can reduce the number of ancillas required, but usually achieve this at the expense of potentially large runtime overheads.

3 Quantum many-body systems

The physical properties of a quantum system are usually captured by a *Hamiltonian* H, which is a Hermitian operator acting on the state space of the system. Much of low temperature and condensed matter physics is concerned with studying properties of states associated to a given Hamiltonian H. For example, the state of inverse temperature β in thermal equilibrium is described by the *Gibbs state*, $\rho_\beta = e^{-\beta H}/Z$, where $Z = \operatorname{tr} e^{-\beta H}$.

Often, one is interested in the properties of the spectral decomposition of $H = \sum_i E_i |E_i\rangle\langle E_i|$, where the eigenvalues E_i are called the *energies* of the corresponding eigenstates $|E_i\rangle$. Note that since H is Hermitian, $E_i \in \mathbb{R}$ and the $|E_i\rangle$ can without loss of generality be assumed to form an orthonormal basis of the state space. Of particular importance are the ground states corresponding to the lowest energy $\min_i E_i$, known as the *ground energy*. Indeed, at low temperatures, the properties of the ground states usually dominate the behaviour of the Gibbs state. In this thesis, we will mostly be concerned with the case when the ground energy is non-degenerate.

When $H = \sum_x E_x |x\rangle\langle x|$ is diagonal in the computation basis, H describes a classical system, i.e. one without any quantum effects. Then, $\rho_\beta = \frac{1}{Z} \sum_x e^{-\beta E_x} |x\rangle\langle x|$ captures the *Gibbs distribution* given by H.

For quantum many-body systems, additional constraints restrict the structure of Hamiltonians considered. Usually, there is an underlying lattice geometry of N sites of local dimension d, and the Hamiltonians of such systems are restricted to be sums of *local* terms,

$$H = \sum_k h_k, \qquad \|h_k\|_{\mathrm{op}} \leq 1, \tag{3.1}$$

where each of the h_k only acts non-trivially in a region of bounded diameter. For example, if H is a nearest-neighbour Hamiltonian, each h_k is a Hermitian operator acting on two neighbouring sites, tensored with the identity operator on all remaining sites. When all h_k are positive operators and the ground energy of H is 0, H is called *frustration-free*.

3.1 Tensor networks

The fundamental challenge of dealing with quantum many-body systems lies in the exponential scaling of the underlying Hilbert space dimension: the state space associated to a system of N sites of local dimension d is $(\mathbb{C}^d)^{\otimes N}$, which has dimension d^N. Usually, d can be assumed to be a small constant (e.g. $d = 2$ for a system of N qubits on a lattice), whereas N generally needs to be reasonably large to exhibit the relevant macroscopic effects. As such, the asymptotic scalings of computational resources with large N are usually the relevant figures of merit for complexities related to any algorithm for quantum many-body systems. The exponential scaling of the state space dimension is particularly problematic for classical numerical simulations, as even just storing a naive representation of the state using d^N complex parameters would require more memory than physically available even for relatively small values of N.

However, the locality structure of the Hamiltonian also leads to structural restrictions of its ground state, and one could therefore hope that in principle, more efficient descriptions of these states might exist. An important class of such descriptions are *tensor networks*, which define states with generally only $O(\mathrm{poly}(N))$ complex parameters. Such tensor network states are generally conjectured to be able to approximate ground states of local Hamiltonians well. We will define some special cases of such tensor networks now and motivate this conjecture below in Section 3.2.

3.1.1 Matrix product states

The simplest type of tensor network states are *matrix product states* (or *MPS*), which describe quantum many-body states in one spatial dimension. These are states of the form

$$|\psi_{\mathrm{MPS}}\rangle \propto \sum_{i_1,\ldots,i_N=0}^{d-1} \mathrm{tr}\Big(A_{i_1}^{(1)} \ldots A_{i_N}^{(N)}\Big) |i_1 \ldots i_N\rangle\,, \tag{3.2}$$

where the $A_i^{(j)}$ are $D\times D$ matrices for $i = 0, \ldots, d-1$ and $j = 1, \ldots, N$. The value of j labels the sites in an open or closed chain of N particles of local dimension d, and D is the so-called *bond dimension.* Provided that D is not too large (e.g. $D = O(\mathrm{poly}(N))$), this yields a description of the state with only polynomially-many parameters. Moreover, the sequential order of the matrices appearing in the MPS description inherently captures the one-dimensional geometry of the state. For translationally invariant states, one often only considers *translationally invariant MPS*, where the matrices $A_i^{(j)} = A_i$ are site-independent.

MPS are particularly suited for classical numerical simulations of one-dimensional quantum many-body systems. Indeed, physically relevant quantities such as expectation values or correlation functions of local observables can be efficiently calculated classically given only the matrices of the MPS, and the latter can in turn be used as variational parameters to obtain states with low energy. For example, the highly successful Density Matrix Renormalisation Group algorithm [12] can be reformulated in terms of MPS [13]. MPS are generally well understood. For example, "canonical forms" for MPS have been derived which essentially classify all MPS [14]. The central ingredient to these canonical forms is a "fundamental theorem" of MPS, which establishes necessary and sufficient conditions for when two sets of matrices generate the same set of states. This fundamental theorem is particularly strong for translationally invariant MPS, which establishes that two matrices generating the same set of MPS are related by a "gauge" transformation. This result, amongst other things, can be used to classify symmetry-protected topological (SPT) phases in one dimension [15, 16].

3.1.2 Projected entangled pair states

Projected entangled pair states (or *PEPS*) are the natural generalisation of MPS in two and more spatial dimensions [17].

Definition 3.1. Let (V, E) be a simple graph with bounded degree and let D, d be positive integers. For each $v \in V$, let $Q^{(v)} : (\mathbb{C}^D)^{\otimes \deg v} \to \mathbb{C}^d$ be a linear operator. On each edge $e \in E$, place a maximally entangled state $\left|\phi_D^+\right\rangle_e = \sqrt{D^{-1}} \sum_{i=0}^{D-1} |ii\rangle$ of two D-dimensional "virtual" particles located at the two ends of e. Then, the *projected entangled pair state* (or *PEPS*) $\left|\psi_{\mathrm{PEPS}}[\{Q^{(v)}\}_{v\in V}]\right\rangle \in (\mathbb{C}^d)^{\otimes |V|}$ is defined as

$$\left|\psi_{\mathrm{PEPS}}[\{Q^{(v)}\}_{v\in V}]\right\rangle \propto \prod_{v\in V} Q^{(v)} \bigotimes_{e\in E} |\phi_D^+\rangle_e\,, \tag{3.3}$$

where each $Q^{(v)}$ acts on the $\deg v$ virtual particles located at v.

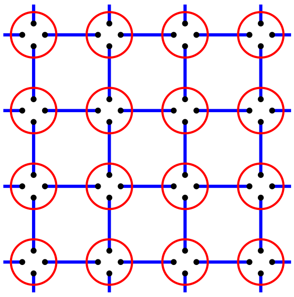


Figure 3.1: PEPS for two-dimensional square lattice. Each physical site comprises four virtual D-dimensional particles. Two neighbouring virtual particles are placed into a maximally entangled pair state (blue). Linear operators (red) supported on the four virtual particles of each site are then applied.

This construction is illustrated in Fig. 3.1 for a two-dimensional square lattice. The operators $Q^{(v)}$ are commonly referred to as *PEPS projectors* (even though they are not necessarily projectors). As with MPS, D is called the *bond dimension* of the PEPS. If $D = O(\text{poly}(N))$, the PEPS projectors provide an efficient description of the state comprising only polynomially many parameters. Moreover, as with MPS, one often only considers translationally invariant PEPS whose projectors $Q^{(v)} = Q$ are site-independent when dealing with translationally invariant systems.

The general theory and classification of PEPS is much less understood than that of MPS. In particular, no "fundamental theorem" analogous to that of MPS exists. In fact, quite the opposite is true: the problem of whether two given PEPS projectors generate the same states for all system sizes can be shown to be undecidable in general [7]. The latter can be proven by encoding two-dimensional tiling problems, which are well-known to be undecidable, into the problem of deciding if two PEPS projectors generate the same set of states.

One important subclass of PEPS are the *injective PEPS*, which are PEPS for which all PEPS projectors $Q^{(v)}$ are invertible.

Definition 3.2. Let (V, E) be a simple graph with bounded degree, D, d be positive integers, and $Q^{(v)} : (\mathbb{C}^D)^{\otimes \deg v} \to \mathbb{C}^d$ be a linear operators. We call the PEPS $\left|\psi_{\text{PEPS}}[\{Q^{(v)}\}_{v\in V}]\right\rangle$ *injective* if $Q^{(v)}$ is invertible for all $v \in V$.

Unlike general PEPS, the class of injective PEPS is well-understood. In particular, a "fundamental theorem" analogous to the one for translationally invariant MPS is known for translationally invariant injective PEPS [18].

In Contributed Article V, the notion of injectivity is generalised. We introduce so-called *semi-injective PEPS*, which are obtained by placing plaquette states of four virtual particles on a two-dimensional lattice and acting with invertible 4-body operators on the virtual particles meeting at each site (see Fig. 3.2). We show that this class of states not only includes all injective PEPS, but also several important examples of states which do not admit an injective PEPS description, such as the AKLT model on square lattices and the CZX model. We show an analogous fundamental theorem for semi-injective PEPS there, and use it to rederive the classification of symmetry-protected topological phases of matter in that framework.

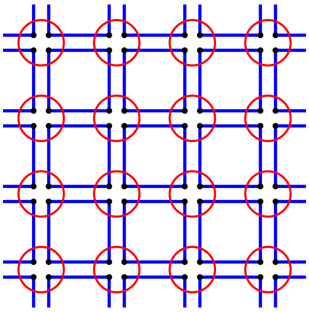


Figure 3.2: Semi-injective PEPS. Invertible 4-body operators (red) act on plaquette states (blue) of four virtual particles.

3.2 Ground states of local Hamiltonians and area laws

In this section, we discuss a heuristic argument for why tensor networks could be suited to capture ground states of local Hamiltonians.

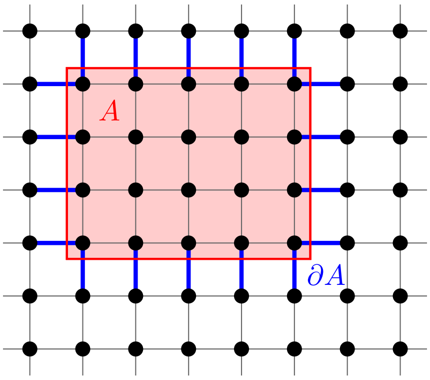


Figure 3.3: Area law. A quantum many-body state $|\psi\rangle$ satisfies an area law with respect to some entanglement entropy S if for any connected subregion A (red) of the lattice, $S(\rho_A) = O(|\partial A|)$, where $\rho_A = \operatorname{tr}_{V\setminus A} |\psi\rangle\langle\psi|$ is the reduced state of $|\psi\rangle$ with respect to the subsystems in A and ∂A is the boundary of A (blue). Generic states satisfy a volume law, $S(\rho_A) = \Omega(|A|)$, with high probability.

Ground states of local Hamiltonians are generally expected to have special entanglement properties. Specifically, for local Hamiltonians defined on a lattice (V, E) with a $\Omega(1)$ spectral gap and a unique ground state, the latter is expected to exhibit a so-called *area law* [19]. This means that certain entanglement entropies, e.g. the *Rényi entropies* $S_\alpha(\rho_A) = \frac{1}{1-\alpha} \log \operatorname{tr} \rho_A^\alpha$, $\alpha > 0$, or the *von Neumann entropy* $S_1(\rho_A) = -\operatorname{tr}(\rho_A \log \rho_A) = \lim_{\alpha\to 1} S_\alpha(\rho_A)$ of reduced states ρ_A with respect to a connected subregion $A \subset V$ of particles, is bounded by $O(|\partial A|)$, where the *boundary* ∂A of A is the set of edges between A and $V\setminus A$ (see Fig 3.3). This property is in stark contrast to generic states. Indeed, it can be shown, as a consequence of the concentration of measure phenomenon, that states picked uniformly at random according to the Haar measure satisfy a *volume law*, i.e. $S_\alpha(\rho_A) = \Omega(|A|)$, with probability close to 1 [20].

This contrast between generic states and ground states of local gapped Hamiltonians has termed the notion of a small "physical corner" of the exponentially-dimensional quantum many-body state space. Moreover, MPS and PEPS with an $O(1)$ bond dimension can be easily seen to fulfil an area law with respect to S_0, the binary logarithm of the Schmidt rank, and hence also with respect to S_α for all $\alpha > 0$. The fact that MPS/PEPS and ground states of local gapped Hamiltonians are conjectured to share this very non-generic property provides a heuristic justification for why such tensor network states could be suited to describe ground states of local gapped Hamiltonians.

In one spatial dimension, i.e. an open or closed chain of N particles of local dimension d, area laws – which in 1D means that $S(\rho_A)$ is bounded by a constant for all connected subchains $A \subset V$ independently of the length of A – have been proven for the unique ground states of local gapped Hamiltonians [21, 22]. Moreover, the precise link between area laws and the existence of efficient MPS descriptions has also been proven in 1D. Specifically, it has been shown that any state which satisfies a 1D area law for S_α with $\alpha \in (0, 1)$ can be approximated by an MPS with small bond dimension [23]. The result remains true even if the constant scaling of $S_\alpha(\rho_A)$ required by the 1D area law is relaxed to a logarithmic scaling in the system size.

Theorem 3.3 ([23, 24])**.** *Let $|\psi\rangle \in (\mathbb{C}^d)^{\otimes N}$ be a state of N quantum systems of local dimension d. For $L = 1, \dots, N-1$, let ρ_L be the reduced state of $|\psi\rangle$ with respect to the first L quantum systems according some pre-specified total order of the N quantum systems. Suppose that there exists some $\alpha \in (0, 1)$ such that for all $L \in \{1, \dots, N-1\}$, $S_\alpha(\rho_L) = O(\log N)$. Then, for all $\epsilon > 0$, there exists an MPS $|\psi_{MPS}\rangle$ of bond dimension $O((N/((1-\alpha)\epsilon))^{\alpha/(1-\alpha)} \operatorname{poly}(N))$ such that $\| |\psi\rangle - |\psi_{MPS}\rangle \|_2^2 < \epsilon$.*

To be precise, the proofs in Refs. [21, 22] of the area law for ground states of local gapped Hamiltonians in 1D only explicitly establish an area law for the von Neumann entropy. However, Theorem 3.3 breaks down for $\alpha = 1$ and moreover, it has been shown that merely looking at area laws with respect to the von Neumann entropy turns out to be inconclusive in general [24]. Hence, strictly speaking, the established versions of the area laws for ground states of gapped Hamiltonians in 1D do not a priori imply the existence of a good MPS approximation with low bond dimension. However, the proof of the 1D area law in [22] using "approximate ground state projectors" actually also proves that all ground states of local gapped Hamiltonians in 1D can be captured by MPS of polynomial bond dimension. This proven connection ultimately also gave rise to provably efficient classical algorithms to find the ground state of gapped local Hamiltonians in 1D in terms of MPS [25–27].

In two (or more) spatial dimensions, the picture is more complicated. While it has been proven that analogues of PEPS for mixed states with polynomial bond dimension can well approximate Gibbs states of local gapped Hamiltonians [28], the corresponding claim for ground states is so far only a (generally believed) conjecture. As a first step to proving it, significant effort has gone into attempting to prove an area law for ground states of local gapped Hamiltonians in 2D. This has been proven in many special cases, including frustration-free Hamiltonians [29], Hamiltonians in the same gapped phase as other Hamiltonians with area law ground states [30, 31], or under certain assumptions on the scaling of the specific heat capacity [32]. However, a general area law in 2D for ground states of local gapped Hamiltonians is still unproven.

Yet, even if a general area law were to be proven, care has to be taken with this approach: In Core Article II, we show that the analogous statement to Theorem 3.3 does not hold in more than one spatial dimensions: there exist states which satisfy an area law in an extremely strong sense, namely with respect to S_0 (and hence with respect to S_α for all $\alpha > 0$), but nevertheless cannot be approximated by any efficient tensor network description. In fact, the result is more general and does not only apply to tensor networks, but any efficient classical description of the state in the sense of having a low Kolmogorov complexity. This in particularly also implies that 2D area law states cannot in general be prepared by polynomial quantum circuits, even when allowing for post-selections of measurement results. Note that the existence of these counterexamples has no implications for properties of ground states of gapped Hamiltonians – in fact, the counterexamples established in Core Article II are provably not eigenstates of any local Hamiltonian. Yet, it signifies that the characterisation of the “physical corner” of the many-body Hilbert space via low entanglement properties is incomplete, and that area laws can only serve as *intuitive* guidelines for the ability to capture the state by PEPS.

4 Important quantum algorithms

In this chapter, we present some elementary techniques and quantum algorithms which are often used as subroutines or building blocks in subsequent quantum algorithms.

4.1 Amplitude amplification

A common way to obtain (up to quadratic) polynomial speedups over classical algorithms is to use *amplitude amplification* techniques, first introduced in Ref. [33]. Suppose that $\mathcal{C}$ is a quantum circuit on n qubits which prepares the state $\mathcal{C}\,|0\rangle^{\otimes n} = \lambda\,|G\rangle + \sqrt{1-\lambda^2}\,|B\rangle$ with $\langle B|G\rangle = 0$ and $\lambda \in (0,1)$, and let U be a unitary on $n+1$ qubits that satisfies $U\,|G\rangle|b\rangle = |G\rangle|1-b\rangle$ and $U\,|B\rangle|b\rangle = |B\rangle|b\rangle$ for $b \in \{0,1\}$. Here, $\mathcal{C}$ can be thought of as preparing a state which has a desired "good" component $|G\rangle$ (e.g., $|G\rangle$ could be the target state which encodes the solution to the problem we are trying to solve) and an undesired "bad" component $|B\rangle$. The unitary U acts as an "oracle" which recognises good states. The naive way to obtain a good state would be to run $U\mathcal{C}$ on $|0\rangle^{\otimes(n+1)}$, followed by a measurement of the last qubit. Then, a good state is found with probability λ^2. If λ is very small (which is usually the case in most practical applications), then $\Omega(1/\lambda^2)$ repetitions of this procedure are required to obtain a bounded-error algorithm.

Amplitude amplification techniques can be used to reduce this to $O(1/\lambda)$ repetitions. To illustrate the simplest case, suppose that the value of λ is known, and we write $\lambda = \sin\theta$ for some small $\theta > 0$. Let R_0 be the n-qubit unitary which satisfies $R_0\,|0\rangle^{\otimes n} = -\,|0\rangle^{\otimes n}$ and $R_0\,|x\rangle = |x\rangle$ for $x \in \{0,1\}^n \backslash \{0\dots 0\}$, and U' be an n-qubit unitary which satisfies $U'\,|G\rangle = -\,|G\rangle$ and $U'\,|B\rangle = |B\rangle$ (note that U' can be obtained from U using one ancilla in $(|0\rangle - |1\rangle)/\sqrt{2}$). Then, it is easy to see that in the two-dimensional subspace $\mathrm{span}_{\mathbb{R}}\{|G\rangle, |B\rangle\}$, the operator $R = \mathcal{C}R_0\mathcal{C}^\dagger U'$ acts as a rotation by 2θ. In particular, since θ is small, $O(1/\theta) = O(1/\lambda)$ repetitions of R will rotate $\mathcal{C}\,|0\rangle^{\otimes n}$ to a state with an $\Omega(1)$ overlap with $|G\rangle$.

In the above illustration, knowing the value of λ was crucial, since applying R too many times will "overshoot" the target state and reduce the overlap with $|G\rangle$ again. An improvement over this simple amplitude amplification method, known as *fixed point search*, generalises this result to the case when only a lower bound on λ is known.

Theorem 4.1 ([34])**.** *Let $\mathcal{C}$ be a quantum circuit on n qubits such that $\mathcal{C}\,|0\rangle^{\otimes n} = \lambda\,|G\rangle + \sqrt{1-\lambda^2}\,|B\rangle$ with $\langle B|G\rangle = 0$ and $\lambda \in (0,1)$, and let U be a unitary on $n+1$ qubits that satisfies $U\,|G\rangle|b\rangle = |G\rangle|1-b\rangle$ and $U\,|B\rangle|b\rangle = |B\rangle|b\rangle$ for $b \in \{0,1\}$. Let $\lambda', \delta \in (0,1)$ with $\lambda' \leq \lambda$. Then, there exists a quantum circuit $\mathrm{FPS}(\mathcal{C}, U, \lambda', \delta)$ on $n+1$ qubits using $O(\log(1/\delta)/\lambda')$ calls to $\mathcal{C}, \mathcal{C}^\dagger, U$ and $O(n^2\log(1/\delta)/\lambda')$ additional gates such that*

$$|\langle G,0|\mathrm{FPS}(\mathcal{C}, U, \lambda', \delta)|0\rangle^{\otimes(n+1)}|^2 \geq 1-\delta^2. \tag{4.1}$$

In Core Article III, we amongst other things also prove a kind of converse of Theorem 4.1, namely that the left hand side of (4.1) is at most $2\ln(2/\delta)\lambda/\lambda'$ if $\lambda \leq \lambda'$, and use this to propose a quantum algorithm to find the smallest ancilla "label" amongst terms with at least some given amplitude in a given superposition.

4.2 Hamiltonian simulation

The dynamics of a quantum state in a closed system described by a Hamiltonian H is governed by the Schrödinger equation[1],

$$i\frac{\mathrm{d}}{\mathrm{d}t}\left|\psi\right\rangle = H\left|\psi\right\rangle. \tag{4.2}$$

The problem of simulating this evolution of a given Hamiltonian and initial state on a quantum computer is known as *Hamiltonian simulation.* For time-independent Hamiltonians H, Eq. (4.2) implies that the state $\left|\psi(t)\right\rangle$ at time t is simply given by

$$\left|\psi(t)\right\rangle = e^{-iHt}\left|\psi(0)\right\rangle, \tag{4.3}$$

so that Hamiltonian simulation essentially amounts to approximating the unitary e^{-iHt} by a short quantum circuit.

Many efficient quantum algorithms for this task are known. The earliest and simplest of these algorithms [35] is based on "trotterisation" techniques. Suppose that H acts on n qubits and can be decomposed into $M = O(\mathrm{poly}(n))$ "local" terms, $H = \sum_{k=1}^{M} h_k$, where each h_k only acts non-trivially on $O(1)$ qubits. Then, using the Trotter formula [36, 37]

$$e^{-iHt} = \left(e^{-ih_1 t/r}\dots e^{-ih_M t/r}\right)^r + O(t^2/r), \tag{4.4}$$

with $r = \Omega(t^2/\epsilon)$ and approximating each $e^{-ih_k t/r}$ as a quantum circuit with $O(\mathrm{polylog}(Mr/\epsilon))$ gates, one obtains a quantum circuit that approximates e^{-iHt} to error $O(\epsilon)$ using at most $O(Mt^2\epsilon^{-1}\,\mathrm{polylog}(M, t, \epsilon^{-1}))$ gates.

In recent years, Hamiltonian simulation algorithms with significantly better runtimes have been developed for various input models of H [38–41]. Here, we exemplarily state one of these results, whose runtime scaling can be shown to be essentially optimal [39]:

Theorem 4.2 ([41])**.** *Let H be an $N \times N$ Hermitian matrix such that there is an $Nd \times Nd$ unitary U and a $d \times d$ unitary $\mathcal{G}$ such that $H = (\langle G| \otimes \mathbb{1})U(|G\rangle \otimes \mathbb{1})$, where $|G\rangle = \mathcal{G}\,|0\rangle \in \mathbb{C}^d$. Let $t > 0$. Then, e^{-iHt} can be simulated to error ϵ and failure probability $O(\epsilon)$ using at most $\lceil \log_2 N\rceil + \lceil \log_2 d\rceil + 2$ qubits, $O(t + \log(1/\epsilon))$ calls to controlled-$\mathcal{G}$, controlled-U, and their inverses, and $O(t \log d + \log(1/\epsilon)\log d)$ additional elementary gates.*

In a certain sense, Hamiltonian simulation can be considered the most fundamental problem in quantum computing. Indeed, it is easy to see that every quantum circuit can be implemented by a series of Hamiltonian evolutions and thus, the problem is BPQ-hard, where BQP is the class of decision problems that can be solved in polynomial time with bounded error on a quantum computer.

4.3 Phase estimation

Suppose that we are given a description of a unitary U as a quantum circuit and a copy of an eigenstate $|\lambda\rangle$ of U with $U|\lambda\rangle = e^{2\pi i\lambda}|\lambda\rangle$ and $\lambda \in [0, 1)$. The *phase estimation* algorithm, first introduced in Ref. [42], provides a way to obtain an estimate of the unknown value of λ given access to a controlled version of U, i.e. $\mathrm{c}U = |0\rangle\langle 0| \otimes \mathbb{1} + |1\rangle\langle 1| \otimes U$.

[1] In quantum mechanics literature, the Schrödinger equation is often given with an additional multiplicative constant $\hbar$, which for convenience we have simply normalised to $\hbar = 1$ since its precise value does not matter here.

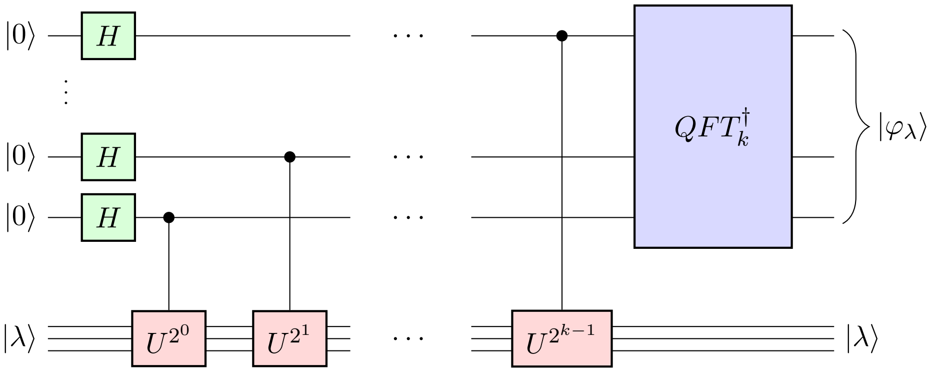


Figure 4.1: Phase estimation. The algorithm uses k ancilla qubits, k Hadamard gates (green), controlled versions of $U^{2^0}, \ldots, U^{2^{k-1}}$ (red) and an inverse quantum Fourier transform (blue) to map $|\lambda\rangle|0\rangle^{\otimes k}$ to $|\lambda\rangle|\varphi_\lambda\rangle$ for any eigenstate $|\lambda\rangle$ of U.

First, we introduce some k ancilla qubits, where the value of k will be chosen later. The phase estimation algorithm uses k Hadamard gates, $\mathrm{c}U^{2^0}, \mathrm{c}U^{2^1}, \ldots, \mathrm{c}U^{2^{k-1}}$, as well as a standard operation known as the inverse "quantum Fourier transform" which uses $O(k^2)$ elementary gates (we refer to Ref. [10] for details of this operation) to perform the map

$$|\lambda\rangle|0\rangle^{\otimes k} \mapsto |\lambda\rangle|\varphi_\lambda\rangle\,, \tag{4.5}$$

where $|\varphi_\lambda\rangle$ is a k-qubit state which can be used to estimate the value of λ (see Fig. 4.1). Indeed, it can be shown that $|\varphi_\lambda\rangle$ has large overlap with the computational basis states that encode the first n binary digits of λ if $k > n$ is chosen sufficiently large.

Theorem 4.3 ([10])**.** *Let U be a unitary acting on a finite-dimensional state space and $|\lambda\rangle$ be an eigenstate of U with $U|\lambda\rangle = e^{2\pi i\lambda}|\lambda\rangle$ with $\lambda \in [0,1)$. Let n be a positive integer, $\epsilon > 0$, and $k = n + \lceil \log\bigl(2 + \frac{1}{2\epsilon}\bigr)\rceil$. Suppose that applying phase estimation of U with k ancilla qubits to $|\lambda\rangle|0\rangle^{\otimes k}$ results in the state $|\lambda\rangle|\varphi_\lambda\rangle$. Then, a measurement of the first n qubits of $|\varphi_\lambda\rangle$ in the computational basis yields the first n binary digits of λ with probability at least $1-\epsilon$.*

Moreover, it is easy to see that by expanding any state in an orthonormal eigenbasis of U, the algorithm can also be extended to any state.

Corollary 4.4 ([10])**.** *Let $U = \sum_i e^{2\pi i\lambda_j}|\lambda_j\rangle\langle\lambda_j|$ be a unitary acting on a finite-dimensional state space $\mathcal{H}$ with an orthonormal eigenbasis $\{|\lambda_j\rangle\}$ and eigenvalues $\{e^{2\pi i\lambda_j}\}$, where $\lambda_j \in [0,1)$. Let n be a positive integer, $\epsilon > 0$, $k = n + \lceil \log\bigl(2 + \frac{1}{2\epsilon}\bigr)\rceil$, and $|\phi\rangle = \sum_j \phi_j |\lambda_j\rangle$, $\sum_j |\phi_j|^2 = 1$, be an arbitrary state in $\mathcal{H}$. Then, applying phase estimation of U with k ancilla qubits to $|\phi\rangle|0\rangle^{\otimes k}$ followed by a measurement of the first n ancilla qubits in the computational basis yields the first n binary digits of λ_j with probability at least $|\phi_j|^2(1-\epsilon)$.*

In the context of Corollary 4.4, phase estimation can be intuitively seen as approximately simulating a projective measurement of the observable[2] $\sum_j \lambda_j |\lambda_j\rangle\langle\lambda_j|$ on the state $|\phi\rangle$. However, the scaling of the number k of ancillas required turns out to be unfavourable if good approximations of not just the eigenvalues but also the residual states are desired. We discuss this aspect in more detail in Section 5.3.

[2] In quantum mechanics, an *observable* is a Hermitian operator acting on the state space. If the observable has spectral decomposition $\sum_E EP_E$, where P_E is the projector onto the eigenspace with eigenvalue E, then a *projective measurement* of that observable on a system in a state $|\psi\rangle$ yields the outcome E with probability $\|P_E|\psi\rangle\|_2^2$ and leaves the system in the state $P_E|\psi\rangle/\|P_E|\psi\rangle\|_2$.

5 Quantum algorithms for state preparation

In this chapter, we present several quantum algorithms for preparing physically relevant states on quantum computers. The central problem we consider here is the preparation of ground states of a given Hamiltonian. Although we will also look at preparing other states, those problems can be reformulated as ground state problems.

The ability to efficiently prepare certain ground states on a quantum computer would have many important applications. Indeed, state preparation tasks are often important in simulation problems. For example, the simulation of quenches in quantum many-body systems using Hamiltonian simulation requires the preparation of the initial ground state of this process. Moreover, many optimisation problems can naturally be formulated as ground state problems.

Yet, ground state preparation is in general a hard problem. Indeed, many variations of the problem of determining the ground energy of a given Hamiltonian – including under various locality constraints – have been shown to be complete for the complexity class QMA [43]. The latter can be thought of as the analogous complexity class to NP for quantum computers, and as such it is generally conjectured that quantum computers cannot solve these problems efficiently. Nevertheless, given the profound applications of ground state preparation in several fields of science, quantum algorithms for this task which at least perform better than naive brute-force algorithms are highly desired. This could include e.g. heuristic algorithms which, despite having exponentially bad worst-case runtime bounds, may work significantly better in practice when applied to physically relevant systems.

Most existing quantum algorithms for ground state preparation are based on one of two methods. First, one could take a trial state $|\phi\rangle$ (e.g. a random product state) and attempt to project it into the ground state, e.g. using phase estimation. The probability of success of such methods generally depend inverse polynomially on $|\phi_0|$, where ϕ_0 is the inner product of $|\phi\rangle$ with the ground state. We discuss this class of *ground space projection algorithms* in Section 5.3 below.

The second class of algorithms are based on variants of the adiabatic theorem, which we will review below. Rigorous runtime bounds on adiabatic algorithms however generally depend inverse polynomially on the minimum spectral gap along a path of Hamiltonians, which in general is exponentially small, and moreover difficult to calculate or bound in practice. Thus, adiabatic algorithms are usually treated as *heuristic* methods to obtain states with (hopefully) good overlap with the ground state. The resulting states could e.g. be subsequently used as a trial state in a ground space projection method.

This combined approach of first generating a potentially good trial state with a heuristic method followed by attempting to project it onto its ground state component is expected to work significantly better than just using random trial states in ground space projection algorithms. Moreover, it is also the current paradigm envisioned e.g. for quantum chemistry applications [44].

This dissertation contributes to both parts of this approach. Core Article I presents an adiabatic algorithm for preparing injective PEPS and certain kinds of Gibbs states. Core Article III presents a quantum algorithm that projects a given trial state onto its ground space component which performs significantly better than phase estimation.

5.1 Adiabatic algorithms

Adiabatic algorithms are a common strategy to prepare ground states of a given Hamiltonian by connecting the target Hamiltonian $H(1) = H$ along a path of Hamiltonians $H(s)$, $s \in [0,1]$, to a "trivial" Hamiltonian $H(0)$ whose ground state can be easily prepared (e.g., a product state). The adiabatic theorem guarantees that if the initial state is in the ground state of $H(0)$, and evolves under a time-dependent Schrödinger equation whose Hamiltonian is slowly changed from $H(0)$ to $H(1)$, then for sufficiently long runtimes, the resulting final state will be close to the ground state of H. This runtime can be bounded in terms of the spectral gap of H and the norms of its derivatives:

Theorem 5.1 ([45, 46])**.** *Let $H(s)$, $s \in [0,1]$, be a twice continuously differentiable path of Hamiltonians acting on a finite-dimensional Hilbert space. Suppose that for all $s \in [0,1]$, $H(s)$ has a non-degenerate ground state $|\lambda_0(s)\rangle$ and a spectral gap of $\Delta(s)$. For $\tau > 0$, let $|\psi_\tau(t)\rangle$ be the solution to the time-dependent Schrödinger equation*

$$i\frac{\mathrm{d}}{\mathrm{d}t}\,|\psi_\tau(t)\rangle = H(t/\tau)\,|\psi_\tau(t)\rangle\,, \quad |\psi_\tau(0)\rangle = |\lambda_0(0)\rangle\,. \tag{5.1}$$

Then,

$$\min_{\theta\in[0,2\pi]} \| |\psi_\tau(\tau)\rangle - e^{i\theta}\,|\lambda_0(1)\rangle\|_2 = O(\epsilon), \tag{5.2}$$

provided that

$$\tau = \Omega\left(\frac{1}{\epsilon}\left(\frac{\|\dot{H}(0)\|_{\mathrm{op}}}{\Delta(0)^2} + \frac{\|\dot{H}(1)\|_{\mathrm{op}}}{\Delta(1)^2} + \int_0^1 \mathrm{d}s\frac{\|\dot{H}\|_{\mathrm{op}}^2}{\Delta^3} + \int_0^1 \mathrm{d}s\frac{\|\ddot{H}\|_{\mathrm{op}}}{\Delta^2}\right)\right). \tag{5.3}$$

In many cases, the inverse polynomial dependence on $\min_s \Delta(s)$ will dominate this runtime bound. Moreover, the adiabatic theorem only provides a Schrödinger evolution – i.e., a time-dependent Hamiltonian and a runtime bound. To turn this into a quantum circuit, the Schrödinger evolution has to be simulated using Hamiltonian simulation algorithms. This will in general result in additional overheads for the number of gates compared to the adiabatic runtime τ. For example, if $H(s)$ is a local Hamiltonian acting on a lattice of N sites for all $s \in [0,1]$, then the currently best Hamiltonian simulation algorithms require $O(\tau N^2 + N\log(1/\epsilon))$ gates [41, 47, 48].

The dependence on ϵ in Theorem 5.1 can in fact be exponentially improved using a suitable reparameterisation $H(s) \mapsto H(f(s))$, where $f : [0,1] \to [0,1]$ is a sufficiently smooth function. More precisely, recall that f is in the *Gevrey class* $1+\alpha$ if f is smooth and there exist $c, K > 0$ such that for all positive integers k and $s \in [0,1]$, $|\mathrm{d}^k f(s)/\mathrm{d}s^k| < Kc^k(k!)^{1+\alpha}$. It is well known that $f(s) = \int_0^s f_\alpha(t)\mathrm{d}t / \int_0^1 f_\alpha(t)\mathrm{d}t$ with $f_\alpha(t) = \exp\bigl(-1/((1-t)t)^{1/\alpha}\bigr)$ is in the Gevery class $1+\alpha$ for $\alpha > 0$ [49]. Using such a reparameterisation, one obtains the following improvement:

Theorem 5.2 ([1, 50])**.** *Let $H(s)$, $s \in [0,1]$, be a smooth path of Hamiltonians acting on a finite-dimensional Hilbert space. Suppose that for all $s \in [0,1]$, $H(s)$ has a non-degenerate ground state $|\lambda_0(s)\rangle$ and a spectral gap of $\Delta(s)$. Suppose moreover that all derivatives of H vanish at 0 and 1, and that H satisfies the Gevrey condition, i.e. there exist $c, K, \alpha > 0$ such that for all positive integers k,*

$$\|\mathrm{d}^k H(s)/\mathrm{d}s^k\|_{\mathrm{op}} < Kc^k(k!)^{1+\alpha}. \tag{5.4}$$

For $\tau > 0$, let $|\psi_\tau(t)\rangle$ be the solution to the time-dependent Schrödinger equation

$$i\frac{\mathrm{d}}{\mathrm{d}t}|\psi_\tau(t)\rangle = H(t/\tau)|\psi_\tau(t)\rangle, \quad |\psi_\tau(0)\rangle = |\lambda_0(0)\rangle. \tag{5.5}$$

Then,

$$\min_{\theta\in[0,2\pi]} \| |\psi_\tau(\tau)\rangle - e^{i\theta}|\lambda_0(1)\rangle \|_2 = O(\epsilon), \tag{5.6}$$

provided that

$$\tau = \Omega\left(\frac{(cK)^2}{\Delta^3}\log^{1+\alpha}\left(\frac{cK}{\Delta\epsilon}\right)\right), \tag{5.7}$$

where $\Delta = \min_{s\in[0,1]}\Delta(s)$.

The polylogarithmic dependence of the adiabatic runtime with respect to the allowed error has been proven in [50], which is based on the adiabatic expansion of [51]. In the supplemental material of Core Article I, we rederive this result by largely following the proof in [50], but unlike [50], we also explicitly establish the dependence on all other parameters, such as the spectral gap and the derivatives of the Hamiltonian.

5.2 Preparing injective PEPS on a quantum computer

PEPS with polynomial bond dimension are generally conjectured to be able to capture ground states of local gapped Hamiltonians. Hence, the problem of preparing ground states of local Hamiltonians is closely liked to the problem of preparing PEPS on a quantum computer. While preparing any MPS with low bond dimension on a quantum computer can be done efficiently [52], PEPS preparation is in general a very hard problem: it has been shown to be PP-hard [53]. The complexity class PP is extremely large since it contains QMA and hence in particular also NP.

Nevertheless, preparing certain subclasses of PEPS could be done efficiently under suitable conditions. The first such quantum algorithm to efficiently prepare "well-conditioned" injective PEPS has been proposed in Ref. [54]. Here, well-conditioned means that the condition numbers of all PEPS projectors $Q^{(v)}$ of $|\psi_{\mathrm{PEPS}}[\{Q^{(v)}\}_{v\in V}]\rangle$ are upper bounded by a positive constant $\kappa = O(1)$. PEPS preparation problems are usually turned into ground state preparation problems by constructing a so-called *parent Hamiltonian* of which the injective PEPS is the unique ground state. One way to construct such a parent Hamiltonian is by inverting the PEPS projectors $Q^{(v)}$ in (3.3) followed by projecting into the orthogonal complement of the maximally entangled pair state. More precisely, it is easy to see that $|\psi_{\mathrm{PEPS}}[\{Q^{(v)}\}_{v\in V}]\rangle$ is the unique ground state of the Hamiltonian $H_{\mathrm{parent}}[\{Q^{(v)}\}_{v\in V}]$ defined as follows.

Definition 5.3. Let (V, E) be a bounded degree graph, D, d be positive integers and $Q^{(v)} : (\mathbb{C}^D)^{\otimes \deg v} \to \mathbb{C}^d$ be invertible linear maps. Then, the *parent Hamiltonian* $H_{\mathrm{parent}}[\{Q^{(v)}\}_{v\in V}]$ of $|\psi_{\mathrm{PEPS}}[\{Q^{(v)}\}_{v\in V}]\rangle$ is defined as

$$H_{\mathrm{parent}}[\{Q^{(v)}\}_{v\in V}] = \sum_{e\in E}\left(\prod_{v\in e} Q^{(v)^{-1}}\right)^\dagger P_e \left(\prod_{v\in e} Q^{(v)^{-1}}\right), \tag{5.8}$$

where P_e is the projector onto the orthogonal complement of $|\phi_D^+\rangle_e$.

Note indeed that (5.8) is a sum of nearest-neighbour terms. Alternative parent Hamiltonian constructions using projectors onto the orthogonal complement of "PEPS tensors" with arbitrary boundaries are also widely considered [55].

The most natural approach to prepare a PEPS is by "growing" it site by site. Accordingly, one considers an ordering $v_1, \ldots, v_N$ of the vertices of V, as well as the intermediate states, which are "partial" PEPS where only the first k of the N PEPS projectors are applied, i.e.

$$|\psi_k\rangle = \left|\psi_{\mathrm{PEPS}}[\{\tilde{Q}_k^{(v)}\}_{v\in V}]\right\rangle \tag{5.9}$$

for $k = 0, \ldots, N$, where

$$\tilde{Q}_k^{(v_i)} = \begin{cases} Q^{(v_i)} & i \leq k \\ \mathbb{1}^{(v_i)} & i > k, \end{cases} \tag{5.10}$$

and $\mathbb{1}^{(v)}$ is a natural embedding of $(\mathbb{C}^D)^{\otimes \deg v}$ into $\mathbb{C}^d$. Note that $|\psi_0\rangle$ is just a product state of $|E|$ maximally entangled pair states, which can be trivially prepared, while $|\psi_N\rangle$ is our target state. The central result of Ref. [54] can now be stated as follows:

Theorem 5.4 ([54])**.** *Let D, d be positive integers, (V, E) be a simple graph of N vertices and bounded degree, and $Q^{(v)} : (\mathbb{C}^D)^{\otimes \deg v} \to \mathbb{C}^d$ be invertible linear maps for $v \in V$. Let $\kappa > 0$ be an upper bound of the condition numbers of all $Q^{(v)}$, and let $v_1, \ldots, v_N$ be an ordering of the vertices in V. For each $k = 0, \ldots, N$, let Δ_k be the spectral gap of the parent Hamiltonian $H_{parent}[\{\tilde{Q}_k^{(v)}\}_{v\in V}]$ of the k^{th} partial PEPS $\left|\psi_{PEPS}[\{\tilde{Q}_k^{(v)}\}_{v\in V}]\right\rangle$, where $\tilde{Q}_k^{(v)}$ is given by Eq.* (5.10)*, and let $\Delta = \min_k \Delta_k$. Then, there exists a bounded-error quantum algorithm generating an ϵ-close state to $\left|\psi_{PEPS}[\{Q^{(v)}\}_{v\in V}]\right\rangle$ in runtime $\tilde{O}(N^4\kappa^2/\Delta)$, where $\tilde{O}$ denotes the complexity up to subpolynomial factors in $N, \Delta^{-1}, \epsilon^{-1}, \kappa$.*

In most physically relevant injective PEPS, κ can be assumed to be a constant, $\kappa = O(1)$. In particular, this is true for translationally invariant injective PEPS since we assume $D, d = O(1)$. Thus, in many cases, the main limiting factor to the runtime of [54] is the contribution coming from the minimum spectral gap Δ. This dependence comes from the fact that the algorithm uses phase estimation[1] to transition from $|\psi_k\rangle$ to $|\psi_{k+1}\rangle$, whose runtime scales inversely proportional to the gap.

A particularly interesting case however arises in the presence of a *uniform gap*, i.e. when $\Delta = \Omega(1)$. For suitable orderings of the vertices, this is essentially the condition that the parent Hamiltonian remains gapped for arbitrary system sizes. In Core Article I, we propose an adiabatic approach for preparing injective PEPS, and show that in the presence of such a uniform gap, an ϵ-close state to $|\psi_N\rangle$ can be prepared by a quantum circuit with $O(N \operatorname{polylog}(N/\epsilon))$ gates (instead of $\tilde{O}(N^4)$) and depth $O(\operatorname{polylog}(N/\epsilon))$. The algorithm moreover applies to more general states, including purifications of Gibbs states of Hamiltonians with commuting local terms as well as the semi-injective PEPS introduced in Contributed Article V, provided that analogous uniform gap conditions for similar parent Hamiltonian constructions are satisfied. In fact, the algorithm applies to any state that is obtained by acting with invertible operators on any product of few-body states of nearby particles, which clearly also includes semi-injective PEPS. To see that purifications of Gibbs states of local Hamiltonians $H = \sum_i h_i$

[1] Strictly speaking, the proof provided in [54] only yields a runtime of $\tilde{O}(N^6\kappa^2/(\Delta\epsilon))$ rather than the one given in Theorem 5.4, since the runtime analysis in [54] does not take the error in the residual state resulting from phase estimation into account. The latter would incur an additional factor of $\tilde{O}(N^2/\epsilon)$ in the runtime for a final error of ϵ. The runtime of $\tilde{O}(N^4\kappa^2/\Delta)$ could however be obtained by replacing phase estimation with one of the other ground space projection algorithms discussed in Section 5.3.

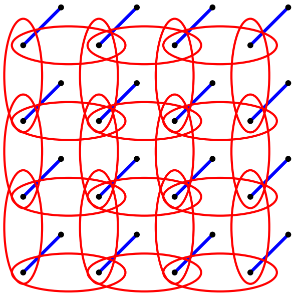


Figure 5.1: Purification of a Gibbs state for nearest-neighbour interactions. Each physical site is placed into a maximally entangled pair state with an ancilla system of the same dimension (blue). By applying $e^{-\beta h_i/2}$ (red) to the corresponding system vertices for each h_i, the reduced state on the physical sites is proportional to $e^{-\beta H}$.

with $[h_i, h_{i'}] = 0$ also fall into this category, consider the graph which contains sites composed of two vertices, one of which we call the "system" vertex and the other the "ancilla" vertex, each of which are placed in a maximally entangled pair state with the other. Then, by applying $e^{-\beta h_i/2}$ to the system vertices of the support of each h_i (see Fig. 5.1), is easy to see that by taking the partial trace over all ancilla vertices, we obtain $\rho \propto e^{-\beta H}$. Since Hamiltonians with commuting local terms include classical Hamiltonians as a special case, the algorithm can also be used to sample from Gibbs distributions of classical Hamiltonians.

5.3 Ground space projection algorithms

Ground space projection algorithms take a trial state $|\phi\rangle$ and attempt to project it onto its ground state component. Their performance will generally depend on the overlap of $|\phi\rangle$ with the ground state, such that these algorithms are best used in combination with e.g. adiabatic or other heuristic algorithms to obtain a good trial state.

For the remainder of this section, let H be an $N \times N$ Hermitian matrix. For convenience, we assume that H has been rescaled such that its spectrum is contained in $[0, 1]$. We assume that we are given the ability to efficiently perform Hamiltonian simulation of H at a "base cost" of Λ (e.g., if the simulation algorithm works in the oracle model [39], Λ is the gate cost of the oracles). Let λ_0 be the lowest eigenvalue of H, and $|\lambda_0\rangle$ be the corresponding eigenstate. We assume for simplicity that λ_0 is non-degenerate (although most results presented in this section generalise to degenerate ground spaces). Suppose that Δ is a known lower bound on the spectral gap of H.

We assume that the trial state $|\phi\rangle$ can be prepared with a circuit $\mathcal{C}_\phi$ using Φ elementary gates. Let $\phi_0 = \langle\lambda_0|\phi\rangle$ be its (generally unknown) overlap with the ground state, and χ be a known lower bound on $|\phi_0|$. We assume that $\chi = e^{-O(\log N)}$. This is an extremely weak assumption, indeed, this is satisfied even for random states with high probability.

The aim here is to obtain a state ϵ-close to $|\lambda_0\rangle$ by (approximately) projecting $|\phi\rangle$ onto its ground state component. If the value of λ_0 is unknown, we also wish to obtain an estimate for its value. A comparison of the runtimes as well as the qubit requirements of various ground space projection and ground energy estimation algorithms is given in Table 5.1. The stated space complexities do not account for qubits needed to perform Hamiltonian simulation. However, depending on the input model of H, there are Hamiltonian simulation algorithms (such as the ones in Refs. [39–41]) that do not change the asymptotic expressions in Table 5.1 (c.f. Theorem 4.2; we also refer to Table 1 of Ref. [41] for an overview).

Algorithm	Gates	Qubits	Required precision
Phase estimation	$\tilde{O}\left(\frac{\Lambda}{\lvert\phi_0\rvert^2\Delta\epsilon} + \frac{\Phi}{\lvert\phi_0\rvert}\right)$	$O\left(\log N + \log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$	$O\left(\lvert\phi_0\rvert\epsilon\Delta\right)$
Filtering	$\tilde{O}\left(\frac{\Lambda}{\lvert\phi_0\rvert\Delta} + \frac{\Phi}{\lvert\phi_0\rvert}\right)$	$O\left(\log N + \log\frac{1}{\epsilon} + \frac{\log\frac{1}{\chi\epsilon}}{\log\log\frac{1}{\chi\epsilon}} \times \log\frac{1}{\Delta}\right)$	$\tilde{O}(\Delta)$
Core Article III, Thm. 1	$\tilde{O}\left(\frac{\Lambda}{\lvert\phi_0\rvert\Delta} + \frac{\Phi}{\lvert\phi_0\rvert}\right)$	$O\left(\log N + \log\log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$	$\tilde{O}(\Delta)$

(a) Ground state preparation algorithms for the case when the ground energy is known beforehand to the required precision.

Algorithm	Gates	Qubits
Phase estimation	$\tilde{O}\left(\frac{\Lambda}{\chi^4\Delta\epsilon} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$
Filtering	$\tilde{O}\left(\frac{\Lambda}{\chi\Delta^{3/2}} + \frac{\Phi}{\chi\sqrt{\Delta}}\right)$	$O\left(\log N + \log\frac{1}{\epsilon} + \frac{\log\frac{1}{\chi\epsilon}}{\log\log\frac{1}{\chi\epsilon}} \times \log\frac{1}{\Delta}\right)$
Core Article III, Thm. 2	$\tilde{O}\left(\frac{\Lambda}{\chi\Delta^{3/2}} + \frac{\Phi}{\chi\sqrt{\Delta}}\right)$	$O\left(\log N + \log\log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$
Core Article III, Thm. 3	$\tilde{O}\left(\frac{\Lambda}{\chi^3\Delta} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$

(b) Ground state preparation algorithms for the case when the ground energy is not known beforehand. The last algorithm has been adjusted to yield the optimal scaling in Δ.

Algorithm	Gates	Qubits
Phase estimation	$\tilde{O}\left(\frac{\Lambda}{\chi^3\xi} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \log\frac{1}{\xi}\right)$
Filtering	$\tilde{O}\left(\frac{\Lambda}{\chi\xi^{3/2}} + \frac{\Phi}{\chi\sqrt{\xi}}\right)$	$O\left(\log N + \frac{\log\frac{1}{\chi}}{\log\log\frac{1}{\chi}} \times \log\frac{1}{\xi}\right)$
Core Article III, Thm. 4	$\tilde{O}\left(\frac{\Lambda}{\chi\xi^{3/2}} + \frac{\Phi}{\chi\sqrt{\xi}}\right)$	$O\left(\log N + \log\frac{1}{\xi}\right)$
Core Article III, Thm. 4	$\tilde{O}\left(\frac{\Lambda}{\chi^3\xi} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \log\frac{1}{\xi}\right)$

(c) Algorithms for estimating the ground energy to an additive precision of $\xi \ll \Delta$. The last algorithm has been adjusted to yield the optimal scaling in ξ.

Table 5.1: Comparisons of different ground space projection and ground energy estimation algorithms. Here, $\tilde{O}$ denotes the complexity up to polylogarithmic factors in N, Δ^{-1}, ϵ^{-1}, $|\phi_0|^{-1}$ and χ^{-1}. The stated runtime complexities already take suitable amplitude amplification techniques into account.

The most straightforward approach to implement a ground space projection in this setting is to use the phase estimation algorithm to approximately perform a measurement of the energy. Indeed, the relation of phase estimation to ground space projection is evident if one takes $U = e^{iHt}$ for some suitable $t > 0$ in Corollary 4.4. One would expect that for a suitable choice of the number k of ancilla qubits in phase estimation, if the measurement of the latter yields a good approximation of λ_0, then the state should be left in a state close to $|\lambda_0\rangle$. One would moreover naively expect the probability of success for this approach to be $\approx |\phi_0|^2$, meaning that $O(1/|\phi_0|^2)$ repetitions would naively be necessary, and that perhaps an improvement to $O(1/|\phi_0|)$ using suitable amplitude amplification techniques may be possible.

In Appendix A and B of Core Article III, we however show that the performance of phase estimation for ground state problems is actually significantly worse than one would naively expect (see Table 5.1). This is especially true if the value of the ground energy is not known beforehand. Roughly speaking, suppressing measurement outcomes significantly lower than the ground energy as well as ensuring a small distance of the residual state to the ground state lead to a required value of k which scales unfavourably with $|\phi_0|$, thus leading to the poor runtime. Moreover, an inverse polynomial dependence on ϵ is common to almost all algorithms based on phase estimation.

The one exception to this is a "filtering" method proposed by Poulin and Wocjan [56], which was originally designed to obtain a state with low expected energy faster than with phase estimation. With a little extra work, it can be shown that this method can also be used for ground state projection as well as ground energy estimation with a significantly faster runtime than the original phase estimation algorithm. This however comes at the cost of requiring a much larger number of ancillas.

This filtering method works as follows. Suppose that $\mathcal{A}_k$ is the phase estimation circuit with k qubits for the unitary U. Then, with the notation of Corollary 4.4, $\mathcal{A}_k\,|\lambda_j\rangle|0\rangle^{\otimes k} = |\lambda_j\rangle|\varphi_j\rangle$ for some k-qubit states $|\varphi_j\rangle$, and thus, for any k-qubit state $|\mu\rangle$, $\mathcal{A}_k^\dagger$ maps $|\phi\rangle|\mu\rangle$ to

$$\sum_j \phi_j \langle\varphi_j|\mu\rangle|\lambda_j\rangle|0\rangle^{\otimes k} + |R\rangle\,, \tag{5.11}$$

where $|R\rangle$ has no overlap with $|0\rangle^{\otimes k}$ on the k ancilla qubits. Hence, starting with some η copies of the state $|\mu\rangle$ on ηk ancilla qubits, an application of η copies of $\mathcal{A}_k^\dagger$ maps $|\phi\rangle|\mu\rangle^{\otimes\eta}$ to

$$\sum_j \phi_j \langle\varphi_j|\mu\rangle^{\eta}|\lambda_j\rangle|0\rangle^{\otimes \eta k} + \left|R'\right\rangle, \tag{5.12}$$

where $|R'\rangle$ has no overlap with $|0\rangle^{\otimes\eta k}$ on the ηk ancilla qubits. The main observation of Ref. [56] is that if $|\mu\rangle$ is a computational basis state encoding the binary representation of some $\mu \in \{0,\ldots,2^k-1\}$, then if $|\mu/2^k-\lambda_J|$ is sufficiently small and η is suitably large, the coefficients $\langle\varphi_j|\mu\rangle^\eta$ act as a "filter function" which keeps only the $|\lambda_J\rangle$ component of $|\phi\rangle$ while exponentially suppressing all other components, thus ultimately leading to an exponential improvement in the runtime dependence on ϵ compared to phase estimation. However, since η needs to be chosen sufficiently large for this to be achieved, this improvement in the runtime comes at the cost of requiring many more ancilla qubits. We refer to Table 5.1 for the precise scalings and Ref. [56] as well as Appendix C of Core Article III for the exact analysis.

The central result of Core Article III is to provide a new approach of ground space projection algorithms that can also be used to determine an unknown ground energy. Like the above filtering method, our algorithms achieve significant runtime improvements over phase estimation, but without the caveat of adding a large number of ancillas.

6 Hybrid quantum-classical approaches for small quantum computers

In this chapter, we explore the potential of quantum computers when the number of available logical qubits is constrained. This restriction will likely apply to early quantum computers. Given the current effort to construct prototypes of small quantum computers [8], the tasks of finding applications for such devices becomes increasingly important. A particularly promising approach is to extend the usefulness of such devices by supplementing them with classical computation. Algorithms that combine quantum and classical computational resources are commonly referred to as *hybrid quantum-classical algorithms*.

Clearly, quantum computers with only few qubits can be readily used to solve a given problem using a correspondingly space-efficient quantum algorithm if the size of the instance is small enough, i.e. if the workspace required by the quantum algorithm for the given problem size is at most the number of available qubits. Many simulation or ground state problems, including the ones presented in Chapter 5, potentially fit this category. Indeed, solving these problems even for small system sizes would already be interesting, firstly because they are inaccessible classically, but also because small system sizes may already provide important insights into some of the large-scale behaviour of the simulated systems.

Hybrid quantum-classical algorithms for problem sizes that are sufficiently small in the above sense have been proposed in the context of “Quantum Approximate Optimisation Algorithms” (QAOA) [57, 58] or “Variational Quantum Eigensolvers” (VQE) [59, 60]. These are generally relevant in the regime when the quantum devices available are not only constrained in the number of qubits, but also in the number of gates they can perform. These algorithms generally attempt to prepare a state $|\psi(\boldsymbol{\theta})\rangle$ which encodes an approximate solution to the given problem, where $|\psi(\boldsymbol{\theta})\rangle$ depends on some parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_l)$. QAOA is mostly proposed for classical optimisation problems such as the Max-Cut problem, which asks for a subset of vertices of a given graph that maximises the number of edges between the subset and its complement. In this algorithm, a classical pre-processing scheme first optimises the expected value of the objective function over $\boldsymbol{\theta}$ in a reasonably-sized parameter space, and then prepares $|\psi(\boldsymbol{\theta})\rangle$ on the quantum computer, followed by a measurement to obtain the approximate solution to the optimisation problem. In VQE algorithms, the objective function is usually the expected energy $\langle\psi(\boldsymbol{\theta})| H |\psi(\boldsymbol{\theta})\rangle$ with respect to some given Hamiltonian H. These algorithms iteratively optimise $\boldsymbol{\theta}$, where in each iteration, a short $\boldsymbol{\theta}$-dependent quantum circuit is applied to a given initial state followed by measurements and using their outcomes in a classical optimisation scheme to update the value of $\boldsymbol{\theta}$. Both of these hybrid algorithms can only be applied if the instance size strictly fits the number of available qubits.

The complementary approach for employing small quantum computers would be to utilise them for solving larger problem instances, even when the latter do not immediately fit the number of available qubits when using established quantum algorithms. Typically, quantum algorithms require a number of qubits at least comparable to the size of the problem instance, meaning that direct application of those algorithms to small devices would restrict their potential to only attacking small problem sizes. Especially for more “classical” problems, such as certain optimisation or decision problems, ways to attack larger instances using small quantum computers combined with classical processing would be highly desirable.

There have been some "circuit level" approaches [61, 62] to this challenge, which we will discuss in Section 6.1 below. Broadly speaking, these approaches try to simulate a general quantum circuit acting on strictly more than M qubits using only M qubits supplemented with additional classical computation. One could apply these results to the circuits of established quantum algorithms which use more qubits than are available, and thus in principle enable the solving of larger problem instances than straightforwardly possible. These circuit simulation approaches do not utilise any "algorithmic" structure of the circuit however, and as such their performances are necessarily constrained by structural properties of the circuit itself.

In Contributed Articles IV and VI, we propose a new approach on the "algorithmic level" for this problem: in a given overarching classical algorithm, we identify the most computationally expensive subroutines and replace them with more efficient quantum algorithms that can be run on the number of available qubits. In Contributed Article VI, the potential of size-limited quantum computers for large problem sizes is explored in the context of Schöning's algorithm for 3SAT. Contributed Article IV turns these ideas into a more general framework for divide-and-conquer algorithms under certain technical conditions, and demonstrates the applicability of these results in the context of Eppstein's algorithm for the cubic Hamiltonian cycle problem.

6.1 Techniques on the circuit level

In Refs. [61, 62], circuit simulation schemes have been proposed. There, an arbitrary quantum circuit $\mathcal{C}$ on n qubits is given, and the goal is to simulate the action of $\mathcal{C}$ using only $M < n$ qubits and classical computation. Whilst a fully classical simulation of $\mathcal{C}$ is generally expected to be exponentially costly [63], the general strategy of these methods is to trade off some of the classical runtime by running parts of the circuit on the availabe M qubits. For simplicity, we first restrict the meaning of a "simulation" of $\mathcal{C}$ to estimating a binary outcome obtained from a measurement of $\mathcal{C}\,|0\rangle^{\otimes n}$.

Definition 6.1. Let $\mathcal{C}$ be a quantum circuit on n qubits and consider the following process: first $\mathcal{C}$ is applied to $|0\rangle^{\otimes n}$, then each qubit is measured in the computational basis, and finally the measurement outcomes are classically post-processed into a single output bit b taking at most $O(\mathrm{poly}(n))$ time. We say that a process *ϵ-simulates* $\mathcal{C}$ if with probability at least $2/3$ it estimates the probability of $b = 1$ to an additive error of at most ϵ.

The first method for such a simulation was proposed in Ref. [61], which describes a scheme for adding "virtual qubits" to an existing quantum computer. The performance of this method is however dependent on the sparseness of the circuit.

Definition 6.2. A quantum circuit composed of one- and two-qubit gates is called *d-sparse* if every qubit of the circuit participates in at most d two-qubit gates.

Although quantum circuits of low sparsity are only a small subclass of general quantum circuits, they already contain many non-trivial examples which are generally believed to be hard to simulate classically [64]. In particular, any quantum circuit of depth $\leq d$ is also d-sparse, and several interesting quantum algorithms requiring only low depth circuits are known [65]. The central result of Ref. [61] provides a simulation of d-sparse quantum circuits using slightly fewer qubits than the original circuits act on.

Theorem 6.3 ([61])**.** *Let M, k, d be positive integers with $M > kd$. Then, any d-sparse quantum circuit on $M+k$ qubits can be ϵ-simulated by $O(2^{O(kd)} \log(1/\epsilon)/\epsilon^2)$ repetitions of a $(d+3)$-sparse quantum circuit on M qubits plus classical computation of runtime $O(2^{O(kd)} \operatorname{poly}(M))$.*

Note in particular that the simulation has runtime $O(\operatorname{poly}(M))$ if $k, d = O(1)$, and that the runtime scales exponentially only in k rather than $M + k$, as in the case of a fully classical simulation. However, due to the condition of $M > kd$, the applicability of this result is restricted to the case where only few qubits are missing. Moreover, although sparse quantum circuits are non-trivial, the limitation to small d is unsatisfactory. Some of these restriction were later overcome in the results of Ref. [62].

Theorem 6.4 ([62])**.** *Let $\mathcal{C}$ be a quantum circuit on n qubits and t gates with the following property: $\mathcal{C}$ can be partitioned into clusters such that K qubits in total are exchanged between different clusters. Suppose that each cluster can be run on M qubits. Then, $\mathcal{C}$ can be ϵ-simulated using M qubits in a total runtime of $O(2^{4K}(n + t)/\epsilon^2)$.*

Ref. [62] also provides a few examples of algorithms where their result is applicable, such as the simulation of Hamiltonians with special interaction graphs as well as certain VQE algorithms. However, the general difficulty with such circuit simulation techniques is that often, given a higher-level description of a quantum algorithm, it is unclear to what extent the relevant properties, such as the decomposability properties required by Theorem 6.4, are fulfilled for the underlying quantum circuit.

6.2 Techniques on the algorithmic level

Approaches on the "algorithmic level", in contrast to the circuit level techniques discussed in the previous section, exploit the structure of the algorithm and potentially modify the latter to obtain hybrid quantum-classical algorithms. One way to utilise few qubits is to take a classical algorithm and replace certain subroutines with faster quantum algorithms that can be run on the number of available qubits. One major challenge of designing hybrid algorithms this way is to identify ways to utilise small quantum computers to actually lead to genuine speedups of the overarching classical algorithm. Indeed, while it is usually possible to speed up smaller structure-independent subroutines, e.g. in the preparation or post-processing phase, such improvements rarely change the asymptotic runtime complexity. To achieve significant (e.g. polynomial) speedups, one needs to attack the computational bottlenecks of the algorithm. Doing this with few qubits is generally non-trivial, as straightforward "bottom-up" methods often break a global structure exploited by the classical algorithm, leading to a suboptimal overarching classical algorithm whose runtime may dominate the overall runtime. Thus, while arbitrarily sized quantum computers can speed up many classical algorithms (e.g. using amplitude amplification techniques), the potential of size-limited quantum computers is significantly less clear given large inputs.

The first result of Contributed Article IV is a general framework to attack the computational bottleneck of classical algorithms with few qubits whilst preserving the structure exploited by the algorithm. This framework applies to classical divide-and-conquer algorithms (as well as algorithms which can be reformulated as such), which are recursive algorithms that call themselves on ever smaller problem instances, according to a suitable problem size metric. The basic idea of the approach is to replace the recursive call with a suitable quantum algorithm deep in the recursion tree once the problem size is small enough to fit the number of available qubits. The second result of Contributed Article IV is a general method to reversibly and time-efficiently generate very space-efficient encodings of large sets using few ancillas. The result specifically bridges the gap between the seemingly irreconcilable properties of reversibility on the one hand, and low runtime and memory overheads on the other (c.f. Section 2.4) such that a polynomial speedup in the general framework can be achieved.

6.2.1 Schöning's algorithm for 3SAT

In Boolean satisfiability (or SAT) problems, one is given a Boolean formula $F : \{0,1\}^n \to \{0,1\}$ over n binary variables. The problem is to decide if a *satisfying assignment* $\mathbf{x} \in \{0,1\}^n$, which satisfies $F(\mathbf{x}) = 1$, exists. In 3SAT, F is restricted to be a conjunction of L clauses, where each clause is a disjunction of at most three literals, i.e., $F(\mathbf{x}) = \bigwedge_{j=1}^{L}(l_1^j \vee l_2^j \vee l_3^j)$, where each literal l_i^j specifies one of the n binary variables or its negation.

3SAT is the canonical NP-complete problem [66], and thus only exponential-time algorithms for both classical and quantum computers are expected to exist. There are however many classical 3SAT algorithms which are significantly faster than brute-force search, which has a runtime of $O^*(2^n)$, where O^* denotes the complexity up to polynomial factors in n. The performance of these algorithms can usually be characterised by a constant $\gamma \in (0,1)$, meaning that they provably solve 3SAT in a runtime of $O^*(2^{\gamma n})$. One of the best and most famous ones is the simple algorithm of Schöning [67], described in Algorithm 6.1.

Algorithm 6.1 Schöning's randomised algorithm for 3SAT.

```
SCHOENING($F$):
  1. Pick an initial assignment $\mathbf{x} \in \{0,1\}^n$ uniformly at random
  2. While $F(\mathbf{x}) = 0$ and we have repeated this at most $3n$ times
      a) Pick an arbitrary clause that is unsatisfied under $\mathbf{x}$
      b) Pick one of the variables in that clause uniformly at random and flip its value in $\mathbf{x}$
  3. Return $\mathbf{x}$
```

Schöning proved in Ref. [67] that if F is satisfiable, SCHOENING(F) returns a satisfying assignment with probability at least $(3/4)^n$. Thus, by repeating this process $O((4/3)^n)$ times, a randomised algorithm is obtained which solves 3SAT with bounded probability of error.

Theorem 6.5 ([67]). *There exists a classical randomised bounded-error algorithm that solves n-variable 3SAT in a runtime of* $O^*(2^{\gamma_0 n})$*, where* $\gamma_0 = \log(4/3) \approx 0.4150$.

Schöning's algorithm has later been improved from $O^*((4/3)^n) \approx O^*(1.3333^n)$ to $O^*(1.3302^n)$ [68], $O^*(1.3297^n)$ [69], $O^*(1.3290^n)$ [70], and $O^*(1.3211^n)$ [71]. Moreover, most classical 3SAT algorithms with provable runtime bounds are either based on the ideas of Schöning, the alternative approach of Ref. [72], which is often termed the "PPSZ" approach, or a combination of those two techniques. Such combined algorithms lead to runtimes of $O^*(1.3237^n)$ [73] or $O^*(1.3222^n)$ [74], while the PPSZ algorithm, originally with a proven runtime of $O^*(1.3633^n)$ [72], has subsequently been improved to $O^*(1.3207^n)$ [75] and $O^*(1.3070^n)$ [76].

In Ref. [77] Ambainis provided a quadratic quantum speedup over Schöning's original algorithm. The quantum algorithm uses $O(n)$ qubits, and essentially quantum-enhances Schöning's algorithm using amplitude amplification.

Theorem 6.6 ([77]). *There exists a bounded-error quantum algorithm which solves n-variable 3SAT in a runtime of* $O^*(2^{\gamma_0 n/2}) \approx O^*(1.1547^n)$ *using* $O(n)$ *qubits, where* $\gamma_0 = \log(4/3) \approx 0.4150$.

It is not immediately clear how Theorem 6.6 can be utilised if fewer qubits than necessary are available. To demonstrate the problem, assume for concreteness that the quantum algorithm in Theorem 6.6 requires βn qubits for some constant $\beta > 0$. Given a quantum computer with only a small fraction of this, say βm qubits for some $m \ll n$, a naive approach would be to use it as an m-variable 3SAT solver. The straightforward approach to solve the n-variable instance would then be to go through all possible assignments of some fixed $n - m$ of the n variables. Each such partial assignment induces an m-variable 3SAT instance which can then be solved on the available quantum computer. The total runtime of this algorithm would however be $O^*(2^{n-m+\gamma_0 m/2}) = O^*(2^{((1-a)+a\gamma_0/2)n})$, where $a = m/n$. In particular, the hybrid algorithm becomes *slower* than the original classical algorithm if $a \lesssim 0.74$. Broadly speaking, this "threshold effect" is caused by the naive hybrid approach breaking the global structure exploited by Schöning's algorithm, which in turn leads to suboptimal classical routines dominating the overall runtime.

The main result of Contributed Article VI is a hybrid quantum-classical speedup of Schöning's algorithm that avoids this threshold effect. More precisely, we show that there exists a function $f : \mathbb{R}^+ \to \mathbb{R}^+$ such that given a quantum computer with $M = cn$ qubits, where $c > 0$ is an arbitrary constant, n-variable 3SAT can be solved in a runtime of $O^*(2^{(\gamma_0 - f(c))n})$. The function f is somewhat involved, but can be shown to scale as $f(c) = \Theta(c/\log(1/c))$ for small values of c. Critically, irrespective of the exact form of $f(c)$, this result constitutes a polynomial speedup over Schöning's algorithm for any $c > 0$. The speedup is achieved by considering the derandomised formulation of Schöning's algorithm as a divide-and-conquer algorithm provided in [78, 79] and using some of the hybrid techniques which are later generalised to a general framework in Contributed Article IV.

6.2.2 Eppstein's algorithm for the cubic Hamiltonian cycle problem

A similar result can be obtained for speeding up Eppstein's algorithm for solving the cubic Hamiltonian cycle problem. This problem asks whether a given cubic graph $G = (V, E)$ with n vertices has a *Hamiltonian cycle*, i.e. a cycle going through every vertex exactly once.

This special case of the general Hamiltonian cycle problem, where no restrictions on the maximum degree of the graphs are placed, is known to be NP-complete [80]. Whilst a trivial brute-force search over vertex-orderings would take $O^*(n!)$ runtime, there exists a straightforward path search algorithm of runtime $O^*(2^n)$. In Ref. [81], Eppstein proposed a divide-and-conquer algorithm that solves the problem in a runtime of $O^*(2^{n/3})$. The algorithm is described in Algorithm 6.2.

The main idea of the algorithm is to introduce a subset $F \subset E$ of "forced" edges, which are edges that the target Hamiltonian cycle is required to contain. Then, by selecting an edge (step 3 of Algorithm 6.2) and creating two sub-instances of either forcing or deleting that edge (steps 4 and 5), the cubic structure of the graph ensures that in either case, additional edges can be forced or deleted in further "trivial reductions" (step 1), thus reducing the problem's overall complexity and leading to the given runtime.

Theorem 6.7 ([81]). *Algorithm 6.2 solves the cubic Hamiltonian cycle problem for n-vertex cubic graphs in a runtime of $O^*(2^{n/3})$.*

Eppstein's algorithm has subsequently been improved from $O^*(2^{n/3}) \approx O^*(1.2599^n)$ to $O^*(1.2509^n)$ [82] and $O^*(1.2312^n)$ [83]. However, the currently fastest known classical algorithm for the cubic Hamiltonian cycle problem with runtime $O^*(1.2009^n)$ is based on a Monte-Carlo approach [84].

Algorithm 6.2 Eppstein's algorithm for the cubic Hamiltonian cycle problem.

```
EPPSTEIN($G, F$):
  1. Repeat the following steps until none of the conditions apply
     a. If $G$ contains a vertex with degree two with at least one unforced incident edge, add all its incident edges to $F$
     b. If $G$ contains a vertex with degree three with exactly two forced edges, remove the unforced edge
     c. If $G$ contains a cycle of four unforced edges such that two of its opposite vertices are each incident to a forced edge and at least one of the other vertices is incident to an unforced edge that is not part of the cycle, then add to $F$ all non-cycle edges that are incident to a vertex of the cycle
  2. Check if any of the following conditions apply
     a. If $G$ contains a vertex of degree 0 or 1, or if $F$ contains three edges meeting at a vertex, return *false*
     b. If $G\backslash F$ is a collection of disjoint 4-cycles and isolated vertices
        i. If $G$ is disconnected, return *false*
        ii. Otherwise, return *true*
     c. If $F$ contains a non-Hamiltonian cycle, return *false*
  3. Choose an edge $yz$ according to the following cases
     a. If $G\backslash F$ contains a 4-cycle, exactly two vertices of which are incident to an edge in $F$, let $y$ be one of the other two vertices of the cycle and let $yz$ be an edge of $G\backslash F$ that does not belong to the cycle
     b. If there is no such 4-cycle, but $F$ is nonempty, let $xy$ be any edge in $F$ and $yz$ be an adjacent edge in $G\backslash F$ such that $yz$ is not part of an isolated 4-cycle in $G\backslash F$
     c. Otherwise, let $yz$ be any edge in $G$ that is not part of an isolated 4-cycle in $G\backslash F$
  4. Call EPPSTEIN($G, F \cup \{yz\}$)
  5. Call EPPSTEIN($G\backslash\{yz\}, F$)
  6. Return the disjunction (logical OR) of steps 4 and 5
```

A quadratic quantum speedup of the improvement to Eppstein's algorithm by Xiao and Nagamochi [83] has been proposed in Ref. [85]. Rather than directly using amplitude amplification, which in this case turns out to only yield a sub-quadratic advantage, they use more recent quantum backtracking techniques [86] to obtain a full quadratic improvement.

Theorem 6.8 ([85]). *There exists a bounded-error quantum algorithm that solves the cubic Hamiltonian cycle problem for n-vertex cubic graphs in a runtime of $O^*(2^{3n/20}) \approx O^*(1.1096^n)$ using $O(\mathrm{poly}(n))$ qubits.*

As with Theorem 6.6, this algorithm requires arbitrarily-sized quantum computers and it is not immediately clear whether it can be applied if the number of qubits is constrained. In Contributed Article IV, we show that there exists a function $f : \mathbb{R}^+ \to \mathbb{R}^+$ such that given a quantum computer with $M = cn$ qubits, where $c > 0$ is an arbitrary constant, the cubic Hamiltonian cycle problem for n-vertex graphs can be solved in a runtime of $O^*(2^{(1/3-f(c))n})$. The function f takes a similar form to the one obtained for the case of speeding up Schöning's algorithm.

Bibliography

[1] Y. Ge, A. Molnár, and J. I. Cirac, "Rapid adiabatic preparation of injective projected entangled pair states and Gibbs states", Physical Review Letters **116**, 080503 (2016).

[2] Y. Ge and J. Eisert, "Area laws and efficient descriptions of quantum many-body states", New Journal of Physics **18**, 083026 (2016).

[3] Y. Ge, J. Tura, and J. I. Cirac, "Faster ground state preparation and high-precision ground energy estimation with fewer qubits", Journal of Mathematical Physics **60**, 022202 (2019).

[4] Y. Ge and V. Dunjko, "A hybrid algorithm framework for small quantum computers with application to finding Hamiltonian cycles", arXiv preprint (2019).

[5] A. Molnar, Y. Ge, N. Schuch, and J. I. Cirac, "A generalization of the injectivity condition for projected entangled pair states", Journal of Mathematical Physics **59**, 021902 (2018).

[6] V. Dunjko, Y. Ge, and J. I. Cirac, "Computational speedups using small quantum devices", Physical Review Letters **121**, 250501 (2018).

[7] G. Scarpa, A. Molnar, Y. Ge, J. J. Garcia-Ripoll, N. Schuch, D. Perez-Garcia, and S. Iblisdir, "Computational complexity of PEPS zero testing", arXiv preprint (2018).

[8] J. Preskill, "Quantum computing in the NISQ era and beyond", Quantum **2**, 79 (2018).

[9] R. Feynman, "Simulating physics with computers", International Journal of Theoretical Physics **21**, 467–488 (1982).

[10] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, 10th anniversary edition (Cambridge University Press, 2011).

[11] H. Buhrman, J. Tromp, and P. Vitányi, "Time and space bounds for reversible simulation", Journal of Physics A: Mathematical and General **34**, 6821–6830 (2001).

[12] S. R. White, "Density matrix formulation for quantum renormalization groups", Physical Review Letters **69**, 2863–2866 (1992).

[13] F. Verstraete, D. Porras, and J. I. Cirac, "Density matrix renormalization group and periodic boundary conditions: a quantum information perspective", Physical Review Letters **93**, 227205 (2004).

[14] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac, "Matrix product state representations", Quantum Information and Computation **7**, 401 (2007).

[15] X. Chen, Z.-C. Gu, and X.-G. Wen, "Classification of gapped symmetric phases in one-dimensional spin systems", Physical Review B **83**, 035107 (2011).

[16] N. Schuch, D. Pérez-García, and I. Cirac, "Classifying quantum phases using matrix product states and projected entangled pair states", Physical Review B **84**, 165139 (2011).

[17] F. Verstraete and J. I. Cirac, "Renormalization algorithms for quantum-many body systems in two and higher dimensions", arXiv preprint (2004).

[18] D. Pérez-García, M. Sanz, C. E. González-Guillén, M. M. Wolf, and J. I. Cirac, "Characterizing symmetries in a projected entangled pair state", New Journal of Physics **12**, 025010 (2010).

[19] J. Eisert, M. Cramer, and M. B. Plenio, "Area laws for the entanglement entropy", Reviews of Modern Physics **82**, 277 (2010).

[20] P. Hayden, D. W. Leung, and A. Winter, "Aspects of generic entanglement", Communications in Mathematical Physics **265**, 95–117 (2006).

[21] M. B. Hastings, "An area law for one-dimensional quantum systems", Journal of Statistical Mechanics: Theory and Experiment **2007**, P08024–P08024 (2007).

[22] I. Arad, A. Kitaev, Z. Landau, and U. Vazirani, "An area law and sub-exponential algorithm for 1D systems", arXiv preprint (2013).

[23] F. Verstraete and J. I. Cirac, "Matrix product states represent ground states faithfully", Physical Review B **73**, 094423 (2006).

[24] N. Schuch, M. M. Wolf, F. Verstraete, and J. I. Cirac, "Entropy scaling and simulability by matrix product states", Physical Review Letters **100**, 030504 (2008).

[25] Z. Landau, U. Vazirani, and T. Vidick, "A polynomial time algorithm for the ground state of one-dimensional gapped local Hamiltonians", Nature Physics **11**, 566–569 (2015).

[26] Y. Huang, "A polynomial-time algorithm for the ground state of one-dimensional gapped Hamiltonians", arXiv preprint (2015).

[27] I. Arad, Z. Landau, U. Vazirani, and T. Vidick, "Rigorous RG algorithms and area laws for low energy eigenstates in 1D", Communications in Mathematical Physics **356**, 65–105 (2017).

[28] A. Molnar, N. Schuch, F. Verstraete, and J. I. Cirac, "Approximating Gibbs states of local Hamiltonians efficiently with projected entangled pair states", Physical Review B **91**, 045138 (2015).

[29] N. de Beaudrap, M. Ohliger, T. J. Osborne, and J. Eisert, "Solving frustration-free spin systems", Physical Review Letters **105**, 060504 (2010).

[30] K. Van Acoleyen, M. Mariën, and F. Verstraete, "Entanglement rates and area laws", Physical Review Letters **111**, 170501 (2013).

[31] M. Mariën, K. M. R. Audenaert, K. Van Acoleyen, and F. Verstraete, "Entanglement rates and the stability of the area law for the entanglement entropy", Communications in Mathematical Physics **346**, 35–73 (2016).

[32] F. G. S. L. Brandão and M. Cramer, "Entanglement area law from specific heat capacity", Physical Review B **92**, 115134 (2015).

[33] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, "Quantum amplitude amplification and estimation", in *Quantum computation and information*, Vol. 305, Contemporary Mathematics (AMS, 2002), pp. 53–74.

[34] T. J. Yoder, G. H. Low, and I. L. Chuang, "Fixed-point quantum search with an optimal number of queries", Physical Review Letters **113**, 210501 (2014).

[35] S. Lloyd, "Universal quantum simulators", Science **273**, 1073–1078 (1996).

[36] H. F. Trotter, "On the product of semi-groups of operators", Proceedings of the American Mathematical Society **10**, 545–551 (1959).

[37] P. R. Chernoff, “Note on product formulas for operator semigroups”, Journal of Functional Analysis **2**, 238–242 (1968).

[38] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma, “Simulating Hamiltonian dynamics with a truncated Taylor series”, Physical Review Letters **114**, 090502 (2015).

[39] D. W. Berry, A. M. Childs, and R. Kothari, “Hamiltonian simulation with nearly optimal dependence on all parameters”, in Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science, FOCS’15 (2015), pp. 792–809.

[40] G. H. Low and I. L. Chuang, “Optimal Hamiltonian simulation by quantum signal processing”, Physical Review Letters **118**, 010501 (2017).

[41] G. H. Low and I. L. Chuang, “Hamiltonian simulation by qubitization”, Quantum **3**, 163 (2019).

[42] A. Y. Kitaev, “Quantum measurements and the abelian stabilizer problem”, arXiv preprint (1995).

[43] S. Gharibian, Y. Huang, Z. Landau, and S. W. Shin, “Quantum Hamiltonian complexity”, Foundations and Trends in Theoretical Computer Science **10**, 159–282 (2015).

[44] M.-H. Yung, J. D. Whitfield, S. Boixo, D. G. Tempel, and A. Aspuru-Guzik, “Introduction to quantum algorithms for physics and chemistry”, in *Quantum information and computation for chemistry* (John Wiley & Sons, Inc., 2014), pp. 67–106.

[45] S. Jansen, M.-B. Ruskai, and R. Seiler, “Bounds for the adiabatic approximation with applications to quantum computation”, Journal of Mathematical Physics **48**, 102111 (2007).

[46] A. M. Childs, *Lecture notes on quantum algorithms* (2017).

[47] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, “Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics”, arXiv preprint (2018).

[48] S. Chakraborty, A. Gilyén, and S. Jeffery, “The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation”, arXiv preprint (2018).

[49] J.-P. Ramis, “Dévissage Gevrey”, Astérisque **59–60**, 173–204 (1978).

[50] G. Nenciu, “Linear adiabatic theory. Exponential estimates”, Communications in Mathematical Physics **152**, 479–496 (1993).

[51] G. A. Hagedorn and A. Joye, “Elementary exponential error estimates for the adiabatic approximation”, Journal of Mathematical Analysis and Applications **267**, 235–246 (2002).

[52] F. Verstraete, M. M. Wolf, and J. Ignacio Cirac, “Quantum computation and quantum-state engineering driven by dissipation”, Nature Physics **5**, 633–636 (2009).

[53] N. Schuch, M. M. Wolf, F. Verstraete, and J. I. Cirac, “Computational complexity of projected entangled pair states”, Physical Review Letters **98**, 140506 (2007).

[54] M. Schwarz, K. Temme, and F. Verstraete, “Preparing projected entangled pair states on a quantum computer”, Physical Review Letters **108**, 110502 (2012).

[55] D. Perez-Garcia, F. Verstraete, J. I. Cirac, and M. M. Wolf, “PEPS as unique ground states of local Hamiltonians”, Quantum Information and Computation **8**, 650–663 (2008).

[56] D. Poulin and P. Wocjan, “Preparing ground states of quantum many-body systems on a quantum computer”, Physical Review Letters **102**, 130503 (2009).

[57] E. Farhi, J. Goldstone, and S. Gutmann, “A quantum approximate optimization algorithm”, arXiv preprint (2014).

[58] E. Farhi, J. Goldstone, and S. Gutmann, “A quantum approximate optimization algorithm applied to a bounded occurrence constraint problem”, arXiv preprint (2014).

[59] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, “The theory of variational hybrid quantum-classical algorithms”, New Journal of Physics **18**, 023023 (2016).

[60] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O’Brien, “A variational eigenvalue solver on a photonic quantum processor”, Nature Communications **5**, 4213 (2014).

[61] S. Bravyi, G. Smith, and J. A. Smolin, “Trading classical and quantum computational resources”, Physical Review X **6**, 021043 (2016).

[62] T. Peng, A. Harrow, M. Ozols, and X. Wu, “Simulating large quantum circuits on a small quantum computer”, arXiv preprint (2019).

[63] S. Aaronson and L. Chen, “Complexity-theoretic foundations of quantum supremacy experiments”, in Proceedings of the 32nd Computational Complexity Conference, CCC’17 (2017), 22:1–22:67.

[64] B. M. Terhal and D. P. DiVincenzo, “Adaptive quantum computation, constant depth quantum circuits and Arthur-Merlin games”, Quantum Information and Computation **4**, 134 (2004).

[65] R. Cleve and J. Watrous, “Fast parallel circuits for the quantum Fourier transform”, in Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science, FOCS’00 (2000), pp. 526–536.

[66] M. R. Garey and D. S. Johnson, *Computers and intractability: a guide to the theory of NP-completeness* (W. H. Freeman & Co., 1990).

[67] T. Schöning, “A probabilistic algorithm for k-SAT and constraint satisfaction problems”, in Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science, FOCS’99 (1999), pp. 410–414.

[68] T. Hofmeister, U. Schöning, R. Schuler, and O. Watanabe, “A probabilistic 3-SAT algorithm further improved”, in Proceedings of the 19th Annual Symposium on Theoretical Aspects of Computer Science, STACS’02 (2002), pp. 192–202.

[69] D. Rolf, “3-SAT $\in RTIME(1.32971^n)$”, Diploma thesis (Department of Computer Science, Humboldt University Berlin, 2003).

[70] S. Baumer and R. Schuler, “Improving a probabilistic 3-SAT algorithm by dynamic search and independent clause pairs”, in Proceedings of the 6th International Conference on Theory and Applications of Satisfiability Testing, SAT’03 (2003), pp. 150–161.

[71] K. Iwama, K. Seto, T. Takai, and S. Tamaki, “Improved randomized algorithms for 3-SAT”, in Proceedings of the 21st International Symposium on Algorithms and Computation, ISAAC’10 (2010), pp. 73–84.

[72] R. Paturi, P. Pudlák, M. E. Saks, and F. Zane, “An improved exponential-time algorithm for k-SAT”, Journal of the ACM **52**, 337–364 (2005).

[73] K. Iwama and S. Tamaki, “Improved upper bounds for 3-SAT”, in Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA’04 (2004), pp. 328–328.

[74] D. Rolf, “Improved bound for the PPSZ/Schöning-algorithm for 3-SAT”, Journal on Satisfiability, Boolean Modeling and Computation **1**, 111–122 (2006).

[75] T. Hertli, R. A. Moser, and D. Scheder, “Improving PPSZ for 3-SAT using critical variables”, in Proceedings of the 28th Annual Symposium on Theoretical Aspects of Computer Science, STACS’11 (2011), pp. 237–248.

[76] T. Hertli, “3-SAT faster and simpler – Unique-SAT bounds for PPSZ hold in general”, SIAM Journal on Computing **43**, 718–729 (2014).

[77] A. Ambainis, “Quantum search algorithms”, SIGACT News **35**, 22–35 (2004).

[78] E. Dantsin, A. Goerdt, E. A. Hirsch, R. Kannan, J. Kleinberg, C. Papadimitriou, P. Raghavan, and U. Schöning, “A deterministic $(2-2/(k+1))^n$ algorithm for k-SAT based on local search”, Theoretical Computer Science **289**, 69–83 (2002).

[79] R. A. Moser and D. Scheder, “A full derandomization of Schöning’s k-SAT algorithm”, in Proceedings of the 43rd Annual ACM Symposium on Theory of Computing, STOC’11 (2011), pp. 245–252.

[80] M. R. Garey, D. S. Johnson, and R. E. Tarjan, “The planar Hamiltonian circuit problem is NP-complete”, SIAM Journal on Computing **5**, 704–714 (1976).

[81] D. Eppstein, “The traveling salesman problem for cubic graphs”, Journal of Graph Algorithms and Applications **11**, 61–81 (2007).

[82] K. Iwama and T. Nakashima, “An improved exact algorithm for cubic graph TSP”, in Proceedings of the 13th Annual International Conference on Computing and Combinatorics, COCOON’07 (2007), pp. 108–117.

[83] M. Xiao and H. Nagamochi, “An exact algorithm for TSP in degree-3 graphs via circuit procedure and amortization on connectivity structure”, Algorithmica **74**, 713–741 (2016).

[84] M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. v. Rooij, and J. O. Wojtaszczyk, “Solving connectivity problems parameterized by treewidth in single exponential time”, in Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science, FOCS’11 (2011), pp. 150–159.

[85] D. J. Moylett, N. Linden, and A. Montanaro, “Quantum speedup of the traveling-salesman problem for bounded-degree graphs”, Physical Review A **95**, 032323 (2017).

[86] A. Montanaro, “Quantum-walk speedup of backtracking algorithms”, Theory of Computing **14**, 1–24 (2018).

A Core articles

A.1 Rapid adiabatic preparation of injective projected entangled pair states and Gibbs states

Rapid adiabatic preparation of injective projected entangled pair states and Gibbs states

Yimin Ge, András Molnár, and J. Ignacio Cirac

In this work, we propose a quantum algorithm to efficiently prepare a particular set of states. This set contains two classes relevant for lattice problems: injective PEPS and purifications of Gibbs states of locally commuting Hamiltonians. The latter contains all classical Hamiltonians, and thus our algorithm can be used to simulate classical problems at finite temperatures.

Our algorithm outperforms all other previously known algorithms for these two problems under the uniform gap condition: we show that the number of elementary quantum gates scales only as $O\left(N \operatorname{polylog}\left(N/\varepsilon\right)\right)$, where N is the system size, ε the allowed error in trace distance, and the degree of the polynomial depends on the geometry of the lattice. This is essentially optimal. We moreover show that the algorithm is parallelisable, and that a circuit depth of $O\left(\operatorname{polylog}(N/\varepsilon)\right)$ can be obtained.

The class of states we consider in this work can more generally be thought of as commuting, finite range, and invertible operators acting on a set of maximally entangled pair states which are distributed on a lattice. We first construct a parent Hamiltonian G, i.e., a local Hamiltonian acting on the same lattice that has the desired state as its unique ground state (for Gibbs states of Hamiltonians with non-commuting local terms, we also show that an approximate quasi-local parent Hamiltonian can be considered above some constant temperature that allows the preparation in polynomial time – this is explored in Section IV of the supplemental material).

This state can in principle be adiabatically prepared by connecting G to a trivial Hamiltonian by a continuous path that has a unique ground state along the entire path. While constructing such a path is simple, this naive approach results in an non-optimal runtime even if the minimum spectral gap is constantly lower bounded. Indeed, the adiabatic theorem only gives an *adiabatic* runtime of $O(N^2\varepsilon^{-1})$. Moreover, the *actual* runtime (which is measured by the number of elementary gates in a quantum circuit) is even worse and scales as $O(N^4\varepsilon^{-1} \operatorname{polylog}(N/\varepsilon))$, which results from the overhead of implementing the adiabatic evolution with Hamiltonian simulation. To obtain the almost optimal runtime, our algorithm uses three main ingredients.

First, we prove a variant of the adiabatic theorem with an almost exponentially better runtime dependence on ε (Theorem 1 in the supplemental material). The main idea of this variant is to use a smooth reparameterisation of the Hamiltonian path using Gevrey class functions.

Second, the locality of the Hamiltonian allows the change of only a few local terms at a time instead of the entire Hamiltonian. More precisely, we construct a sequence of $O(N)$ Hamiltonian paths such that along each path, only $O(1)$ local terms change. The improved adiabatic theorem ensures that the accumulated error along the sequence remains small.

Finally, we exploit the locality of the Hamiltonian further by using Lieb-Robinson bounds to show that for the locally changed Hamiltonian paths, local terms which are far away from the local change do not significantly contribute to the evolution (Theorem 8 of the supplemental material). This is essential to obtain a small number of elementary gates. Indeed, while the sequence of local changes is sufficient to obtain a small *adiabatic* runtime, using Hamiltonians acting on the whole lattice would lead to a large overhead from Hamiltonian simulation. This final ingredient shows that at each step, it is in fact sufficient to evolve with a Hamiltonian that acts only on $O(\operatorname{polylog} N)$ sites, which almost exponentially reduces the overhead from Hamiltonian simulation, rendering it almost negligible. Moreover, the small subsystems required at each step means that the evolution of disjoint subsystems can be parallelised, which finally results in only a polylogarithmic circuit depth.

Our analysis relies on the existence of a gap $\Delta = \Omega(1)$ along the entire path, we however also show that this assumption can be significantly relaxed to only a gap at the beginning of each path in the sequence of the Hamiltonians (Theorem 9 of the supplemental material). We call this the *uniform gap* condition. The Hamiltonians for which we require a gap coincide with the final target Hamiltonian at smaller system sizes. Thus, the uniform gap condition is a natural assumption in the context of local Hamiltonians as it informally only requires that the spectral gap is well-behaved with respect to varying system sizes.

Statement of individual contribution

This work is the result of frequent discussions between András Molnár, J. Ignacio Cirac, and myself. J. Ignacio Cirac initially proposed the idea of preparing Gibbs states and injective PEPS adiabatically. I recognised the obstacle to the runtime coming from Hamiltonian simulation, especially when comparing our algorithm with classical Gibbs sampling algorithms, and guided our results and methods towards the finally obtained runtime. With regular advice from J. Ignacio Cirac, András Molnár and I jointly worked out the details of the algorithm and proofs involved. I was in charge of writing all parts of this article with the exception of Sections I, II and IV of the supplemental material.

I, Yimin Ge, am the principal author of this article and was extensively involved in all parts of it.

Permission to include:

Yimin Ge, András Molnár, and J. Ignacio Cirac.
Rapid adiabatic preparation of injective projected entangled pair states and Gibbs states.
Physical Review Letters, 116, 080503 (2016).

December 2017

APS Copyright Policies and Frequently Asked Questions

(…)

As the author of an APS-published article, may I include my article or a portion of my article in my thesis or dissertation?

Yes, the author has the right to use the article or a portion of the article in a thesis or dissertation without requesting permission from APS, provided the bibliographic citation and the APS copyright credit line are given on the appropriate pages.

(…)

FAQ Version: December 12, 2017

Rapid Adiabatic Preparation of Injective Projected Entangled Pair States and Gibbs States

Yimin Ge, András Molnár, and J. Ignacio Cirac
Max-Planck-Institut für Quantenoptik, D-85748 Garching, Germany

(Received 14 August 2015; revised manuscript received 22 December 2015; published 25 February 2016)

We propose a quantum algorithm for many-body state preparation. It is especially suited for injective projected entangled pair states and thermal states of local commuting Hamiltonians on a lattice. We show that for a uniform gap and sufficiently smooth paths, an adiabatic runtime and circuit depth of $O(\text{polylog}N)$ can be achieved for $O(N)$ spins. This is an almost exponential improvement over previous bounds. The total number of elementary gates scales as $O(N\text{polylog}N)$. This is also faster than the best known upper bound of $O(N^2)$ on the mixing times of Monte Carlo Markov chain algorithms for sampling classical systems in thermal equilibrium.

DOI: 10.1103/PhysRevLett.116.080503

Quantum computers are expected to have a deep impact in the simulation of *quantum* many-body systems, as initially envisioned by Feynman [1]. In fact, quantum algorithms have potential applications in diverse branches of science, ranging from condensed matter physics, atom physics, high-energy physics, to quantum chemistry [2]. Lloyd [3] was the first to devise a quantum algorithm to simulate the dynamics generated by few-body interacting Hamiltonians. When combined with the adiabatic theorem [4,5], the resulting algorithms allow one to prepare ground states of local Hamiltonians, and thus to investigate certain quantum many-body systems at zero temperature. Quantum algorithms have also been introduced to prepare so-called projected entangled pair states (PEPS) [6–8], which are believed to approximate ground states of local gapped Hamiltonians. Furthermore, quantum algorithms have also been proposed to sample from Gibbs distributions [9–14], which describe physical systems in thermal equilibrium. The computational time of most of these algorithms is hard to compare with that of their classical counterparts, as it depends on specific (e.g., spectral) properties of the Hamiltonians which are not known beforehand. However, they do not suffer from the sign problem [15], which indicates that they could provide significant speedups.

Quantum computers may also offer advantages in the simulation of *classical* many-body systems. For instance, quantum annealing algorithms [16,17] have been devised to prepare the lowest energy spin configuration of a few-body interacting classical Hamiltonian, which has obvious applications in optimization problems. Quantum algorithms have also been proposed to sample from their Gibbs distributions at finite temperature [18–23]. Apart from applications in classical statistical mechanics, similar problems appear in other areas of intensive research, e.g., machine learning. Speedups as a function of spectral gaps have been analyzed in Refs. [12,21,22]; the scaling with large system sizes, which is of particular interest for applications in deep machine learning [24], is however not optimal.

In this Letter we propose and analyze a quantum algorithm to *efficiently* prepare a particular set of states. This set contains two classes relevant for lattice problems: (i) injective PEPS [25]; (ii) Gibbs states of locally commuting Hamiltonians. Class (ii) contains all classical Hamiltonians, and thus the quantum algorithm allows us to sample Gibbs distributions of classical problems at finite temperature.

Our algorithm outperforms all other currently known algorithms for these two problems in the case that the minimum gap Δ occurring in the adiabatic paths (to be defined below) is lower bounded by a constant. We show that the computational time for a quantum computer, given by the number of elementary gates in a quantum circuit, scales only as

$$T = O(N\text{polylog}(N/\epsilon)), \tag{1}$$

where N is the number of local Hamiltonian terms, ε the allowed error in trace distance and the degree of the polynomial depends on the geometry of the lattice. Note that an obvious lower bound on the computational time is $\Omega(N)$, as each of the spins has to be addressed at least once. Thus, Eq. (1) is almost optimal. Furthermore, the algorithm is parallelizable, so that the depth of the circuit becomes

$$D = O(\text{polylog}(N/\varepsilon)). \tag{2}$$

This parallelization may also become very natural and relevant in analog quantum simulation, as is the case for atoms in optical lattices [26].

One of the best classical algorithms to sample according to the Gibbs distribution of a general classical Hamiltonian is the well-known Metropolis algorithm [27]. The currently best upper bound to its computational time is $T = O(N^2/\Delta_{\mathrm{stoch}})$ [28], where Δ_{stoch} is the gap of the generator of the stochastic matrix. We will see that given any stochastic matrix, one can always construct a quantum adiabatic algorithm with the same gap $\Delta = \Delta_{\mathrm{stoch}}$, and thus we obtain a potential quantum speedup of almost a factor of N. Under parallelization, the circuit depth is almost exponentially shorter. Our algorithm to prepare injective PEPS also provides a better scaling than the one presented in Ref. [7].

The class of states we consider in this Letter can be thought of as commuting finite range operators acting on a set of maximally entangled pair states (Fig. 1). More precisely, consider a regular lattice in some dimension, and let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the associated (infinite) graph. We endow $\mathcal{G}$ with a distance d, the minimum number of edges separating two vertices in $\mathcal{V}$. We associate a d-dimensional Hilbert space, $\mathcal{H}_v$, to each of the vertices $v \in \mathcal{V}$. Consider the set $\mathbf{\Lambda}$ of interaction supports, i.e., $\mathbf{\Lambda}$ is a collection of sets of vertices whose relative distance is at most a constant R, the interaction length, and consider for each $\lambda \in \mathbf{\Lambda}$ an interaction Q_λ which is an operator supported on $\otimes_{v\in\lambda} \mathcal{H}_v$. We assume that they are strictly positive, $\mathbb{1} \geq Q_\lambda > q_0 \mathbb{1}$, and mutually commute, $[Q_\lambda, Q_{\lambda'}] = 0$. Consider also a set $\mathbf{\Upsilon}$ of mutually excluding pairs of neighboring vertices. Moreover, let Λ_N be a finite subset of $\mathbf{\Lambda}$ with $|\Lambda_N| = N$, and define

$$|\phi_N\rangle \propto \prod_{\lambda\in\Lambda_N} Q_\lambda \bigotimes_{\mu\in\mathbf{\Upsilon}_N} |\phi^+\rangle_\mu, \tag{3}$$

where $\mathbf{\Upsilon}_N = \{\mu \in \mathbf{\Upsilon} | \mu \cap (\bigcup_{\lambda\in\Lambda_N} \lambda) \neq \emptyset\}$ is the set of pairs with a vertex in Λ_N, and $|\phi^+\rangle = \sum_{i=1}^d |ii\rangle$ is an unnormalized maximally entangled state between the pairs of vertices in $\mathbf{\Upsilon}_N$. We will give a quantum algorithm to prepare the state, Eq. (3), and analyze the runtime as a function of N and other spectral properties. In the following, we drop the subindex N to ease the notation.

As mentioned above, Eq. (3) includes two relevant classes of states. The first is the class of injective PEPS. The graph is composed of nodes, each of them including a set of vertices [Fig. 2(a)]. In this case, Υ contains pairs of vertices in nearest neighbor nodes, whereas Λ contains each node. The operators Q_λ act on different nodes, and therefore trivially commute. The resulting state is just a PEPS, which is injective since each Q_λ is invertible. In fact, every injective PEPS can be expressed in this form up to a local unitary using a QR decomposition. The second class is the class of Gibbs states of commuting Hamiltonians [29]. To see this, consider the graph which contains sites composed of two vertices, one of them is called "system" and the other "ancilla." The set Υ contains all sites, whereas Λ contains interacting system vertices [Fig. 2(b)]. The relation with Gibbs states is evident if we write $Q_\lambda = e^{-\beta h_\lambda/2}$, where $\|h_\lambda\| < 1$, and take into account that they mutually commute. It is easy to see that if we trace the ancillas, we obtain

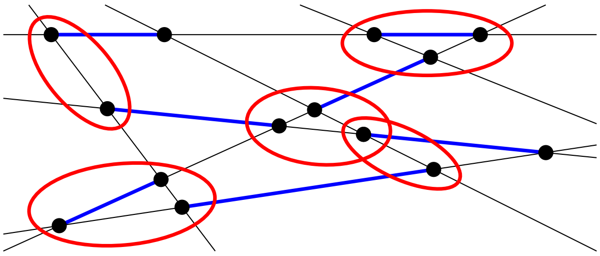


FIG. 1. The general class of states, Eq. (3). Finite range operators (red) acting on a collection of maximally entangled pair states (blue) distributed on a graph.

$$\rho \propto e^{-\beta H}, \tag{4}$$

where $H = \sum_{\lambda\in\Lambda} h_\lambda$.

The state Eq. (3) is the unique ground state of a frustration-free local Hamiltonian that can be written as

$$G = \sum_{\mu\in\Upsilon} G_\mu, \tag{5}$$

with

$$G_\mu = \left(\prod_{\lambda\in\Lambda_\mu} Q_\lambda^{-1}\right) P_\mu \left(\prod_{\lambda\in\Lambda_\mu} Q_\lambda^{-1}\right), \tag{6}$$

where $\Lambda_\mu = \{\lambda \in \Lambda | \lambda \cap \mu \neq \emptyset\}$ is the set of supports whose interactions act nontrivially on μ, and P_μ is the projector onto the subspace orthogonal to $|\phi^+\rangle_\mu$. Notice that since each G_μ is supported in a region of radius R around μ, G is indeed local.

The state Eq. (3) can be prepared using an adiabatic algorithm. For that, we define a path $Q_\lambda(s)$ with unique ground state $|\phi(s)\rangle$, where $s \in [0, 1]$, with $Q_\lambda(0) = \mathbb{1}$ and $Q_\lambda(1) = Q_\lambda$. We can choose $Q_\lambda(s) = (1 - s)\mathbb{1} + sQ_\lambda$.

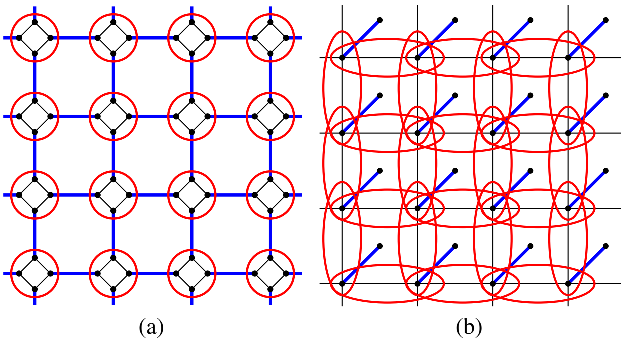


FIG. 2. (a) Projected entangled pair states. (b) Purification of a thermal state. For each system qudit, we introduce an ancilla to be placed in a maximally entangled pair with its system particle, then apply $e^{-\beta H/2}$ to the system.

In the case of the thermal state, we can also choose $Q_\lambda(s) = e^{-\beta s h_\lambda/2}$. Then, by starting with $|\phi(0)\rangle$ and changing the parameter $s: 0 \to 1$ sufficiently slowly, we will end up in the desired state $|\phi(1)\rangle$. The runtime for this preparation, as measured by the number of elementary quantum gates, is unpractical, however, as it scales as $T = O(N^4\Delta^{-3}\varepsilon^{-1}\mathrm{polylog}(N/\varepsilon))$, where ε is the tolerated error and Δ is the minimum spectral gap along the path. Indeed, the adiabatic theorem [30] gives an adiabatic runtime of $\tau = O(N^2\Delta^{-3}\varepsilon^{-1})$ so that Hamiltonian simulation [31] gives $T = O(\tau N^2 \mathrm{polylog}(N/\varepsilon))$.

To obtain a better scaling, we first use a variant of the adiabatic theorem with almost exponentially better runtime dependence on the error using a sufficiently smooth reparameterization of the Hamiltonian path. The quadratic scaling of the runtime with the derivative of the Hamiltonian, however, leads to an unpractical dependence on N since the Hamiltonian contains $O(N)$ terms that change with time. To avoid this, we change the Q's individually, leading to an adiabatic runtime of $\tau = O(N\log^{1+\alpha}(N/\varepsilon\Delta)\Delta^{-3})$. This, however, uses Hamiltonians acting on the whole system, despite only the change of a single Q, which would result in an additional factor of $O(N^2\mathrm{polylog}(N/\varepsilon))$ for the computational time measured by the number of elementary gates. We circumvent this problem by using Lieb-Robinson bounds [32] and the frustration freeness to show that under the assumption of a uniformly lower bounded spectral gap, it is at each step sufficient to evolve with a Hamiltonian acting only on $O(\mathrm{polylog}(N/\varepsilon))$ sites instead of the full lattice.

Thus, define a sequence of N Hamiltonian paths by enumerating the elements of Λ as $\lambda_1, \ldots, \lambda_N$, and define

$$G_n(s) = \sum_{\substack{\mu\in\Upsilon \\ d(\mu,\lambda_n)<\chi\log^{1+\alpha}(N/\varepsilon)}} G_{n,\mu}(s) \tag{7}$$

for $n = 1, \ldots, N$, where χ is a constant control parameter, and

$$G_{n,\mu}(s) = \left(\prod_{\substack{m \\ \lambda_m\in\Lambda_\mu}} A_{n,m}^{-1}(s)\right) P_\mu \left(\prod_{\substack{m \\ \lambda_m\in\Lambda_\mu}} A_{n,m}^{-1}(s)\right) \tag{8}$$

with

$$A_{n,m}(s) = \begin{cases} Q_{\lambda_m}(1) & m < n \\ Q_{\lambda_m}(s) & m = n \\ Q_{\lambda_m}(0) = \mathbb{1} & m > n. \end{cases} \tag{9}$$

Notice that G_n is supported on a region of radius $O(\log^{1+\alpha}(N/\varepsilon))$ and dG_n/ds is supported on a region of bounded size. By reparameterizing $G_n(s) \to G_n(f(s))$ with f, a function in the Gevrey class $1+\alpha$, we can assume G_n to be in the same Gevrey class [33].

Consider the sequence of Schrödinger equations

$$i\frac{d}{dt}|\psi_n\rangle = G_n\left(\frac{t}{\tau_n}\right)|\psi_n\rangle, \qquad |\psi_{n+1}(0)\rangle = |\psi_n(\tau_n)\rangle \tag{10}$$

for times $\tau_n = O(\log(N/\varepsilon)^{1+\alpha})$, starting in $|\psi_1(0)\rangle = |\phi(0)\rangle$, the trivial ground state of $G_1(0)$. The algorithm proceeds by running Hamiltonian simulation [31] on this sequence of adiabatic evolutions. Since at all times we only evolve with Hamiltonians acting on $O(\mathrm{polylog}(N/\varepsilon))$ sites, the number of gates only grows as Eq. (1). Moreover, the evolution of consecutive G_n's can be parallelized if their support is disjoint, i.e., if $G_n, \ldots, G_{n+l}$ have disjoint supports, the subsequence can be replaced by their sum without altering the evolution. Since $|\mathrm{supp}G_n| = O(\mathrm{polylog}(N/\varepsilon))$, it is clear that an ordering of the λ_n can be chosen such that subsequences of length $\Omega(N/\mathrm{polylog}(N/\varepsilon))$ of the G_ns can be parallelized at a time, resulting in a circuit of depth Eq. (2), an almost exponential improvement over previous bounds.

In the following, we show that for a uniformly lower bounded gap, the error of the above algorithm is bounded by ε. First, we use that under sufficient smoothness conditions on a Hamiltonian path $G(s)$, the final error can be almost exponentially small in the adiabatic runtime. Indeed, as proven in the Supplemental Material [36], if G is in the Gevrey class $1+\alpha$ and $d^kG/ds^k = 0$ at $s = 0, 1$ for all $k \geq 1$, then an adiabatic runtime of

$$\tau = O\left(\log^{1+\alpha}\left(\frac{K}{\varepsilon\Delta}\right)\frac{K^2}{\Delta^3}\right) \tag{11}$$

is sufficient for an error ε, where Δ is the minimum gap of $G(s)$ and $K = |\mathrm{supp}dG/ds|$ if G is local. The required smoothness conditions can always be achieved with a suitable reparametrization of the path $G(s) \to G(f(s))$.

This allows us to implement the global change of the Hamiltonian, Eq. (5), as a sequence of N local changes. Define the sequence of Hamiltonian paths,

$$\tilde{G}_n(s) = \sum_{\mu\in\Upsilon} G_{n,\mu}(s). \tag{12}$$

Notice that Eq. (12) is the same as Eq. (7), but contains all local terms $G_{n,\mu}$. The weak dependence on ε^{-1} in Eq. (11) ensures that the accumulated error under the sequential evolution with Eq. (12) remains small. Indeed, for a final error ε, it is sufficient that the evolution with each $\tilde{G}_n$ in this sequence only generates an error of at most ε/N. Equation (11) and $|\mathrm{supp}d\tilde{G}_n/ds| = O(1)$ imply that this can be achieved in a time $\tau_n = O(\log^{1+\alpha}(N/\varepsilon\Delta_n)\Delta_n^{-3})$, where Δ_n is the minimum spectral gap of $\tilde{G}_n$. A decomposition into a circuit then requires $T = O(N^3\mathrm{polylog}(N/\varepsilon\Delta)\Delta^{-3})$ elementary gates, where $\Delta = \min_n\Delta_n$. This is already an improvement by a factor N over the naive change of the entire Hamiltonian, assuming similar behaviour of Δ compared to the spectral gap of the original path $G(s)$.

Assuming that $\Delta = \Omega(1)$, we can further improve on this using Lieb-Robinson bounds to localize the effect of the adiabatic change. Indeed, we show in the Supplemental Material [36] that local terms in Eq. (12) which are supported at a distance $\Omega(\log^{1+\alpha}(N/\varepsilon))$ away from the support of $d\tilde{G}_n/ds$ do not significantly contribute to the unitary evolution generated by Eq. (12). This allows the replacement of Eq. (12) with Eq. (7) without significantly altering the evolution and thus the final state. Notice that G_n only acts on $O(\text{polylog}(N/\varepsilon))$ sites and $\tau_n = O(\text{polylog}(N/\varepsilon))$ for all n. Thus, its unitary evolution can be simulated with only $O(\text{polylog}(N/\varepsilon))$ gates. Hence, we finally obtain a number of gates in the circuit model that grows only as Eq. (1) for a constant error and lower bounded spectral gap. Using the described parallelization, we finally obtain a circuit depth, Eq. (2), as claimed.

In the analysis above, we have assumed a gap $\geq \Delta$ along all N paths. This assumption can in fact be relaxed to a gap at either $s = 0$ or $s = 1$ (see Supplemental Material [36]), using the positivity condition on Q_λ. We thus say that the system has a uniformly lower bounded gap Δ if for all finite subsets $\Lambda \subset \mathbf{\Lambda}$, the Hamiltonian, Eq. (5), has a spectral gap $\geq \Delta$. Under this assumption [47], the circuit depth, Eq. (2), can be guaranteed [48].

For the preparation of thermal states of classical Hamiltonians H, it is natural to compare these results with the performance of classical Monte Carlo Markov chain algorithms for Gibbs sampling such as the Metropolis algorithm or Glauber dynamics. Notice that due to the nature of their implementation, a fair comparison of performance should compare the mixing time of a discrete-time Markov chain to the number of elementary quantum gates, whereas the mixing time of a continuous-time Markov chain should be compared to the circuit depth. The best known upper bound on the discrete mixing time for Monte Carlo Markov chain algorithms for sampling from Gibbs distributions of classical Hamiltonians given just the promise of a spectral gap scales as $O(N^2)$. Although under certain additional assumptions such as translational invariance [49,50], weak mixing in two dimensions [51], or high temperature [52], the existence of a logarithmic Sobolev constant and hence the rapid (discrete) mixing time of $O(N \log N)$ can be proven, no such proof exists for the general case to the best of our knowledge. Our scheme thus outperforms classical Monte Carlo algorithms whenever rapid mixing cannot be shown even in the presence of a uniform gap.

Note that any classical Monte Carlo algorithm can be realized as an adiabatic algorithm, as has, e.g., been observed in Ref. [20]. Indeed, if S is the generator matrix of a continuous-time Monte Carlo algorithm that satisfies detailed balance with respect to the Gibbs distribution, $G = -e^{\beta H/2} S e^{-\beta H/2}$ is Hermitian. This Hamiltonian has the same spectrum as $-S$ and has the unique ground state $e^{-\beta H/2}|+\rangle^{\otimes N}$. For classical Hamiltonians H, this state has the same measurement statistics as ρ_β for observables that are products of σ_Z. By introducing an ancilla for every particle and applying the map $|i\rangle \mapsto |ii\rangle$, the purified version of the thermal state can also be recovered, and its parent Hamiltonian has the same spectrum as $-S$ within the symmetric subspace. Thus, any classical system with a uniform spectral gap for the generator matrix can be turned into a rapid adiabatic algorithm.

For quantum Hamiltonians, notice the restriction to commuting local terms. For noncommuting local terms, an approximate quasilocal parent Hamiltonian can be considered above some constant temperature that allows the preparation in polynomial time. We describe this procedure in the Supplemental Material [36].

For the preparation of injective PEPS, the given algorithm is similar to Ref. [7], which, however, requires a runtime of $O(N^4)$ in the uniformly gapped case, due to the use of phase estimation and the "Marriot-Watrous trick," which are computationally expensive for large systems. The better runtime of the present algorithm is largely due to replacing these subroutines by a local adiabatic change.

Throughout the analysis of this Letter, we focused on the case where a uniform constantly lower-bounded spectral gap is assumed. This assumption is only used to obtain a small number of elementary gates and circuit depth, whereas the adiabatic runtime of $\tau = O(N \log^{1+\alpha}(N/\varepsilon\Delta)\Delta^{-3})$ is independent of this assumption [53]. In contrast, the runtime of the algorithm to prepare PEPS given in Ref. [7] only grows as $T \sim \Delta^{-1}$ for small gaps, and for thermal states, algorithms based on quantum walks, phase estimation, and the quantum Zeno effect have been proposed with a runtime of $T \sim \Delta^{-1/2}$ [12,21,22], albeit with worse scaling in the system size. We believe that similar techniques can be applied to our scheme of local changes to obtain a good scaling of the runtime for both large system sizes and small spectral gaps. Moreover, it would be interesting to investigate if this scheme of local changes can also be applied to speed up *classical* Monte Carlo algorithms.

We have also shown that the algorithm can be parallelized, thus giving rise to a circuit depth that scales only polylogarithmically with N. This is particularly attractive for certain experimental realizations of analog quantum simulators, such as with atoms in optical lattices [54] or trapped ions [55], where this could be carried out in a very natural way.

We thank A. Lucia, D. Pérez-García, A. Sinclair, and D. Stilck França for helpful discussions. This work was supported by the EU Integrated Project SIQS.

[1] R. Feynman, Int. J. Theor. Phys. **21**, 467 (1982).

[2] Special issue on quantum simulation, edited by A. Trabesinger, Nature Physics Insight—Quantum Simulation **8**, 263 (2012).

[3] S. Lloyd, Science **273**, 1073 (1996).

[4] T. Kato, J. Phys. Soc. Jpn. **5**, 435 (1950).

[5] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, arXiv:quant-ph/0001106.

[6] F. Verstraete and J. I. Cirac, arXiv:cond-mat/0407066.

[7] M. Schwarz, K. Temme, and F. Verstraete, Phys. Rev. Lett. **108**, 110502 (2012).

[8] M. Schwarz, K. Temme, F. Verstraete, D. Perez-Garcia, and T. S. Cubitt, Phys. Rev. A **88**, 032321 (2013).

[9] B. M. Terhal and D. P. DiVincenzo, Phys. Rev. A **61**, 022301 (2000).

[10] E. Bilgin and S. Boixo, Phys. Rev. Lett. **105**, 170405 (2010).

[11] K. Temme, T. J. Osborne, K. G. Vollbrecht, D. Poulin, and F. Verstraete, Nature (London) **471**, 87 (2011).

[12] M.-H. Yung and A. Aspuru-Guzik, Proc. Natl. Acad. Sci. U.S.A. **109**, 754 (2012).

[13] A. Riera, C. Gogolin, and J. Eisert, Phys. Rev. Lett. **108**, 080402 (2012).

[14] M. J. Kastoryano and F. G. S. L. Brandão, arXiv:1409.3435.

[15] *Proceedings of the Ninth Taniguchi International Symposium, Monte Carlo Methods in Equilibrium and Nonequilibrium Systems*, edited by M. E. Suzuki (Springer, Susono, Japan, 1987).

[16] T. Kadowaki and H. Nishimori, Phys. Rev. E **58**, 5355 (1998).

[17] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, Science **292**, 472 (2001).

[18] P. C. Richter, Phys. Rev. A **76**, 042306 (2007).

[19] D. A. Lidar and O. Biham, Phys. Rev. E **56**, 3661 (1997).

[20] R. D. Somma, C. D. Batista, and G. Ortiz, Phys. Rev. Lett. **99**, 030603 (2007).

[21] P. Wocjan and A. Abeyesinghe, Phys. Rev. A **78**, 042336 (2008).

[22] R. D. Somma, S. Boixo, H. Barnum, and E. Knill, Phys. Rev. Lett. **101**, 130504 (2008).

[23] M.-H. Yung, D. Nagaj, J. D. Whitfield, and A. Aspuru-Guzik, Phys. Rev. A **82**, 060302 (2010).

[24] See, for example, Y. Bengio, Found. Trends Mach. Learn. **2**, 1 (2009).

[25] D. Perez-Garcia, F. Verstraete, J. I. Cirac, and M. M. Wolf, Quantum Inf. Comput. **8**, 0650 (2008).

[26] I. Bloch, J. Dalibard, and W. Zwerger, Rev. Mod. Phys. **80**, 885 (2008).

[27] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, J. Chem. Phys. **21**, 1087 (1953).

[28] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times* (American Mathematical Society, Providence, 2008).

[29] See also A. E. Feiguin and I. Klich, arXiv:1308.0756, for a similar parent Hamiltonian construction.

[30] S. Jansen, M.-B. Ruskai, and R. Seiler, J. Math. Phys. (N.Y.) **48**, 102111 (2007).

[31] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma, in *Proceedings of the 46th Annual ACM Symposium on Theory of Computing STOC '14* (ACM, New York, 2014), p. 283.

[32] E. H. Lieb and D. W. Robinson, Commun. Math. Phys. **28**, 251 (1972).

[33] Recall that a function $f:[0,1]\to\mathbb{R}^m$ is in the Gevrey class $1+\alpha$ [34] (with respect to the norm $\|\cdots\|$) if there exist constants $c, K>0$ such that $\|d^k f(s)/ds^k\| \leq K c^k (k!)^{1+\alpha}$ for all k. It is well known [35] that $f(s)=\int_0^s f_\alpha(t)dt/\int_0^1 f_\alpha(t)dt$, with $f_\alpha(t)=\exp\{-1/[(1-t)t]^{1/\alpha}\}$, is in the Gevrey class $1+\alpha$ for all $\alpha>0$.

[34] M. Gevrey, Annales scientifiques de l'École Normale Supérieure **35**, 129 (1918).

[35] J.-P. Ramis, Astérisque **59–60**, 173 (1978).

[36] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.116.080503 for the proof of the adiabatic theorem with almost exponential error decay, the locality of the local adiabatic change, relaxations on the assumption of the uniform gap, and Gibbs state preparation in the non-commuting case for high temperatures, which includes Refs. [37–46].

[37] G. Nenciu, Commun. Math. Phys. **152**, 479 (1993).

[38] G. A. Hagedorn and A. Joye, J. Math. Anal. Appl. **267**, 235 (2002).

[39] M. Hastings and T. Koma, Commun. Math. Phys. **265**, 781 (2006).

[40] B. Nachtergaele and R. Sims, Commun. Math. Phys. **265**, 119 (2006).

[41] S. Bravyi, M. B. Hastings, and F. Verstraete, Phys. Rev. Lett. **97**, 050401 (2006).

[42] R. Kotecký and D. Preiss, Commun. Math. Phys. **103**, 491 (1986).

[43] A. Molnar, N. Schuch, F. Verstraete, and J. I. Cirac, Phys. Rev. B **91**, 045138 (2015).

[44] D. A. Klarner, Can. J. Math. **19**, 851 (1967).

[45] D. A. Lidar, A. T. Rezakhani, and A. Hamma, J. Math. Phys. (N.Y.) **50**, 102106 (2009).

[46] A. T. Rezakhani, A. K. Pimachev, and D. A. Lidar, Phys. Rev. A **82**, 052305 (2010).

[47] In fact, this assumption can be relaxed to only hold for the N subsets $\Lambda_n=\{\lambda_1,\ldots,\lambda_n\}, n=1,\ldots,N$, for each problem size N, i.e., the sets being used in the algorithm, instead of all finite subsets.

[48] For thermal states, it can be shown perturbatively that there exists some constant value β_c such that the condition on the uniform spectral gap is always satisfied for $\beta<\beta_c$. For practical applications, it can however be hoped that this condition is satisfied for significantly higher β.

[49] F. Martinelli and E. Olivieri, Commun. Math. Phys. **161**, 447 (1994).

[50] F. Martinelli and E. Olivieri, Commun. Math. Phys. **161**, 487 (1994).

[51] F. Martinelli, E. Olivieri, and R. H. Schonmann, Commun. Math. Phys. **165**, 33 (1994).

[52] T. Hayes, in *47th Annual IEEE Symposium on Foundations of Computer Science, FOCS '06 (2006)*, (IEEE, Los Alamitos, 2006) pp. 39–46.

[53] See also S. Boixo and R. D. Somma, Phys. Rev. A **81**, 032308 (2010) for lower bounds on generic adiabatic runtimes.

[54] I. Bloch, J. Dalibard, and S. Nascimbene, Nat. Phys. **8**, 267 (2012).

[55] R. Blatt and C. F. Roos, Nat. Phys. **8**, 277 (2012).

Rapid adiabatic preparation of injective PEPS and Gibbs sates: Supplemental Material

Yimin Ge,[1] András Molnár,[1] and J. Ignacio Cirac[1]

[1] *Max-Planck-Institut für Quantenoptik, D-85748 Garching, Germany*

I. PROOF OF THE ADIABATIC THEOREM WITH ALMOST EXPONENTIAL ERROR DECAY

In this section, we prove a variant of the adiabatic theorem that only requires a runtime almost exponentially small in the allowed error. Our proof largely follows the proof given in [1], which is based on the method of adiabatic expansion [2]. The adiabatic expansion in [2] establishes an approximation of the time-dependent Schrödinger evolution in terms of the instantaneous ground state and its derivatives, but on its own does not necessarily imply an adiabatic theorem because it assumes a special initial state. Our proof, like [1], resolves this problem by exploiting the Gevrey-class condition which allows to satisfy these initial conditions, and uses this expansion to establish a bound on the required runtime. However, unlike [1], which only proves the almost exponential dependence of the runtime with respect to accuracy, our proof also explicitly establishes the dependence on all other parameters such as the spectral gap and the bound on the Hamiltonian derivatives [3].

Consider a smooth path of Hamiltonians, $G(s)$, $s \in [0,1]$, acting on a finite-dimensional Hilbert space $\mathcal{H}$. Let $\phi(s)$ [4] be the ground state of $G(s)$ and $\psi(\epsilon, s)$ the solution of the following Schrödinger equation:

$$i\epsilon\dot{\psi}(\epsilon, s) = G(s)\psi(\epsilon, s), \quad \psi(0) = \phi(0), \tag{13}$$

where $1/\epsilon = \tau$ is the runtime of the adiabatic algorithm, and $\dot{}$ denotes derivative with respect to s. We assume furthermore that the ground state energy of $G(s)$ is 0 (i.e., we fix the phase of ψ) and that it has a gap at least Δ throughout the whole path. By an appropriate choice of the phase of ϕ, we can without loss of generality assume that $\left\langle \dot{\phi}(s) \middle| \phi(s) \right\rangle = 0$. In the following, we will sometimes drop the explicit dependence on s to simplify the notation. Unless otherwise stated, $\|.\|$ will always denote the operator norm for operators and the Euclidean vector norm for vectors (it will always be clear from the context which one is used). In this section, we prove the following theorem.

Theorem 1. *Suppose that all derivatives of G vanish at 0 and at 1, and moreover that it satisfies the following Gevrey condition: there exist non-negative constants K, c and α such that for all $k \geq 1$,*

$$\|G^{(k)}\| \leq K c^k \frac{[k!]^{1+\alpha}}{(k+1)^2}. \tag{14}$$

Then,

$$\min_\theta \|\psi(\epsilon, 1) - e^{i\theta}\phi(1)\| \leq 8ce\frac{K}{\Delta}\left(\frac{4\pi^2}{3}\right)^3 \cdot \exp\left\{-\left(\frac{1}{4ec^2}\left(\frac{3}{4\pi^2}\right)^5 \tau\frac{\Delta^3}{K^2}\right)^{\frac{1}{1+\alpha}}\right\}. \tag{15}$$

Notice that we don't require the Gevrey condition (14) to hold for $k = 0$. Therefore, in the application of Theorem 1 in the main text, $K = O(1)$ since along the paths $\tilde{G}_n(s)$ (as defined in (12) in the main text), only $O(1)$ local terms change.

Proof of Theorem 1. Following the adiabatic expansion method from [1, 2], we search $\psi(\epsilon, s)$ in the form of an asymptotic series expansion by constructing vectors $\phi_j(s)$, $s \in [0,1]$, $j \geq 0$, such that for all $M > 0$,

$$\left\|\psi(\epsilon, s) - \sum_{j=0}^{M-1} \phi_j(s)\epsilon^j\right\| = O(\epsilon^M). \tag{16}$$

We first show an explicit expression for ϕ_j provided that such an expansion exists. Second,we prove that the expansion really exists if $G^{(k)}(0) = 0$ for all k by giving an explicit error bound. Third, to connect the expansion to the adiabatic theorem, we show that

$$\min_\theta \|\psi(\epsilon, 1) - e^{i\theta}\phi(1)\| \leq 2\|\psi(\epsilon, 1) - \sum_{j=0}^{M-1} \phi_j(s)\epsilon^j\|, \tag{17}$$

for some θ for all M if $G^{(k)}(1) = 0$ for all k. This already proves an error bound of $O(\epsilon^M)$ for any M. Finally, if G is Gevrey class, then the error bound can be expressed with the help of the parameters appearing in the Gevrey condition and using a suitable choice of M yields to the bound in Eq. (15).

Explicit form of ϕ_j. To satisfy the equation at $s = 0$, we require $\phi_0(0) = \phi(0)$ and $\phi_j(0) = 0$ for all $j > 0$. Furthermore, substituting back the ansatz to the Schrödinger equation Eq.(13), following [2], we arrive at the recursion

$$\phi_j = \varphi_j \phi + iG^{-1}\dot{\phi}_{j-1}, \quad \varphi_j = i\int_0^t \mathrm{d}t' \left\langle \dot{\phi}(t') \middle| G^{-1}(t') \middle| \dot{\phi}_{j-1}(t') \right\rangle, \tag{18}$$

for all $j > 0$, where $\varphi_j(s)$ is a complex number and G^{-1} is the pseudo-inverse of G, and initial values are

$$\phi_0(s) = \phi(s) \tag{19}$$

$$\varphi_0(s) = 1. \tag{20}$$

Note that $\varphi_j(0)$ has to be zero in order for $\phi_j(0)$ to be zero, but this is not a sufficient condition. We will investigate below when $\phi_j(0) = 0$ can be satisfied.

Existence of the expansion. To satisfy $\phi_j(0) = 0$ for $j > 0$, $\dot{\phi}_{j-1}(0)$ needs to be parallel to $\phi(0)$. This is satisfied if all derivatives of G are 0 at $s = 0$ (see Lemma 2 below). We show that if this condition is fulfilled, then the expansion exists.

Define the truncation of the asymptotic series expansion,

$$\psi_M = \sum_{j=0}^{M-1} \phi_j \epsilon^j. \tag{21}$$

Note that if $\|\psi - \psi_M\| = O(\epsilon^{M-1})$, then the expansion exists. Indeed, then $\|\psi - \psi_M\| = \|\psi - \psi_{M+1} + \epsilon^M \phi_M\| = O(\epsilon^M)$. By construction, ψ_M almost satisfies the Schrödinger equation: $i\epsilon\dot{\psi}_M = G\psi_M + i\epsilon^M\dot{\phi}_{M-1}$. Let $U(s)$ be the dynamics generated by G/ϵ. Then, $\|\psi_M(\epsilon, s) - \psi(\epsilon, s)\| = \|U(s)^\dagger \psi_M(\epsilon, s) - \phi(0)\|$ and

$$\left\|U(s)^\dagger \psi_M(\epsilon, s) - \phi(0)\right\| = \left\|\int_0^s \mathrm{d}s' \frac{\mathrm{d}}{\mathrm{d}s'}\left(U^\dagger \psi_M\right)\right\| = \left\|\epsilon^{M-1}\int_0^s \mathrm{d}s' U^\dagger \dot{\phi}_{M-1}\right\| \leq \epsilon^{M-1}\int_0^s \mathrm{d}s' \left\|\dot{\phi}_{M-1}\right\|, \tag{22}$$

where we used that if the first M derivatives of G are 0, then $\psi_M(\epsilon, 0) = \phi(0)$. This proves the existence of the expansion.

Connecting the expansion to the adiabatic theorem. Using the triangle inequality, we obtain

$$\min_\theta \|\psi(\epsilon, 1) - e^{i\theta}\phi(1)\| \leq \|\psi(\epsilon, 1) - \psi_M(\epsilon, 1)\| + \min_\theta \|\psi_M(\epsilon, 1) - e^{i\theta}\phi(1)\|. \tag{23}$$

In Lemma 2, we prove that if the first M derivatives of $G(s)$ vanish at $s = 1$, then $\phi_j(1)$ is parallel to $\phi(1)$ for all $j = 1, \ldots, M$. Therefore, $\psi_M(\epsilon, 1)$ is parallel to $\phi(1)$, so that $\min_\theta \|\psi_M(\epsilon, 1) - e^{i\theta}\phi(1)\| = |\|\psi_M(\epsilon, 1)\| - 1|$. But $1 = \|\psi(\epsilon, 1)\|$, so using the triangle inequality, we get $\min_\theta \|\psi_M(\epsilon, 1) - e^{i\theta}\phi(1)\| \leq \|\psi_M(\epsilon, 1) - \psi(\epsilon, 1)\|$. Therefore,

$$\min_\theta \|\psi(\epsilon, 1) - e^{i\theta}\phi(1)\| \leq 2\|\psi(\epsilon, 1) - \psi_M(\epsilon, 1)\|. \tag{24}$$

Choice of M. From Eq. (22) and (24),

$$\min_\theta \|\psi(\epsilon, 1) - e^{i\theta}\phi(1)\| \leq 2\|\psi(\epsilon, 1) - \psi_M(\epsilon, 1)\| \leq 2\epsilon^{M-1}\int_0^1 \|\dot{\phi}_{M-1}\|, \tag{25}$$

so we only need to bound $\|\dot{\phi}_{M-1}\|$. We do this by using that G is Gevrey class. From Lemma 7 below,

$$\|\dot{\phi}_{M-1}\| \leq 2\frac{4\pi^2}{3} \cdot 2c\frac{K}{\Delta}\left(\frac{4\pi^2}{3}\right)^2 \cdot \left[\frac{K^2}{\Delta^3}4c^2\left(\frac{4\pi^2}{3}\right)^5\right]^{M-1} \frac{[M!]^{1+\alpha}}{(M+1)^2}. \tag{26}$$

Therefore,

$$\|\psi(\epsilon, 1) - \phi(1)\| \leq \epsilon^{M-1} \cdot 8c\frac{K}{\Delta}\left(\frac{4\pi^2}{3}\right)^3 \cdot \left[\frac{K^2}{\Delta^3}4c^2\left(\frac{4\pi^2}{3}\right)^5\right]^{M-1} \frac{[M!]^{1+\alpha}}{(M+1)^2}. \tag{27}$$

Changing M to $M+1$ and using that $[(M+1)!]^{1+\alpha}/(M+2)^2 \leq M^{(1+\alpha)M}$, we obtain

$$\|\psi(\epsilon,1)-\phi(1)\| \leq 8c\frac{K}{\Delta}\left(\frac{4\pi^2}{3}\right)^3 \cdot \left[\frac{K^2}{\Delta^3}4c^2\left(\frac{4\pi^2}{3}\right)^5\frac{M^{1+\alpha}}{\tau}\right]^M . \tag{28}$$

This is true for any M, so setting

$$M = \left\lfloor \left(\frac{\tau\Delta^3}{eK^2 4c^2\left(\frac{4\pi^2}{3}\right)^5}\right)^{\frac{1}{1+\alpha}} \right\rfloor , \tag{29}$$

we obtain

$$\|\psi(\epsilon,1)-\phi(1)\| \leq 8ce\frac{K}{\Delta}\left(\frac{4\pi^2}{3}\right)^3 \cdot \exp\left\{-\left(\tau\frac{\Delta^3}{K^2}\frac{1}{4ec^2}\left(\frac{3}{4\pi^2}\right)^5\right)^{\frac{1}{1+\alpha}}\right\} . \tag{30}$$

This proves Theorem 1. □

We have repeatedly used the following lemma in the proof of Theorem 1.

Lemma 2. *If $G^{(k)}(s_0) = 0$ for some $s_0 \in [0,1]$ and for all $k = 1 \dots M$, then*

(i) $\phi^{(k)}(s_0)$ is parallel to $\phi(s_0)$ for all $k = 0, \dots, M$,

(ii) $\left[G^{(-1)}\right]^{(k)}(s_0) = 0$ for all $k = 1, \dots, M$,

(iii) $\phi_k^{(l)}(s_0)$ is parallel to $\phi(s_0)$ for all $k = 0, \dots, M$ and $l = 0, \dots, M-k$.

Proof. $G\phi = 0$, so $(G\phi)^{(k)} = \sum_{j=0}^{k}\binom{k}{j}G^{(j)}\phi^{(k-j)} = 0$ for all k. Applying this for $k \leq M$ and evaluating the result at s_0, the derivatives of G vanish, thus $G\phi^{(k)} = 0$ and therefore $\phi^{(k)}$ is parallel to ϕ at s_0, which proves (i).

To prove (ii), use the Cauchy formula

$$G^{-1} = \frac{1}{2\pi}\oint_\Gamma (z-G)^{-1}\frac{1}{z}\mathrm{d}z, \tag{31}$$

where $\Gamma = \{z \in \mathbb{C} \mid |z| = \Delta/2\}$ is a fixed curve around 0. Taking the kth derivative of Eq. (31) and evaluating it at s_0, we see that the derivatives of G^{-1} also disappear.

To prove (iii), we proceed by induction on k. By (i), the claim is true for $k = 0$. For $k > 0$, we have $\phi_k^{(l)} = (\varphi_k\phi)^{(l)} + i(G^{-1}\phi_{k-1}^{(1)})^{(l)}$. By (i), the first term is parallel to ϕ at s_0. The second term consists of derivatives of G^{-1} and derivatives of $\phi_k^{(1)}$. By (ii), the derivatives of G^{-1} vanish at s_0, so that the only remaining term is $iG^{-1}\phi_{k-1}^{(l+1)}$. But this term also vanishes at s_0 by the induction hypothesis. This proves (iii). □

In the remainder of this section, we derive the bound on the norm of $\dot{\phi}_{M-1}$ which was used in the proof of Theorem 1, following the analysis in [1]. First, we recall two technical lemmas from [1], which will be used repeatedly.

Lemma 3. *Let p, q be non-negative integers and $r = p+q$. Then,*

$$\sum_{l=0}^{k}\binom{k}{l}\frac{[(l+p)!(k-l+q)!]^{1+\alpha}}{(l+p+1)^2(k-l+q+1)^2} \leq \frac{[(k+r)!]^{1+\alpha}}{(k+r+1)^2}\frac{4\pi^2}{3}. \tag{32}$$

Proof. Notice that if $r = p+q$, then

$$\binom{k}{l}[(l+p)!(k+q-l)!]^{1+\alpha} = \frac{\binom{k}{l}[(k+r)!]^{1+\alpha}}{\binom{k+r}{l+p}^{1+\alpha}} \leq [(k+r)!]^{1+\alpha}. \tag{33}$$

To upper-bound the summation, divide the sum into two parts at $\lfloor(k-p+q)/2\rfloor$. If $l < \lfloor(k-p+q)/2\rfloor$, then $(k-l+q+1) > \lfloor(k+p+q)/2\rfloor + 1$. Otherwise, if $l \geq \lfloor(k-p+q)/2\rfloor$, then $(l+p+1) \geq \lfloor(k+p+q)/2\rfloor + 1$. Therefore,

$$\sum_{l=0}^{k}\frac{1}{(l+p+1)^2(k-l+q+1)^2} \leq 2\sum_{l=0}^{\infty}\frac{1}{(l+1)^2}\frac{1}{(\lfloor(k+p+q)/2\rfloor+1)^2}. \tag{34}$$

This can be upper-bounded by $4\pi^2/3$ as $k+p+q+1 \leq 2(\lfloor(k+p+q)/2\rfloor+1)$. This finishes the proof of Lemma 3. □

We now use Lemma 3 to prove that if A and B are Gevrey-class, then their product is also Gevrey-class.

Lemma 4. *Let $A(s), B(s)$ $(s \in [0,1])$ be smooth paths of either vectors in $\mathcal{H}$ or operators in $\mathcal{B}(\mathcal{H})$ satisfying*

$$\|A^{(k)}\| \leq Cd^k \frac{[(k+p)!]^{1+\alpha}}{(k+p+1)^2} \quad \text{and} \quad \|B^{(k)}\| \leq Ef^k \frac{[(k+q)!]^{1+\alpha}}{(k+q+1)^2} \tag{35}$$

for some non-negative constants C, d, E, f, non-negative integers p, q, and for all $k \geq 0$. Then,

$$\|(AB)^{(k)}\| \leq \frac{4\pi^2}{3} CEg^k \frac{[(k+r)!]^{1+\alpha}}{(k+r+1)^2} \tag{36}$$

for all $k \geq 0$, where $g = max(d, f)$ and $r = p + q$.

Proof. We have

$$\|(AB)^{(k)}\| \leq \sum_l \binom{k}{l} \|A^{(l)}\| \|B^{(k-l)}\|, \tag{37}$$

so inserting the bounds (35) and upper-bounding d and f by g, we obtain

$$\|(AB)^{(k)}\| \leq CEg^k \sum_l \binom{k}{l} \frac{[(l+p)!]^{1+\alpha}}{(l+p+1)^2} \frac{[(k-l+q)!]^{1+\alpha}}{(k-l+q+1)^2}. \tag{38}$$

Using Lemma 3 to upper-bound the r.h.s. of this expression proves Lemma 4. □

Next we give a bound on the derivatives of the pseudo-inverse G^{-1}. As G is non-invertible, the proof consists of two steps: first reducing the problem to the invertible case, then proving that the inverse of an invertible Gevrey-class operator is again Gevrey-class (assuming that the inverse is uniformly bounded).

Lemma 5. *If G satisfies Eq. (14), then for all $k \geq 0$,*

$$\|(G^{-1})^{(k)}\| \leq \frac{2}{\Delta} \left(\frac{K}{\Delta} 2c \frac{4\pi^2}{3} \right)^k \frac{[k!]^{1+\alpha}}{(k+1)^2}. \tag{39}$$

Proof. First, write the pseudo-inverse using the Cauchy formula,

$$G^{-1} = \frac{1}{2\pi i} \oint_\Gamma \frac{1}{G-z} \frac{1}{z} \mathrm{d}z, \tag{40}$$

where $\Gamma = \{z \in \mathbb{C} \big| |z| = \Delta/2\}$ is a fixed, s-independent curve. Taking the kth derivative of Eq. (40) (with respect to s), we get

$$(G^{-1})^{(k)} = \frac{1}{2\pi i} \oint_\Gamma [(G-z)^{-1}]^{(k)} \frac{1}{z} \mathrm{d}z. \tag{41}$$

Thus, the norm of the pseudo-inverse can be bounded by the triangle inequality,

$$\|(G^{-1})^{(k)}\| \leq \max_{z \in \Gamma} \|[(G-z)^{-1}]^{(k)}\|. \tag{42}$$

Note that $G - z$ is invertible and $\|(G-z)^{-1}\| \leq 2/\Delta$ for $z \in \Gamma$. We now show that $(G-z)^{-1}$ is also Gevrey-class, more precisely that

$$\left\| \left[(G-z)^{-1} \right]^{(k)} \right\| \leq \frac{2}{\Delta} \left(\frac{2}{\Delta} Kc \frac{4\pi^2}{3} \right)^k \frac{[k!]^{1+\alpha}}{(k+1)^2} \tag{43}$$

for $k \geq 0$. To show this, we proceed by induction. For $k = 0$, the bound obviously holds. Taking the kth derivative of $(G-z)(G-z)^{-1} = \mathbb{1}$, we get

$$\left[(G-z)^{-1} \right]^{(k)} = (G-z)^{-1} \sum_{l=1}^{k} \binom{k}{l} (G-z)^{(l)} \left[(G-z)^{-1} \right]^{(k-l)}. \tag{44}$$

Using the induction hypothesis and collecting terms (notice that $l \geq 1$ and $k-l \leq k-1$), we get

$$\left\| \left[(G-z)^{-1}\right]^{(k)} \right\| \leq \frac{2}{\Delta}\left(\frac{2}{\Delta}Kc\right)^{k}\left(\frac{4\pi^2}{3}\right)^{k-1}\sum_{l=1}^{k}\binom{k}{l}\frac{[l!(k-l)!]^{1+\alpha}}{(l+1)^2(k-l+1)^2}. \tag{45}$$

Using Lemma 3 to upper-bound the sum in (45), we get

$$\left\| \left[(G-z)^{-1}\right]^{(k)} \right\| \leq \frac{2}{\Delta}\left(\frac{2}{\Delta}Kc\frac{4\pi^2}{3}\right)^{k}\frac{[k!]^{1+\alpha}}{(k+1)^2}. \tag{46}$$

This proves (43). Substituting this bound into Eq. (42) proves Lemma 5. □

Next, we prove that the ground state is also Gevrey-class (with the special choice of the phase as above).

Lemma 6. *If G satisfies Eq. (14), then the ground state ϕ satisfies*

$$\left\|\phi^{(k)}\right\| \leq \left(2c\frac{K}{\Delta}\left(\frac{4\pi^2}{3}\right)^2\right)^{k}\frac{[k!]^{1+\alpha}}{(k+1)^2}. \tag{47}$$

for all $k \geq 0$, where K, c and α are defined in Eq. (14) and Δ is the minimal gap of G.

Proof. We proceed by induction on k. For $k=0$, (47) just reads $\|\phi\| \leq 1$. For $k>0$, notice that $G\phi = 0$ and $\phi^{(1)} = -G^{-1}G^{(1)}\phi$ since the phase of ϕ is chosen such that $\left\langle \dot{\phi} \middle| \phi \right\rangle = 0$. Therefore,

$$\left\|\phi^{(k)}\right\| = \left\|\left[G^{-1}G^{(1)}\phi\right]^{(k-1)}\right\|. \tag{48}$$

Expanding the derivatives, we get

$$\left\|\phi^{(k)}\right\| \leq \sum_{l=0}^{k-1}\binom{k-1}{l}\left\|\left[G^{-1}G^{(1)}\right]^{(k-l-1)}\right\|\left\|\phi^{(l)}\right\|. \tag{49}$$

The right hand side can be bounded using the induction hypothesis as the higest derivative of ϕ appearing there is the $(k-1)$th. For that, we first derive a bound on the norm of the derivatives of $G^{-1}G^{(1)}$. This can be done by applying Lemma 4 to $G^{(1)}$ and G^{-1} and using Lemma 5 to obtain

$$\left\|[G^{-1}G^{(1)}]^{(k)}\right\| \leq \left(Kc\frac{2}{\Delta}\frac{4\pi^2}{3}\right)^{k+1}\frac{[(k+1)!]^{1+\alpha}}{(k+2)^2} \tag{50}$$

for $k \geq 0$. Substituting this bound into (49), we obtain

$$\|\phi^{(k)}\| \leq \sum_{l=0}^{k-1}\binom{k-1}{l}\left(Kc\frac{2}{\Delta}\frac{4\pi^2}{3}\right)^{k-l}\frac{[(k-l)!]^{1+\alpha}}{(k-l+1)^2}\left(Kc\frac{2}{\Delta}\left(\frac{4\pi^2}{3}\right)^2\right)^{l}\frac{[l!]^{1+\alpha}}{(l+1)^2}. \tag{51}$$

Notice that $l \leq k-1$, so

$$\|\phi^{(k)}\| \leq \left(Kc\frac{2}{\Delta}\frac{4\pi^2}{3}\right)^{k}\left(\frac{4\pi^2}{3}\right)^{k-1}\sum_{l=0}^{k-1}\binom{k-1}{l}\frac{[(k-l)!]^{1+\alpha}}{(k-l+1)^2}\frac{[l!]^{1+\alpha}}{(l+1)^2}. \tag{52}$$

Thus, using Lemma 3, we obtain

$$\left\|\phi^{(k)}\right\| \leq \left(Kc\frac{2}{\Delta}\left(\frac{4\pi^2}{3}\right)^2\right)^{k}\frac{[k!]^{1+\alpha}}{(k+1)^2}, \tag{53}$$

which proves Lemma 6. □

We are now in the position to bound $\|\dot{\phi}_{M-1}\|$. Instead of bounding it directly, we prove a general bound on all $\|\phi_j^{(k)}\|$. The desired bound is obtained then by setting $j = M-1$ and $k = 1$.

Lemma 7. *For all $j, k \geq 0$, the vectors ϕ_j and scalars φ_j defined in Eq. (18) satisfy*

$$\|\phi_j^{(k)}\| \leq A_1 A_2^j A_3^k \frac{[(k+j)!]^{1+\alpha}}{(k+j+1)^2} \quad \text{and} \quad |\varphi_j^{(k)}| \leq A_2^j A_3^k \frac{[(k+j)!]^{1+\alpha}}{(k+j+1)^2}, \tag{54}$$

where the constants A_1, A_2 and A_3 can be expressed with the constants appearing in Eq. (14):

$$A_1 = \frac{8\pi^2}{3}, \quad A_3 = 2c\frac{K}{\Delta}\left(\frac{4\pi^2}{3}\right)^2, \quad A_2 = 4c^2\frac{K^2}{\Delta^3}\left(\frac{4\pi^2}{3}\right)^5. \tag{55}$$

Proof. We proceed by induction on j, using the recursion in relation (18). We first bound $|\varphi_j|$ using the induction hypothesis, then bound $|\varphi_j^{(k)}|$ for $k > 0$ before bounding $\|\phi_j^{(k)}\|$.

Base case. We have $\varphi_0(s) = 1$ and $\phi_0(s) = \phi(s)$, so (54) holds for $j = 0$ since

$$A_1 \geq 1 \quad \text{and} \quad A_3 \geq 2c\frac{K}{\Delta}\left(\frac{4\pi^2}{3}\right)^2. \tag{56}$$

Bound on $|\varphi_j|$, $j \geq 1$. $|\varphi_j|$ can be bounded by the maximum value of the integrand in Eq. (18),

$$|\varphi_j| \leq \|G^{-1}\dot{\phi}\|\|\dot{\phi}_{j-1}\| \leq \|G^{-1}\| \cdot \|\dot{\phi}\| \cdot \|\dot{\phi}_{j-1}\|. \tag{57}$$

Using the bound on $\|\dot{\phi}\|$ from Lemma 6, $\|G^{-1}\|$ from Lemma 7 and the induction hypothesis on $\|\dot{\phi}_{j-1}\|$, we get

$$|\varphi_j| \leq \frac{2}{\Delta} \cdot A_3 \frac{1}{4} \cdot A_1 A_2^{j-1} A_3 \frac{[(j)!]^{1+\alpha}}{(j+1)^2} \leq A_2^j \frac{[(j)!]^{1+\alpha}}{(j+1)^2} \tag{58}$$

since

$$1 \geq A_1 \frac{A_3^2}{A_2} \frac{2}{\Delta} \frac{1}{4}. \tag{59}$$

Bound on $\|\varphi_j^{(k)}\|$. We now bound $|\varphi_j^{(k)}|$ for $k > 0$. First, from the induction hypothesis,

$$\|\dot{\phi}_{j-1}^{(k)}\| \leq A_1 A_2^{j-1} A_3^{k+1} \frac{[(k+j)!]^{1+\alpha}}{(k+j+1)^2}. \tag{60}$$

Then, using Lemma 4 and Lemma 5, we get that for all $k \geq 0$,

$$\left\|\left(G^{-1}\dot{\phi}_{j-1}\right)^{(k)}\right\| \leq \frac{4\pi^2}{3}\frac{2}{\Delta} A_1 A_2^{j-1} A_3^{k+1} \frac{[(k+j)!]^{1+\alpha}}{(k+j+1)^2}. \tag{61}$$

Moreover, from Lemma 6,

$$\|\dot{\phi}^{(k)}\| \leq A_3^{k+1} \frac{[(k+1)!]^{1+\alpha}}{(k+2)^2}. \tag{62}$$

Since $\dot{\varphi}_j = \left\langle \dot{\phi} \middle| G^{-1}\dot{\phi}_{j-1} \right\rangle$, Lemma 4, Eq. (61) and (62) imply that

$$|\varphi_j^{(k+1)}| = |\dot{\varphi}_j^{(k)}| \leq \left(\frac{4\pi^2}{3}\right)^2 \frac{2}{\Delta} A_3 A_1 A_2^{j-1} A_3^{k+1} \frac{[(k+j+1)!]^{1+\alpha}}{(k+j+2)^2} \tag{63}$$

for $k \geq 0$. Changing $k+1$ to k gives

$$|\varphi_j^{(k)}| \leq \left(\frac{4\pi^2}{3}\right)^2 \frac{2}{\Delta} A_3 A_1 A_2^{j-1} A_3^{k} \frac{[(k+j)!]^{1+\alpha}}{(k+j+1)^2} \leq A_2^j A_3^k \frac{[(k+j)!]^{1+\alpha}}{(k+j+1)^2}, \tag{64}$$

since $1 \geq A_1 \frac{A_3}{A_2} \frac{2}{\Delta} \left(\frac{4\pi^2}{3}\right)^2$.

Bound on $\|\phi_j^{(k)}\|$. We now bound $\|\phi_j^{(k)}\|$. Using the bound on $\|\phi^{(k)}\|$ from Lemma 6 and $|\varphi_j^{(k)}|$, Lemma 4 implies

$$\|(\varphi_j\phi)^k\| \leq \frac{4\pi^2}{3} A_2^j A_3^k \frac{[(k+j)!]^{1+\alpha}}{(k+j+1)^2}. \tag{65}$$

Finally, using Eq. (18),

$$\|\phi_j^{(k)}\| \leq 2\max\left(\left\|(\varphi_j\phi)^{(k)}\right\|, \left\|\left(G^{-1}\dot{\phi}_{j-1}\right)^{(k)}\right\|\right). \tag{66}$$

Hence, since

$$2\frac{4\pi^2}{3} \leq A_1 \quad \text{and} \quad \frac{4\pi^2}{3}\frac{2}{\Delta}A_3 \leq A_2, \tag{67}$$

we obtain

$$\|\phi_j^{(k)}\| \leq A_1 A_2^j A_3^k \frac{[(k+j)!]^{1+\alpha}}{(k+j+1)^2}, \tag{68}$$

which finishes the proof of Lemma 7 and hence the proof of Theorem 1. □

II. LOCALITY OF LOCAL ADIABATIC CHANGE

We show in this section that G_n and $\tilde{G}_n$, as defined in Eq. (7) and in Eq. (12) in the main text, generate basically the same dynamics. The proof relies on $\tilde{G}(0)$ being frustration-free, and a runtime of $\tau = O(\log^{1+\alpha}(N/\varepsilon))$, because it turns out that the achieved locality scales linearly with the runtime. We also use the Lieb-Robinson bound [5–8], which states that if H is a local (possibly time-dependent) Hamiltonian with uniformly bounded interaction strengths, $U(t)$ is the unitary evolution generated by H, and O_A, O_B are operators supported on regions A, B, respectively, then

$$\|[U(t)O_A U^\dagger(t), O_B]\| \leq c\min(|A|,|B|)\|O_A\|\|O_B\|\exp\left(\gamma t - \nu L\right), \tag{69}$$

where L is the distance between A and B, and c, γ, ν are constants depending only on the geometry of the lattice and the maximum interaction strength.

The following theorem justifies the replacement of (12) with (7) in the main text, without significantly altering the evolution and thus the final state.

Theorem 8. *Let* $\tilde{G}(s) = \sum_{\mu\in\Upsilon} G_\mu(s)$ *be a frustration-free Hamiltonian path with* $O(N)$ *local terms such that* $|\operatorname{supp}\frac{\mathrm{d}}{\mathrm{d}s}\tilde{G}| = O(1)$*, and let* G *be a localised version of* $\tilde{G}$*, i.e.,*

$$G(s) = \sum_{\mu\in\Upsilon'} G_\mu(s), \quad \Upsilon' = \left\{\mu\in\Upsilon \mid d\Big(\operatorname{supp}\frac{\mathrm{d}}{\mathrm{d}s}\tilde{G}, \operatorname{supp}G_\mu\Big) < \chi\tau\right\} \tag{70}$$

for some constant χ *and adiabatic runtime* τ*. Let* ψ *and* $\tilde{\psi}$ *be the evolved states under* G *and* $\tilde{G}$ *respectively, i.e.,*

$$i\frac{\mathrm{d}}{\mathrm{d}t}\psi(t) = G\left(\frac{t}{\tau}\right)\psi(t), \quad i\frac{\mathrm{d}}{\mathrm{d}t}\tilde{\psi}(t) = \tilde{G}\left(\frac{t}{\tau}\right)\tilde{\psi}(t), \quad t\in[0,\tau], \quad \psi(0) = \tilde{\psi}(0) = \phi(0), \tag{71}$$

where $\phi(0)$ *is the ground state of* $\tilde{G}(0)$*. Then, for sufficiently large* $\chi = O(1)$*,*

$$\|\tilde{\psi}(\tau) - \psi(\tau)\| \leq cN^2\tau^2 e^{(\gamma - v\chi/2)\tau}, \tag{72}$$

where c, γ, v *are the constants from* (69)*. In particular, if* $\tau = O(\log^{1+\alpha}(N/\varepsilon))$*, then*

$$\|\tilde{\psi}(\tau) - \psi(\tau)\| \leq \varepsilon/N \tag{73}$$

for sufficiently large $\chi = O(1)$*.*

Proof. For any $\Omega \subseteq \Upsilon$, let $G_\Omega = \sum_{\mu\in\Omega} G_\mu$ and $U_\Omega(t,s)$ be the unitary dynamics generated by G_Ω. Then U_Ω satisfies

$$\partial_t U_\Omega(t,s) = -iG_\Omega(t/\tau)U_\Omega(t,s), \tag{74}$$

$$\partial_s U_\Omega(t,s) = -iU_\Omega(t,s)G_\Omega(s/\tau), \tag{75}$$

$$U_\Omega(t,s) = \mathcal{T}\exp\left\{-i\int\limits_s^t dt' G_\Omega(t'/\tau)\right\}. \tag{76}$$

Notice that $G_{\Upsilon'} = G$ and $G_\Upsilon = \tilde{G}$. We write $U = U_{\Upsilon'}$ and $\tilde{U} = U_\Upsilon$. Let B be the boundary of Υ', that is, $B = \{\mu \in \Upsilon \mid d(\lambda,\mu) = \lceil \chi\tau \rceil\}$ and $\bar{B} = \Upsilon \backslash B$. Then, since $\tilde{G}(0)$ is frustration-free and all terms outside of Υ' are constant, $U_{\bar{B}}(t,0)\phi(0) = U(t,0)\phi(0)$ as $U_{\bar{B}} = U \otimes U_{\bar{B}\backslash\Upsilon'}$ and $U_{\bar{B}\backslash\Upsilon'}\phi(0) = \phi(0)$. In other words, $G_{\bar{B}}$ generates the same dynamics as G. Thus,

$$\|\tilde{\psi}(\tau) - \psi(\tau)\| = \|\tilde{U}\phi(0) - U\phi(0)\| = \|\tilde{U}\phi(0) - U_{\bar{B}}\phi(0)\| = \|\phi(0) - \tilde{U}^\dagger U_{\bar{B}}\phi(0)\|, \tag{77}$$

where $\tilde{U}$ and $U_{\bar{B}}$ are evaluated at $(\tau, 0)$. Let $V(t) = \tilde{U}^\dagger(t,0)U_{\bar{B}}(t,0)$. Then, since $G_B = \tilde{G} - G_{\bar{B}}$, Eq. (74) implies

$$\frac{\mathrm{d}}{\mathrm{d}t}V = i\tilde{U}^\dagger(t,0)G_B(t/\tau)\tilde{U}(t,0)V(t). \tag{78}$$

We now approximate V with a local unitary to obtain a bound for (77). Let $X = \{\mu \in \Upsilon \mid d(\mu, B) \le r\}$ for some r to be specified below, and let

$$V_X(t) = \mathcal{T}\exp\left\{i\int_0^t dt' U_X^\dagger(t',0)G_B(t'/\tau)U_X(t',0)\right\} \tag{79}$$

For $r = \frac{1}{2}\chi\tau$ and sufficiently large $\chi = O(1)$, X and $\operatorname{supp}\dot{G}$ are disjoint since $|\operatorname{supp}\dot{G}| = O(1)$, so $G_X(t/\tau) = G_X(0)$ for all $t \in [0,\tau]$. Because of frustration-freeness, $G_X(t/\tau)\phi(0) = 0$, and thus the dynamics generated by G_X acts trivially on the initial state, i.e., $U_X(t,0)\phi(0) = \phi(0)$. Thus, V_X also acts trivially on the initial state, $V_X(t)\phi(0) = \phi(0)$. Hence, substituting this into Eq. (77), we get

$$\|\phi(0) - \tilde{U}^\dagger U\phi(0)\| = \|V_X\phi(0) - \tilde{U}^\dagger U_{\bar{B}}\phi(0)\| = \|\phi(0) - V_X^\dagger V\phi(0)\|. \tag{80}$$

From the definition of V and of V_X,

$$\frac{d}{dt}(V_X^\dagger V) = iV_X^\dagger(\tilde{U}^\dagger G_B\tilde{U} - U_X^\dagger G_B U_X)V, \tag{81}$$

where G_B is evaluated at t/τ. Thus, by integrating (81) and using the triangle inequality and unitary invariance of the operator norm,

$$\|\phi(0) - V_X^\dagger V\phi(0)\| \le \int_0^\tau \mathrm{d}t\|\tilde{U}^\dagger G_B\tilde{U} - U_X^\dagger G_B U_X\| = \int_0^\tau \mathrm{d}t\|G_B - \tilde{U}U_X^\dagger G_B U_X\tilde{U}^\dagger\|, \tag{82}$$

where the unitary evolutions are taken from 0 to t and G_B is evaluated at t/τ. Observe that

$$\partial_s\left(\tilde{U}(t,s)U_X^\dagger(t,s)\right) = -i\tilde{U}(t,s)G_{\bar{X}}(s/\tau)U_X^\dagger(t,s), \tag{83}$$

where $\bar{X} = \Upsilon\backslash X$. Integrating (83) over s, we get

$$G_B(t/\tau) - \tilde{U}U_X^\dagger G_B(t/\tau)U_X\tilde{U}^\dagger = -i\int_0^t \mathrm{d}s\tilde{U}(t,s)\left[G_{\bar{X}}(s/\tau), U_X^\dagger(t,s)G_B(t/\tau)U_X(t,s)\right]\tilde{U}^\dagger(t,s). \tag{84}$$

Therefore, using the triangle inequality and the unitary invariance of the norm, we get

$$\|\psi(\tau) - \tilde{\psi}(\tau)\| \le \int_0^\tau \mathrm{d}t\int_0^t \mathrm{d}s\left\|\left[U_X^\dagger(t,s)G_B(t/\tau)U_X(t,s), G_{\bar{X}}(s/\tau)\right]\right\| \tag{85}$$

$$\le \int_0^\tau \mathrm{d}t\int_0^t ds\; cN^2e^{\gamma(t-s)-\nu r} \le cN^2\tau^2e^{\gamma\tau-\nu r}, \tag{86}$$

where the second line follows from the Lieb-Robinson bound as B and $\bar{X}$ are separated by a distance $r = \frac{1}{2}\chi\tau$. This proves Theorem 8. □

III. RELAXATIONS ON THE ASSUMPTION OF A UNIFORM GAP ALONG THE PATH

In this section, we show that the assumption of a spectral gap along the entire path of $\tilde{G}_n$ can be relaxed.

Theorem 9. *Suppose that $\tilde{G}_n(0)$ has a spectral gap of at least $\delta > 0$. Then $\tilde{G}_n(s)$ has a spectral gap of at least $q_0^2\delta$ for all $s \in [0,1]$, where q_0 satisfies that $\mathbb{1} \geq Q_\lambda \geq q_0\mathbb{1}$ (with Q_λ as in Eq. (3) in the main text). In particular, a uniform gap as defined in the main text implies a constantly lower bounded gap along the entire Hamiltonian path in the given algorithm.*

Proof. Since $\tilde{G}_n(0)$ is positive semidefinite and has a non-trivial kernel, the spectral gap condition of $\tilde{G}_n(0)$ is equivalent to $G_n(0)^2 \geq \delta G_n(0)$. Note that $\mathbb{1} \geq A_{n,m}(s) \geq q_0\mathbb{1}$ (with $A_{n,m}$ as in Eq. (9)). Let $X_n(s) = q_0^2 A_{n,n}^{-1}(s)\tilde{G}_n(0)A_{n,n}^{-1}(s)$. Then,

$$\tilde{G}_n(s) = \sum_{\mu\in\Upsilon}\left(\prod_{m:\lambda_m\in\Lambda_\mu} A_{n,m}^{-1}(s)\right)P_\mu\left(\prod_{m:\lambda_m\in\Lambda_\mu} A_{n,m}^{-1}(s)\right) \tag{87}$$

$$\geq q_0^2 A_{n,n}^{-1}(s)\sum_{\mu\in\Upsilon}\left(\prod_{\substack{m:\lambda_m\in\Lambda_\mu\\ m\neq n}} A_{n,m}^{-1}(s)\right)P_\mu\left(\prod_{\substack{m:\lambda_m\in\Lambda_\mu\\ m\neq n}} A_{n,m}^{-1}(s)\right)A_{n,n}^{-1}(s) \tag{88}$$

$$= q_0^2 A_{n,n}^{-1}(s)\tilde{G}_n(0)A_{n,n}^{-1}(s) \tag{89}$$

$$= X_n(s), \tag{90}$$

where we used in the second line $\|A_{n,n}^{-1}(s)\| \leq q_0^{-1}$. Notice that $\tilde{G}_n(s)$ and $X_n(s)$ have the same kernel and are both positive semidefinite. Thus, the gap of $\tilde{G}_n(s)$ is lower bounded by the gap of $X_n(s)$. But since $G_n(0)^2 \geq \delta G_n(0)$, we also have $X_n(s)^2 \geq q_0^2\delta X_n(s)$ since $A_{n,n}^{-2}(s) \geq \mathbb{1}$. Thus, $\tilde{G}_n(s)$ has a spectral gap of at least $\Delta_n \geq q_0^2\delta$. □

IV. GIBBS STATE PREPARATION IN THE NON-COMMUTING CASE FOR HIGH TEMPERATURES

The algorithm we presented to prepare a purifiaction of the Gibbs state of a Hamiltonian used explicitly that the Hamiltonian is a sum of commuting terms. Thus, one may wonder if one can apply it directly to Gibbs states of non-commuting Hamiltonians H. The genaral answer is no. The reason is that even though a parent Hamiltonian can still be defined as

$$G^{nl}(\beta) = \sum_{\mu\in\Upsilon} G_\mu^{nl}(\beta) = \sum_{\mu\in\Upsilon} e^{\frac{\beta}{2}H}P_\mu e^{-\beta H}P_\mu e^{\frac{\beta}{2}H}, \tag{91}$$

now the terms are not local (hence the superscript nl), and the norm of each term may be exponentially large in N. Thus, adiabatic state preparation using (91) directly takes exponential time. However, in this section we show that for sufficiently high, but constant temperatures, one can approximate G^{nl} by an $r = O(\log N)$-local Hamiltonian G^r which is a sum of $O(\text{poly}(N))$ terms. We also show that in this case, G^{nl} (and thus also G^r) has a $\Omega(1)$ spectral gap and $O(N)$ norm. Because of the existence of the gap, the ground state of G^r is a good approximation of the ground state of G^{nl}.

Using the adiabatic theorem, the following algorithm runs in $O(\text{poly}(N))$ time for high enough (but $\Omega(1)$) temperatures and gives a good approximation of the purification of the Gibbs state of a non-commuting Hamiltonian:

1. Prepare the ground state of $G^r(0) = \sum_{\mu\in\Upsilon} P_\mu$

2. Calculate $G^r(\beta)$

3. Prepare adiabatically the ground state of $G^r(\beta)$

We first use the cluster expansion [9] to construct the approximating Hamiltonian G^r. We also show that the norm of G^{nl} is $O(N)$. Finally, we show that the gap of G^{nl} is $\Omega(1)$. For simplicity, assume that $H = \sum_{\lambda\in\Lambda}$ is a sum of nearest-neighbour interactions, although the results and proofs generalise to other types of interactions. We also assume that $\|h_\lambda\| \leq 1$.

Cluster expansion. We now show that G^{nl} can be approximated by an $r = O(\log N)$-local Hamiltonian G^r. More precisely, we show the following result.

Theorem 10. *For sufficiently small (but constant) β, there exists an $r = O(\log N)$-local Hamiltonian G^r with $O(\text{poly}(N))$ terms such that*

$$\|G^{nl} - G^r\| < O(1/\text{poly}(N)). \tag{92}$$

Moreover, the terms of G^r can be calculated in $O(\text{poly}(N))$ time.

For any function f defined on the subsets of Λ, define the *Möbius transformations*

$$\hat{f}(\Omega) := \sum_{\Theta\subseteq\Omega} f(\Theta), \tag{93}$$

$$\check{f}(\Omega) := \sum_{\Theta\subseteq\Omega} (-1)^{|\Omega\setminus\Theta|} f(\Theta). \tag{94}$$

It is straightforward to check that the following Lemma holds [10].

Lemma 11 (Möbius inversion)**.**

$$\hat{\check{f}} = \check{\hat{f}} = f. \tag{95}$$

For any $\Omega \subseteq \Lambda$, let $H_\Omega = \sum_{\lambda\in\Omega} h_\lambda$, and let $f_\mu(\Omega) = e^{\beta H_\Omega} P_\mu e^{-2\beta H_\Omega} P_\mu e^{\beta H_\Omega}$ for any $\mu \in \Upsilon$. Using Lemma 11, one can express G^{nl}_μ as

$$G^{nl}_\mu = f_\mu(\Lambda) = \sum_{\Omega\subseteq\Lambda} \check{f}_\mu(\Omega). \tag{96}$$

This so-called *cluster expansion* has many interesting properties.

Lemma 12. *Let $\mu \in \Upsilon$. If $\Omega \subseteq \Lambda$ is such that μ is disjoint from Ω and $\Theta \subseteq \Lambda$ is disjoint from Ω, then*

$$\check{f}_\mu(\Omega \cup \Theta) = 0. \tag{97}$$

Proof. We have

$$\check{f}_\mu(\Omega \cup \Theta) = \sum_{\Omega'\subseteq\Omega, \Theta'\subseteq\Theta} (-1)^{|\Omega\setminus\Omega'|}(-1)^{|\Theta\setminus\Theta'|} e^{\beta H_{\Theta'}} P_\mu e^{-2\beta H_{\Theta'}} P_\mu e^{\beta H_{\Theta'}} = 0, \tag{98}$$

since $H_{\Omega'}$ commutes with P_μ and with $H_{\Theta'}$, and the sum over Ω' is 0. □

Lemma 12 states that $\check{f}_\mu$ is non-zero only for connected sets of edges that, in addition, contain μ. Another interesting property of $\check{f}_\mu$ is that its norm can be bounded as follows.

Lemma 13. *For any $\Omega \subset \Lambda$ and any edge μ,*

$$\|\check{f}_\mu(\Omega)\| \le (e^{4\beta} - 1)^{|\Omega|}. \tag{99}$$

Proof. Expanding the exponentials, one gets

$$\check{f}_\mu(\Omega) = \sum_{\Theta\subseteq\Omega} (-1)^{|\Omega\setminus\Theta|} \sum_{w_1,w_2,w_3\in\Theta^*} \frac{(-\beta)^{|w_1|}\cdot(2\beta)^{|w_2|}\cdot(-\beta)^{|w_3|}}{|w_1|!\cdot|w_2|!\cdot|w_3|!} h_{w_1} P_\mu h_{w_2} P_\mu h_{w_3}, \tag{100}$$

where Θ^* is the set of all finite sequences of elements of Θ, and for any $w \in \Theta^*$, $|w|$ denotes the length of w and $h_w = h_{\lambda_1} \dots h_{\lambda_{|w|}}$ if $w = (\lambda_1, \dots, \lambda_{|w|})$.

Consider the set $A = \text{supp}(w_1) \cup \text{supp}(w_2) \cup \text{supp}(w_3)$. If $A \neq \Omega$, then the alternating sum in (100) over all Θ such that $A \subseteq \Theta \subseteq \Omega$ is 0. Thus,

$$\|\check{f}_\mu(\Omega)\| \le \sum_{\substack{w_1, w_2, w_3 \in \Omega^* \\ \text{supp}(w_1)\cup\text{supp}(w_2)\cup\text{supp}(w_3) = \Omega}} \frac{\beta^{|w_1|}\cdot(2\beta)^{|w_2|}\cdot\beta^{|w_3|}}{|w_1|!\cdot|w_2|!\cdot|w_3|!}. \tag{101}$$

But this is exactly a Möbius transform, so

$$\|\check{f}_\mu(\Omega)\| \leq \sum_{\Theta\subseteq\Omega}(-1)^{|\Omega\setminus\Theta|}\sum_{w_1,w_2,w_3\in\Theta^*}\frac{\beta^{|w_1|}\cdot(2\beta)^{|w_2|}\cdot\beta^{|w_3|}}{|w_1|!\cdot|w_2|!\cdot|w_3|!} = \sum_{\Theta\subseteq\Omega}(-1)^{|\Omega\setminus\Theta|}e^{4\beta|\Theta|} = (e^{4\beta}-1)^{|\Omega|}. \tag{102}$$

□

We are now in a position to prove Theorem 10.

Proof of Theorem 10. Using (96), we can write G_μ^{nl} as a sum of local terms where the norm of the terms decay exponentially with their support. As the number of terms with a given size is bounded by the lattice growth constant [11], $\|G_\mu^{nl}\| = O(1)$ above some temperature. Indeed, let η be the lattice growth constant, so that the number of sets of connected edges containing μ and of size M is bounded by $e^{\eta M}$. Then,

$$\|G_\mu^{nl}\| \leq \sum_{\Omega\subseteq\Lambda}\|\check{f}_\mu(\Omega)\| \leq \sum_{M\geq 0}e^{\eta M}(e^{4\beta}-1)^M = O(1) \tag{103}$$

if β is sufficiently small (but constant). In this case, G_μ^{nl} can be approximated by an r-local operator G_μ^r by omitting all connected sets Ω of size at least r. The error of this approximation is

$$\|G_\mu^{nl} - G_\mu^r\| \leq \sum_{|\Omega|\geq r}\|\check{f}_\mu(\Omega)\| \leq \sum_{M\geq r}e^{\eta M}(e^{4\beta}-1)^M = \frac{\left(e^\eta\left(e^{4\beta}-1\right)\right)^r}{1-e^\eta\left(e^{4\beta}-1\right)} = \frac{y^r}{1-y}, \tag{104}$$

where $y = e^\eta\left(e^{4\beta}-1\right)$. Therefore, above some constant temperature, the cluster expansion can be truncated at $|\Omega| \leq O(\log N)$, giving an error of $O(1/\operatorname{poly}(N))$. This results in a $O(\log(N))$-local Hamiltonian

$$G^r = \sum_\mu \sum_{|\Omega|\leq O(\log N)}\check{f}_\mu(\Omega) \tag{105}$$

with $O(\operatorname{poly}(N))$ terms. Note that this Hamiltonian can now be calculated in $O(\operatorname{poly}(N))$ time. Indeed, there are $O(\operatorname{poly}(N))$ terms $\check{f}_\mu(\Omega)$, and each term can be evaluated in $O(\operatorname{poly}(N))$ time since there are at most $O(\operatorname{poly}(N))$ subsets of each Ω. □

Gap of G^{nl}. It remains to be shown that at sufficiently high (but constant) temperatures, the parent Hamiltonian is gapped.

Theorem 14. *For sufficiently small (but constant) β, G^{nl} has a spectral gap of $\Omega(1)$.*

Proof. To show the existence of a gap, we use that $G^{nl} \geq 0$ is frustration-free, so it is enough to show that

$$(G^{nl})^2 \geq \Delta G^{nl} \tag{106}$$

for some $\Delta = \Omega(1)$. Expanding G^{nl}, (106) is equivalent to

$$\sum_\mu (G_\mu^{nl})^2 + \sum_{\mu\neq\nu}G_\mu^{nl}G_\nu^{nl} \geq \sum_\mu \Delta G_\mu^{nl}. \tag{107}$$

Using Eq. (104) with $r=1$, we get that G_μ^{nl} is close to $P_\mu = G_\mu^{r=1}$ for high temperatures and thus it is gapped and the gap is close to 1. Therefore, it is enough to show that for some other constant $\Delta' < 1$,

$$\sum_{\mu\neq\nu}G_\mu^{nl}G_\nu^{nl} \geq -\sum_\mu \Delta' G_\mu^{nl}. \tag{108}$$

We upper bound the r.h.s. by lower bounding $\sum_\mu G_\mu^{nl}$ as

$$\sum_\mu G_\mu^{nl} \geq \frac{1}{x}\sum_r e^{-r}\sum_{d(\mu,\nu)=r} G_\mu^{nl} + G_\nu^{nl}, \tag{109}$$

where $x = 2\sum_r e^{-r} C_d r^d = O(1)$ is the number of times a single term is counted, and d is the dimension of the lattice. Therefore, it is enough to prove that for a given r and any pair μ, ν with $d(\mu,\nu) = r$,

$$G_\mu^{nl} G_\nu^{nl} \geq -\Delta' \frac{1}{x} e^{-r} (G_\mu^{nl} + G_\nu^{nl}). \tag{110}$$

Note that the kernel of the LHS of (110) is contained in the kernel of the RHS. Next, $G_\mu^{nl} + G_\nu^{nl}$ can be lower bounded by

$$G_\mu^{nl} + G_\nu^{nl} \geq \frac{1}{2}\left(1 - P_{Ker(G_\mu^{nl}+G_\nu^{nl})}\right), \tag{111}$$

since $G_\mu^{nl} + G_\nu^{nl} \approx P_\mu + P_\nu$, which has gap 1, and at sufficiently high (but constant) temperature the difference is sufficiently small.

To lower bound the l.h.s of (110), write $G_\mu^{nl} = \left|G_\mu^{\lceil r/2\rceil}\right| + X_\mu$ and $G_\nu^{nl} = \left|G_\nu^{\lceil r/2\rceil}\right| + X_\nu$. $\left|G_\nu^{\lceil r/2\rceil}\right|$ and $\left|G_\mu^{\lceil r/2\rceil}\right|$ commute as they are supported on two disjoint regions, and they are positive, thus their product is also positive. The norm of X_μ, X_ν is bounded by (with $y = e^\eta\left(e^{4\beta} - 1\right)$ as defined in the proof of Theorem 10)

$$\|X_\mu\| = \|G_\mu - |G_\mu^{\lceil r/2\rceil}|\| \leq \|G_\mu - G_\mu^{\lceil r/2\rceil}\| + \|G_\mu^{\lceil r/2\rceil} - |G_\mu^{\lceil r/2\rceil}|\| \leq 3\frac{y^{\lceil r/2\rceil}}{1-y}, \tag{112}$$

since $\|G_\nu^{nl} - G_\nu^{\lceil r/2\rceil}\| \leq y^{\lceil r/2\rceil}/(1-y)$ by (104), and thus $G_\nu^{\lceil r/2\rceil} \geq -y^{\lceil r/2\rceil}/(1-y)$, so $\left|G_\nu^{\lceil r/2\rceil} - |G_\nu^{\lceil r/2\rceil}|\right| \leq 2y^{\lceil r/2\rceil}/(1-y)$. Using that above some constant temperature $\|G_\mu^r\| < 2$, we get that

$$\left\|G_\mu^{nl} G_\nu^{nl} - |G_\mu^{\lceil r/2\rceil}||G_\nu^{\lceil r/2\rceil}|\right\| \leq 18\frac{y^{\lceil r/2\rceil}}{1-y} \leq \Delta' \frac{1}{2x} e^{-r} \tag{113}$$

for sufficiently small (but constant) β. Therefore for any μ, ν pair, the following is true:

$$G_\mu^{nl} G_\nu^{nl} \geq -18\frac{y^{\lceil r/2\rceil}}{1-y}[1 - P_{Ker(G_\mu^{nl} G_\nu^{nl})}] \geq -\Delta' \frac{1}{2x} e^{-r}[1 - P_{Ker(G_\mu^{nl}+G_\nu^{nl})}] \geq -\Delta' \frac{1}{x} e^{-r}(G_\mu^{nl} + G_\nu^{nl}), \tag{114}$$

as the kernel of $G_\mu^{nl} G_\nu^{nl}$ contains the kernel of $G_\mu^{nl} + G_\nu^{nl}$. This proves Eq. (110) and thus Theorem 14. □

[1] G. Nenciu, Comm. Math. Phys. **152**, 479 (1993).
[2] G. A. Hagedorn and A. Joye, Journal of Mathematical Analysis and Applications **267**, 235 (2002).
[3] Exponentially small errors have also been reported in [12], however, $\xi(n)$ appearing in Eq. (22) of that paper should be defined as the supremum over S_γ instead of $[0,1]$, which implies a dependence of this quantity on N. Once this is taken into account, it is unclear how the arguments of that paper imply an exponentially small error for arbitrarily large runtimes. Nevertheless, numerical evidence in [13] suggests that the error can be viewed as exponentially small for sufficiently small runtimes.
[4] In the following, we will omit kets and bras to simply notation.
[5] E. H. Lieb and D. W. Robinson, Comm. Math. Phys. **28**, 251 (1972).
[6] M. Hastings and T. Koma, Comm. Math. Phys. **265**, 781 (2006), arXiv:math-ph/0507008 [math-ph].
[7] B. Nachtergaele and R. Sims, Comm. Math. Phys. **265**, 119 (2006), arXiv:math-ph/0506030 [math-ph].
[8] S. Bravyi, M. B. Hastings, and F. Verstraete, Phys. Rev. Lett. **97**, 050401 (2006), arXiv:quant-ph/0603121 [quant-ph].
[9] R. Kotecký and D. Preiss, Comm. Math. Phys. **103**, 491 (1986).
[10] See also A. Molnar, N. Schuch, F. Verstraete, and J. I. Cirac, Phys. Rev. B **91**, 045138 (2015), arXiv:1406.2973 [quant-ph].
[11] D. A. Klarner, Canad. J. Math. **19**, 851 (1967).
[12] D. A. Lidar, A. T. Rezakhani, and A. Hamma, Journal of Mathematical Physics **50**, 102106 (2009).
[13] A. T. Rezakhani, A. K. Pimachev, and D. A. Lidar, Phys. Rev. A **82**, 052305 (2010), arXiv:1008.0863 [quant-ph].

A.2 Area laws and efficient descriptions of quantum many-body states

Area laws and efficient descriptions of quantum many-body states

Yimin Ge and Jens Eisert

Prior to this work, it was a common jargon that quantum many-body states on regular lattices which satisfy a so-called *area law* are, in some sense, "easy" to describe. Such area laws require that certain Rényi entropies S_α of the partial state with respect to a subregion of the lattice scale at most with the boundary of that subregion rather than with its volume, as is the case for randomly chosen quantum many-body states with high probability. Proving such an area law for ground states of all local gapped Hamiltonians in two or more spatial dimensions has become a milestone open problem in the quantum information theory approach to condensed-matter physics. Yet, the quest for proving a general area law should only be seen as an intermediate step towards the bigger programme of understanding whether all ground states of local gapped Hamiltonians can be faithfully represented by efficient tensor network states. In this endeavour, it seems a key step to ask whether satisfying an area law automatically implies the existence of an efficient tensor network description. This has indeed been proven in one spatial dimension: any state that satisfies an area law for some S_α with $\alpha \in (0, 1)$ can be approximated to constant trace distance error by an MPS with polynomial bond dimension. This lead to the common belief that also in two or more spatial dimensions, any area law state can be captured by a PEPS with polynomial bond dimension.

In this work, we prove the existence of counterexamples to this conjecture: strictly speaking, area laws and the existence of efficient tensor network descriptions are unrelated. We show that there exist states which satisfy an area law for *every* S_α but still, no efficient tensor network description can be found. In fact, they cannot be captured by any states that have – in the broadest possible sense – an efficient classical description. The more general conclusion of this work is that while indeed, the set of area law states, commonly referred to as the "physical corner", is a very small subset of the large many-body Hilbert space, the set of efficiently describable states is in turn tiny compared to the "corner" of area law states.

In Section 2, we introduce a precise definition of our very general notion of efficient (i.e., polynomial) classical descriptions of quantum states (Definition 1). We only require that the Kolmogorov complexity of the list of coefficients of the quantum state in the standard basis scales at most polynomially with the system size. Hence, this definition includes an extremely large range of states. Three important examples of such efficient descriptions are highlighted: tensor networks, quantum circuits with post-selection, and eigenstates of local Hamiltonians.

The central observation of this work, stated as Theorem 6 in Section 3, is that roughly speaking, in two or more spatial dimensions, the set of states that satisfy an area law for S_α contains a subspace whose dimension scales exponentially with the system size. This remains true even if in addition we also require the states to be translationally invariant. The main idea of the proof is the observation that any state on a $(D-1)$-dimensional lattice satisfies a D-dimensional area law when that state is embedded into a D-dimensional lattice, and that this embedding can also be performed in a translationally invariant way.

Section 4 then makes precise the notion of approximability by polynomially classically describable states, and we show that any Hilbert space of exponentially large dimension contains a state that cannot be approximated by polynomially classically describable states to constant trace distance error (Theorem 8). In particular, there exist translationally invariant quantum many-body states in two or more spatial dimensions which satisfy an area law for all S_α but cannot be approximated by efficient tensor network states, quantum circuits with post-selection, or eigenstates of local Hamiltonians (Corollaries 9-11). The proof of Theorem 8 uses a counting argument of ϵ-nets. An alternative proof using communication complexity is later given in Appendix A.

Corollary 12 and Theorem 16 (which is proven in Appendix B) generalise Theorem 6 to states that are both translationally and rotationally invariant, while Corollary 13 (proven in Appendix C) uses error-correcting codes to show a variant with decaying correlation functions – a well-known feature of ground states of gapped Hamiltonians.

Statement of individual contribution

This work was motivated by a discussion between Jens Eisert and myself. Jens Eisert mentioned his belief that the commonly stated folklore conjecture of area law states automatically being describable by efficient tensor networks may in fact be wrong. I subsequently formulated both proofs of the existence of counterexamples, including the variants for translationally and rotationally invariant states as well as states with decaying correlation functions. I was also in charge of writing all parts of this article, with the exception of parts of Sections 1 and 7.

I, Yimin Ge, am the principal author of this article and was extensively involved in all parts of it.

Permission to include:

Yimin Ge and Jens Eisert.
Area laws and efficient descriptions of quantum many-body states.
New Journal of Physics, 18, 083026 (2016).

https://iopscience.iop.org/article/10.1088/1367-2630/18/8/083026

Betreff: Re: Permission to use article in thesis (doi:10.1088/1367-2630/18/8/083026)
Von: Permissions <permissions@ioppublishing.org>
Datum: 06.09.2019, 10:40
An: Yimin Ge <yimin.ge@mpq.mpg.de>

Dear Dr Ge

Thank you for your email and for taking the time to seek this permission.

When you transferred the copyright in your article to New Journal of Physics, you were granted back certain rights, including the right to include the Final Published Version of the article within any thesis or dissertation. Please note you may need to obtain separate permission for any third party content you included within your article.

Please include citation details, "© IOP Publishing & Deutsche Physikalische Gesellschaft. CC BY 3.0. Reproduced with permission. All rights reserved" and for online use, a link to the Version of Record.

The only restriction is that if, at a later date, you wanted your thesis/dissertation to be published commercially, further permission would be required.

I wish you the best of luck with the completion of your thesis/dissertation.

Kind regards,

Tom Slader
Editorial Assistant

Copyright & Permissions Team
Gemma Alaway – Senior Rights & Permissions Adviser
Christina Colwell - Rights & Permissions Assistant

Contact Details
E-mail: permissions@iop.org
For further information about copyright and how to request permission: https://publishingsupport.iopscience.iop.org/copyright-journals/
See also: https://publishingsupport.iopscience.iop.org/
Please see our Author Rights Policy https://publishingsupport.iopscience.iop.org/author-rights-policies/

New Journal of Physics
The open access journal at the forefront of physics

Deutsche Physikalische Gesellschaft DPG
IOP Institute of Physics
Published in partnership with: Deutsche Physikalische Gesellschaft and the Institute of Physics

PAPER

Area laws and efficient descriptions of quantum many-body states

Yimin Ge[1] **and Jens Eisert**[2,3]

[1] Max-Planck-Institut für Quantenoptik, D-85748 Garching, Germany
[2] Dahlem Center for Complex Quantum Systems, Freie Universität Berlin, D-14195 Berlin, Germany
[3] Author to whom any correspondence should be addressed.

E-mail: jenseisert@gmail.com

Keywords: entanglement, area laws, tensor network states, Kolmogorov complexity

CrossMark

OPEN ACCESS

RECEIVED
24 February 2016

REVISED
23 May 2016

ACCEPTED FOR PUBLICATION
31 May 2016

PUBLISHED
5 August 2016

Abstract

It is commonly believed that area laws for entanglement entropies imply that a quantum many-body state can be faithfully represented by efficient tensor network states—a conjecture frequently stated in the context of numerical simulations and analytical considerations. In this work, we show that this is in general not the case, except in one-dimension. We prove that the set of quantum many-body states that satisfy an area law for all Renyi entropies contains a subspace of exponential dimension. We then show that there are states satisfying area laws for all Renyi entropies but cannot be approximated by states with a classical description of small Kolmogorov complexity, including polynomial projected entangled pair states or states of multi-scale entanglement renormalisation. Not even a quantum computer with post-selection can efficiently prepare all quantum states fulfilling an area law, and we show that not all area law states can be eigenstates of local Hamiltonians. We also prove translationally and rotationally invariant instances of these results, and show a variation with decaying correlations using quantum error-correcting codes.

1. Introduction

Complex interacting quantum systems show a wealth of exciting phenomena, ranging from phase transitions of zero temperature to notions of topological order. A significant proportion of condensed matter physics is concerned with understanding the features emergent in quantum lattice systems with local interactions. However, naive numerical descriptions of such quantum systems require prohibitive resources, for the simple reason that the dimension of the underlying Hilbert space grows exponentially in the system size.

Yet, it has become clear in recent years that ground states—and a number of other natural states—usually occupy only a tiny fraction of this Hilbert space. This subset, which is sometimes referred to as the 'physical corner' of the Hilbert space (figure 3(a)), is commonly characterised by states having little entanglement. More precisely, they are characterised by the *area law* [1]: entanglement entropies grow only like the boundary area of any subset A of lattice sites

$$S(\rho_A) = O(|\partial A|) \tag{1}$$

and not extensively like its volume $|A|$ (figure 1). Such area laws have been proven for all gapped spin models in $D = 1$ [2–6]. In $D \geqslant 2$, area laws have only been proven in special cases, including free gapped bosonic and fermionic models [7–9], ground states in the same gapped phase as ones satisfying an area law [10, 11], models which have a suitable scaling for heat capacities [12], models whose Hamiltonian spectra satisfy related conditions [13, 14], frustration-free spin models [15], and models exhibiting local topological order [16]. The general expectation is that all gapped lattice models satisfy an area law. Proving a general area law for gapped lattice models in $D \geqslant 2$ has indeed become a milestone open problem in condensed-matter physics.

Area laws are at the core of powerful numerical algorithms, such as DMRG [17]. In $D = 1$, the situation is particularly clear: matrix-product states [18] essentially 'parameterise' those one-dimensional quantum states that satisfy an area law for some Renyi entropy S_α with $\alpha \in (0, 1)$. They approximate all such states provably well, which explains why essentially machine precision can be reached with such numerical tools [19, 20]. A

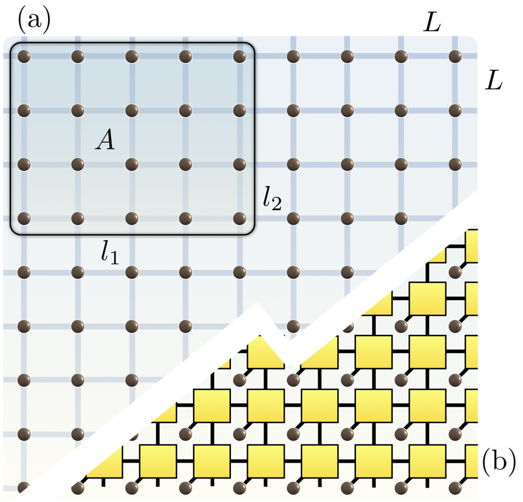


Figure 1. (a) There exist quantum states on D-dimensional cubic lattices in $D \geqslant 2$ such that $S_\alpha(\rho_A) = O(|\partial A|)$ for all $\alpha > 0$, but which cannot be approximated by efficient tensor network states, such as (b) polynomial projected entangled pair states.

common jargon is that similarly, projected entangled pair states (PEPS) [21], can approximate all states satisfying area laws in higher dimensions. In the same way, one expects those instances of tensor network states to capture the 'physical corner'.

In this work, we show that this jargon is not right: strictly speaking, area laws and the existence of efficient tensor network descriptions are unrelated. We show that there exist states that satisfy an area law for *every* Renyi entropy[4]

$$S_\alpha(\rho) = \frac{1}{1-\alpha}\log_2 \mathrm{tr}(\rho^\alpha), \qquad \alpha \in [0, \infty), \tag{2}$$

but still, no efficient PEPS can be found. The same holds for multi-scale entanglement renormalisation (MERA) ansatzes [22], as well as all classes of states that have a short description (the precise meaning of this will be defined below). Not even a quantum computer with post-selection can efficiently prepare all states satisfying area laws. Moreover, not all states satisfying area laws are eigenstates of local Hamiltonians.

These conclusions follow from the main result of this work: in $D \geqslant 2$, the set of states satisfying area laws for all S_α contains a subspace whose dimension scales exponentially with the system size. By considering a very general notion of quantum state descriptions based on the theory of *quantum Kolmogorov complexity* [23], we then infer that this large subspace cannot be captured by efficient tensor network states.

However, our results should not be seen to indicate that area laws are not appropriate *intuitive* guidelines for approximations with tensor network states. We rather provide a significant step towards precisely delineating the boundary between those quantum many-body states that can be efficiently captured and those that cannot. We thus contribute to the discussion why PEPS and other tensor network states approximate natural states so well. Area laws without further qualifiers are, strictly speaking, inappropriate for this purpose as the 'corner' they parameterise is exponentially large. This work is hence a strong reminder that the programme of identifying that boundary is not finished yet.

2. Classically efficiently described states

We first review the concept of efficient classical descriptions of quantum states. The focus is on tensor network states, but the notion of efficient classical descriptions can be formulated in a much more general way. For our purposes, the following definition of efficiently describable quantum states will suffice (see also [23] for alternative definitions).

Definition 1 (Classical descriptions). A *classical description* of a pure quantum state $|\psi\rangle \in \mathcal{S}((\mathbb{C}^d)^{\otimes N})$ is a Turing machine that outputs the coefficients of $|\psi\rangle$ in the standard basis $\{|\mathbf{x}\rangle : \mathbf{x} \in [d]^N\}$ and halts. The *length* of

[4] Here, $S_1 = \lim_{\alpha \downarrow 1} S_\alpha = S$ is the familiar von-Neumann entropy and S_0 the binary logarithm of the Schmidt rank.

the classical description is the size of the Turing machine[5]. We say that the description is *polynomial* if its length is polynomial in N.

We emphasise that for a polynomial classical description we only require the *size* of the Turing machine to be polynomial, but not the *run-time* (which is necessarily exponential). Notice that the shortest length of a classical description for a given quantum state is a natural generalisation of the Kolmogorov complexity[6] to quantum states [23].

Example 2 (Tensor networks). States that can be written as polynomial tensor networks (i.e., they are defined on arbitrary graphs with bounded degree, have at most $O(\mathrm{poly}(N))$ bond-dimension and their tensor entries' Kolmogorov complexity is at most $O(\mathrm{poly}(N)))$ are polynomially classically described states in the sense of definition 1. In particular, PEPS and MERA states with $O(\mathrm{poly}(N))$ bond-dimension and tensor entries of at most $O(\mathrm{poly}(N))$ Kolmogorov complexity are polynomially classically described states.

As a further interesting special case, we highlight that states that can be prepared by polynomial quantum circuits, even with post-selected measurement results, fall under our definition of classically described states.

Example 3 (Quantum circuits with post-selection). Suppose that $|\psi\rangle$ can be prepared by a quantum circuit of $O(\mathrm{poly}(N))$ gates from $|0\rangle^{\otimes O(\mathrm{poly}(N))}$, where we allow for post-selected measurement results in the computational basis. Then, a Turing machine that classically simulates the circuit constitutes a polynomial classical description in the sense of definition 1.

Example 4 (Eigenstates of local Hamiltonians). Suppose that $|\psi\rangle$ is an eigenvector of a local Hamiltonian with bounded interaction strength. Such Hamiltonians can be specified to arbitrary (but fixed) precision with polynomial Kolmogorov complexity. Thus, a Turing machine that starts from a polynomial description of the Hamiltonian and computes $|\psi\rangle$ by brute-force diagonalisation constitutes a polynomial classical description of $|\psi\rangle$ in the sense of definition 1.

3. Area laws and the exponential 'corner' of Hilbert space

Throughout the remainder of this work, we consider quantum lattice systems of local dimension d, arranged on a cubic lattice $[L]^D$ of fixed dimension $D > 1$, where $[L] \coloneqq \{0, \ldots, L-1\}$. We show in this section that the set of states satisfying area laws for all S_α contains subspaces of exponential dimension. This result is then used in section 4 to conclude that such states in general do not have an efficient classical description. The case $D = 1$ is excluded since in this case, the question at hand has already been settled with the opposite conclusion [19, 20]. The local dimension is small and taken to be $d = 3$ for most of this work. There is no obvious fundamental reason, however, why such a construction should not also be possible for $d = 2$.

In the focus of attention are states that satisfy an area law for all α-Renyi entropies, in particular also for $\alpha < 1$.

Definition 5 (Strong area laws). A pure state $|\psi\rangle \in \mathcal{S}((\mathbb{C}^d)^{\otimes L^D})$ is said to satisfy a *strong area law* if there exists a universal constant c such that for all regions $A \subset [L]^D$, we have $S_0(\psi_A) \leqslant c|\partial A|$, where $\psi_A = \mathrm{tr}_{\bar{A}}\ |\psi\rangle\langle\psi|$.

Since $S_\alpha(\rho) \leqslant S_0(\rho)$ for all $\alpha > 0$, strong area law states in this sense also exhibit area laws for all Renyi entropies. Definition 5 is hence even stronger than the area laws usually quoted [1, 19, 20]. Here and later, we write $\psi = |\psi\rangle\langle\psi|$. For simplicity, we will for the remainder of this paper restrict our consideration to cubic regions only. It should be clear, however, that all arguments generalise to arbitrary regions $A \subset [L]^D$.

We now turn to showing that the 'physical corner' of states satisfying area laws in this strong sense is still very large: it contains subspaces of dimension $\exp(\Omega(L^{D-1}))$. We prove this by providing a specific class of quantum states that have that property. At the heart of the construction is an embedding of states defined on a $(D-1)$-dimensional qubit lattice into the D-dimensional qutrit one. Denote with $\mathcal{H}_L \subset \mathrm{span}\{|1\rangle, |2\rangle\}^{\otimes L^{D-1}}$ the subspace of translationally invariant states (with respect to periodic boundary conditions) on a $(D-1)$-

[5] For readers who are not familiar with Turing machines, a less formal but for our purposes equivalent definition is that a classical description of $|\psi\rangle$ is a (classical) computer program that computes the coefficients of $|\psi\rangle$ in the standard basis. The length of the description is then simply the length of the program.

[6] Recall that the Kolmogorov complexity of a classical string w is the size of the shortest Turing machine (or computer program) that outputs w and halts. It can be thought of as the shortest possible (classical) description of w. For an introduction to Turing machines and Kolmogorov complexity, see e.g. [24].

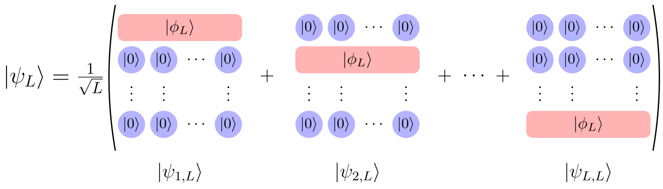


Figure 2. Schematic drawing of $|\psi_L\rangle$ in $D = 2$. $|\phi_L\rangle$ is an arbitrary translationally invariant state vector on L^{D-1} qubits with basis states $|1\rangle$, $|2\rangle$ in $(D-1)$-dimensions. Schmidt decompositions for $|\psi_L\rangle$ with respect to bi-partitions of the lattice can be readily obtained from the corresponding Schmidt decompositions of $|\phi_L\rangle$.

dimensional cubic lattice of L^{D-1} qubits. It is easy to show that $\dim(\mathcal{H}_L) \geqslant 2^{L^{D-1}}/L^{D-1}$. We start from the simplest translationally invariant construction on $\mathcal{H} := (\mathbb{C}^3)^{\otimes L^D}$ and discuss rotational invariance and decaying correlations below.

Theorem 6 (States satisfying strong area laws). *There exists an injective linear isometry $f\colon \mathcal{H}_L \to \mathcal{H}$ with the property that for all $|\phi_L\rangle \in \mathcal{H}_L$, $f(|\phi_L\rangle)$ satisfies a strong area law and is translationally invariant in all D directions.*

Proof. Given a state vector $|\phi_L\rangle \in \mathcal{H}_L$, define

$$|\psi_{k,L}\rangle := |0\rangle^{\otimes(k-1)L^{D-1}} \otimes |\phi_L\rangle \otimes |0\rangle^{\otimes(L-k)L^{D-1}} \in \mathcal{H}, \tag{3}$$

with $|\phi_L\rangle$ at the kth hyperplane of the lattice (figure 2). Define

$$|\psi_L\rangle := L^{-1/2}\sum_{k=1}^{L}|\psi_{k,L}\rangle, \tag{4}$$

which is translationally invariant. Any such state vector will satisfy a strong area law (in fact, a sub-area law): for any cubic subset $A = [l_1]\times\cdots\times[l_D]$, we have for the reduced state $(\psi_L)_A = \mathrm{tr}_{\bar{A}}\ |\psi_L\rangle\langle\psi_L|$ that

$$S_0((\psi_L)_A) \leqslant \log_2(2^{l_1\cdots l_{D-1}}l_D + 1) \leqslant 2\sum_{j=1}^{D}\prod_{k,k\neq j} l_k = |\partial A|, \tag{5}$$

where we used that the Schmidt rank with respect to the bi-partition $A, \bar{A}$ for each $|\psi_{k,L}\rangle$ with $k \in [l_D]$ is at most $2^{l_1\cdots l_{D-1}}$, and that since $|\phi_L\rangle$ is only supported on span$\{|1\rangle, |2\rangle\}$, the Schmidt vectors of $|\psi_{k,L}\rangle$ and $|\psi_{k',L}\rangle$ are orthogonal for $k \neq k' \in [l_D]$ such that in the distinguished Dth direction, the contribution to the Schmidt rank is additive and thus linear in l_D. Setting $f(|\phi_L\rangle) := |\psi_L\rangle$, we see that f has the desired properties.

4. Area laws and approximation by efficiently describable states

We now precisely state what we call an approximation of given pure states by polynomially classically described states.

Definition 7 (Approximation of quantum many-body states). A family of pure states $|\psi_L\rangle$ can be *approximated by polynomially classically described states* if for all $\varepsilon > 0$, there exist a polynomial p and pure states $|\omega_L\rangle$ with a classical description of length at most $p(L)$ such that for all L, $\| |\psi_L\rangle\langle\psi_L| - |\omega_L\rangle\langle\omega_L| \|_1 \leqslant \varepsilon$.

Note that this is exactly the sense in which matrix-product states provide an efficient approximation of all one-dimensional states that satisfy an area law for some S_α with $\alpha \in (0, 1)$ [19]. We remark that definition 7 can be weakened without altering the results. We now turn to the main result:

Theorem 8 (Impossibility of approximating area law states). *Let $\tilde{\mathcal{H}}_L$ be a Hilbert space of dimension* $\exp(\Omega(\mathrm{poly}(L)))$. *Then there exist states in $\tilde{\mathcal{H}}_L$ that cannot be approximated by polynomially classically described states. In particular, not all translationally invariant strong area law states can be approximated by polynomially classically described states.*

Theorem 8 can be easily proven using a counting argument of ϵ-nets. Indeed, the number of states that can be parameterised by $O(\mathrm{poly}(L))$ many bits is at most $2^{O(\mathrm{poly}(L))}$. However, an ϵ-net covering the space of pure states in $\mathbb{C}^q$ requires at least $(1/\varepsilon)^{\Omega(q)}$ elements [25], which is much larger than $2^{O(\mathrm{poly}(L))}$ if $q = \exp(\Omega(\mathrm{poly}(L)))$ (see also [26, 27] on the topic of ε-nets for many-body states). Thus, the set of quantum states in $\tilde{\mathcal{H}}_L$ that have a polynomial classical description cannot form an ε-net for $\tilde{\mathcal{H}}_L$, which proves theorem 8.

We nevertheless also review the more involved proof from [23] using communication complexity in appendix A. This proof could, due to its more constructive nature, provide some insight into the structure of some strong area law states that cannot be approximated by polynomially classically described states.

4.1. Tensor network states

We saw that our definition of polynomial classical descriptions encompasses all efficient tensor network descriptions. Thus

Corollary 9 (Tensor network states cannot approximate area law states). *There exist translationally invariant strong area law states that cannot be approximated by polynomial tensor network states in the sense of example 2. In particular, not all translationally invariant strong area law states can be approximated by polynomial PEPS or MERA states.*

Notice the restriction to tensor networks whose tensor entries have a polynomial Kolmogorov complexity. This is required to ensure that the tensor network description is in fact polynomial. Indeed, a classical description depending on only polynomially many parameters $\lambda_1, \ldots, \lambda_{O(\mathrm{poly}(N))}$ (e.g., a PEPS with polynomial bond-dimension) is not necessarily already polynomial—for the latter, it is also necessary that each of the λ_i themselves can be stored efficiently. The notion of Kolmogorov complexity allows for the most general definition of tensor networks that can be stored with polynomial classical memory.

4.2. Quantum circuits

Example 3 shows that states prepared by a polynomial quantum circuit with post-selected measurement results have a polynomial classical description. Thus

Corollary 10 (Post-selected quantum circuits cannot prepare area law states). *There exist translationally invariant strong area law states that cannot be approximated by a polynomial quantum circuit with post-selection in the sense of example 3.*

In the light of the computational power of post-selected quantum computation [28], this may be remarkable.

4.3. Eigenstates of local Hamiltonians

Example 4 shows that eigenstates of local Hamiltonians with bounded interaction strengths also have a polynomial classical description. Thus

Corollary 11 (Area law states without parent Hamiltonian). *There exist translationally invariant strong area law states that cannot be approximated by eigenstates of local Hamiltonians.*

5. Rotationally invarant states and area laws

So far, the states in consideration were translationally but not rotationally invariant. However, by taking the superposition of appropriate rotations of (4), one can alter the above argument such that all involved states are also rotationally invariant, i.e. remain invariant under 90° rotations of the lattice. The details of this construction are given in appendix B.

Corollary 12 (Approximation for translationally and rotationally invariant states). *There exist translationally and rotationally invariant strong area law states that cannot be approximated by polynomially classically described states. In particular, corollaries 9–11 also hold for translationally and rotationally invariant states.*

6. Decaying correlations and area laws

One might wonder whether an exponentially dimensional subspace of strong area law states can be constructed while imposing decaying two-point correlations for distant observables, a property known to occur in ground states of local gapped Hamiltonians [29, 30]. It follows immediately from their definition that the states constructed in theorem 6 (and theorem 16 in appendix B) already satisfy an algebraic decay. Indeed, for all L and all local observables A, B on disjoint supports separated by an arbitrary distance ℓ,

$$\langle\psi_L|\, AB\,|\psi_L\rangle - \langle\psi_L|\, A\,|\psi_L\rangle\langle\psi_L|\, B\,|\psi_L\rangle = O(L^{-1}) = O(\ell^{-1}). \tag{6}$$

Using quantum error-correcting codes, it is however also possible to construct variations of the previous results such that for all L and all local observables A, B with disjoint supports,

$$\langle\psi_L|\, AB\,|\psi_L\rangle - \langle\psi_L|\, A\,|\psi_L\rangle\langle\psi_L|\, B\,|\psi_L\rangle = 0. \tag{7}$$

The details of this construction are given in appendix C.

Corollary 13 (Approximation for area law states with vanishing correlations of local observables). *There exist strong area law states with vanishing two-point correlations of all local observables on disjoint supports that cannot be approximated by polynomially classically described states. In particular, corollaries 9–11 also hold for states with vanishing correlations of local observables on disjoint supports.*

The translationally and rotationally invariant construction only gives algebraic decay (equation (6)). However, we conjecture that there also exist strong area law states which are translationally and rotationally invariant and simultaneously have exponentially small correlations for all local observables, but still cannot be approximated by polynomially classically described states.

7. Conclusion and outlook

We have shown that the set of states satisfying an area law in $D \geqslant 2$ comprises many states that do not have an efficient classical description: they cannot be described by efficient tensor networks, cannot be prepared by polynomial quantum circuits with post-selected measurements, and are also not eigenstates of local Hamiltonians. We have hence proven that the connection between entanglement properties and the existence of an efficient description is far more intricate than anticipated. These results are based on the simple observation that an arbitrary quantum state in $(D-1)$ dimensions that is embedded into D dimensions satisfies a D-dimensional area law, thus implying that the set of area law states contains a subspace of exponential dimension. In other words, in $D \geqslant 2$, it is possible to 'dilute' the entanglement content and still arrive at a strong area law. We also demonstrated that the exponential scaling persists even if various physical properties, such as translational and rotational invariance, or decaying correlations of local observables, are imposed. We note however that while the latter can be extended to non-local observables of size $O(L^{D-1})$, our notion of decaying correlations is weaker than the exponential clustering property for ground states of gapped Hamiltonians, since this can involve all regions of unbounded size [5, 29]. It remains open whether our results are impeded if the stronger notion of exponential decay of correlations is imposed.

Area laws indeed suggest the expected entanglement behaviour of naturally occurring ground states. However, when put in precise contact with questions of numerical simulation, it turns out that satisfying an area law alone is not sufficient for efficient approximation. Picking up the metaphor of the introduction, the 'corner of states that can be efficiently described' is tiny compared to the 'physical corner' (figure 3).

A particularly exciting perspective arises from the observation that states with small entanglement content can go along with states having divergent bond dimensions in PEPS approximations. This may be taken as a suggestion that there may be states that are in the same phase if symmetries are imposed, but are being classified as being in different phases in a classification of phases of matter building upon tensor network descriptions [31–33]. It is the hope that the present work can be taken as a starting point of further endeavours towards understanding the complexity of quantum many-body states.

Acknowledgments

We thank J I Cirac, A Ferris, M Friesdorf, C Gogolin, Y-K Liu, A Molnár, X Ni, N Schuch, and H Wilming for helpful discussions. We acknowledge funding from the BMBF, the DFG (EI 519/7-1, CRC 183), the EU (RAQUEL, SIQS, AQuS), and the ERC (TAQ).

Appendix A. Proof of theorem 8 using communication complexity

We now review the alternative proof of theorem 8 using communication complexity, which was given in [23]. Suppose two distant parties, Alice and Bob, each possess an n-bit string, $\mathbf{x}$ and $\mathbf{y}$, respectively. No communication between Alice and Bob is allowed, but they can communicate with a third party, Charlie, whose task is to guess whether or not $\mathbf{x} = \mathbf{y}$. We demand that Charlie may guess the wrong answer with a small (fixed) probability of at most $\delta > 0$. This is called the *equality problem*, which we denote by $\mathrm{EQ}(n)$. We now state some

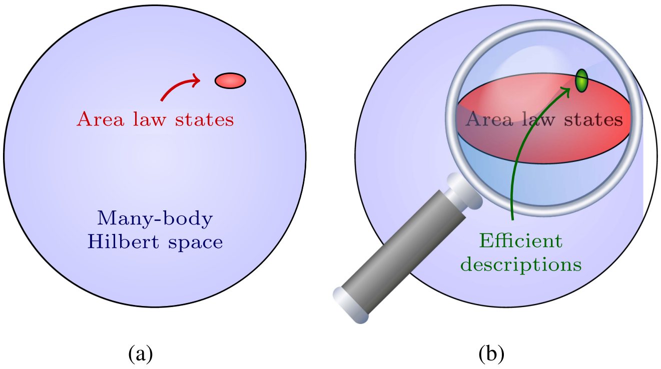


Figure 3. (a) The set of area law states is a tiny 'corner' of the many-body Hilbert space. (b) The set of states that can be efficiently described is tiny compared to the 'corner' of area law states.

known results [23, 34, 35] on the communication complexity, i.e. the minimum amount of communication required for solving the equality problem.

Lemma 14 (Equality problem for classical communication). *If Alice and Bob can only send classical information to Charlie, at least* $\Omega(\sqrt{n})$ *bits of communication are required to solve* EQ(n).

Lemma 15 (Quantum solution to equality problem).

(1) If Alice and Bob can send quantum information to Charlie, there exists a protocol for EQ(n) *using only* $O(\log n)$ *qubits of communication that is of the following form: Alice and Bob each prepare* $O(\log n)$ *qubit states* $|h(\mathbf{x})\rangle$ *and*[7] $|h(\mathbf{y})\rangle$*, respectively, which they send to Charlie. Charlie then applies a quantum circuit to* $|h(\mathbf{x})\rangle|h(\mathbf{y})\rangle|0\rangle$*, followed by a measurement of a single qubit whose outcome determines Charlie's guess.*

(2) There exists an $\varepsilon > 0$ *independent of n such that the protocol in (1) still works if instead, Alice and Bob send states to Charlie which are* ε*-close in trace distance*[8] *to* $|h(\mathbf{x})\rangle$ *and* $|h(\mathbf{y})\rangle$.

We now turn to the proof of theorem 8 .

Proof of theorem 8. We prove the claim by contradiction. Suppose that every state vector in $\tilde{\mathcal{H}}_L$ can be approximated by polynomially classically described states. Then in particular, all M-qubit states can be approximated by states with a classical description of length $O(\text{poly}(N))$, where $M := \lfloor \log_2 \dim(\tilde{\mathcal{H}}_L) \rfloor$. Fix $\delta \in (0,\ 1)$ and let $\varepsilon > 0$ be as in lemma 15 (2). By lemma 15 (1), we can choose n with $\log n = \Theta(M)$ such that M qubits of communication suffice to solve EQ(n).

By assumption, $|h(\mathbf{x})\rangle$ and $|h(\mathbf{y})\rangle$ can be ε-approximated by states which have an $O(\text{poly}(M))$ classical description. By lemma 15 (2), these states can be used instead of $|h(\mathbf{x})\rangle$ and $|h(\mathbf{y})\rangle$ in the quantum protocol to solve EQ(n). Now consider an alternative protocol using only classical communication to solve EQ(n) as follows: Alice and Bob send the classical description of their states to Charlie, who simulates the quantum circuit and the measurement from lemma 15 using the classical descriptions of the states. This protocol solves EQ(n) using only $O(\text{poly}(M)) = O(\text{poly}(\log n))$ bits of communication, contradicting lemma 14. Finally, by setting $\tilde{\mathcal{H}}_L := f(\mathcal{H}_L)$ with f and $\mathcal{H}_L$ as in theorem 6, the second part of theorem 8 follows. □

Appendix B. Translationally and rotationally invariant states

Corollary 12 follows directly from theorem 8 and the following theorem.

[7] The exact form of $|h(\mathbf{x})\rangle$ and $|h(\mathbf{y})\rangle$ is not important for our purpose—we will only need that they consist of $O(\log n)$ qubits. Interested readers are referred to [35].

[8] This was argued in [23] for the Euclidean vector distance but it is clear that the same holds for the trace distance.

Theorem 16 (Translationally and rotationally invariant area law states). *There exists an injective linear isometry g: $\mathcal{G}_L \to \mathcal{H}$ with* $\dim(\mathcal{G}_L) = \exp(\Omega(L^{D-1}))$ *such that for all* $|\phi_L\rangle \in \mathcal{G}_L$, $g(|\phi_L\rangle)$ *satisfies a strong area law and is translationally and rotationally invariant in all D directions.*

Theorem 16 can be proven with a minor modification of the proof of theorem 6. To start with, we replace $|\phi_L\rangle$ for each L by state vector on the translationally invariant subset $\mathcal{H}_L \subset (\mathbb{C}^2)^{\otimes L^{D-1}}$ which is also mirror symmetric, i.e. invariant under reflections, in all $(D - 1)$ directions. Notice that the exact choice of the plane of symmetry in a given direction does not matter since we assume $|\phi_L\rangle$ to be translationally invariant. With $|\psi_L\rangle$ as in (4), we then consider, for the entire $[L]^D$ lattice, state vectors of the form

$$|\Psi_L\rangle := D^{-1/2}\sum_{j=1}^{D}\mathcal{R}_j\ |\psi_L\rangle, \tag{B1}$$

where $\mathbb{I} = \mathcal{R}_1, \ldots, \mathcal{R}_D$ rotate the entire lattice system such that $|\phi_L\rangle$ is arranged along each line of the cubic lattice in dimension D. Such a state is translationally and rotationally invariant, following from mirror symmetry. These states satisfy a strong area law: for any cubic subset $A \subset [L]^D$,

$$(\psi_L)_A = \mathrm{tr}_{\bar{A}}\ |\psi_L\rangle\langle\psi_L| = D^{-1}\sum_{j=1}^{D}\mathrm{tr}_{\bar{A}}(\mathcal{R}_j|\psi_L\rangle\ \langle\psi_L|\mathcal{R}_j^\dagger), \tag{B2}$$

since for $j \neq k$,

$$\mathrm{tr}_{\bar{A}}(\mathcal{R}_j|\psi_L\rangle\ \langle\psi_L|\mathcal{R}_k^\dagger) = 0. \tag{B3}$$

This can be seen by taking the partial trace with respect to a set C first. For simplicity of notation, for $D = 2$, consider w.l.o.g. distinguished subsets $A \subset [L]^D$ for which $A \cap C = \varnothing$ for $C := L \times [L]$. Then

$$\mathrm{tr}_{\bar{A}}(|\psi_L\rangle\ \langle\psi_L|\mathcal{R}_2^\dagger) = \mathrm{tr}_{\bar{A}\setminus C}\sum_{\mathbf{x}\in S}\langle\mathbf{x}|\psi_L\rangle\ \langle\psi_L|\mathcal{R}_2^\dagger|\mathbf{x}\ \rangle = 0, \tag{B4}$$

where $S = \{\mathbf{x}|\ \exists j{:}\ x_j \neq 0 \wedge x_k = 0\ \forall k \in [L]\setminus\{j\}\}$. An analogous argument holds for any dimension D. From these considerations, it follows that the area law is inherited by the area law valid for each individual $\mathcal{R}_j|\psi_L\rangle$. It is furthermore clear that the exponential scaling of the dimension is not affected by restricting to the subspace $\mathcal{G}_L \subset \mathcal{H}_L$ of mirror symmetric states. □

Appendix C. States with vanishing two point correlation functions for local observables

To prove corollary 13, consider a non-degenerate $[\![n,\ k,\ \Delta]\!]$-quantum error-correcting code C with $k/n = \Theta(1)$ and $\Delta/n = \Theta(1)$ [36]. Here n denotes the block size and k the number of encoded qubits. Δ is the so-called distance of the code. Since C is non-degenerate, the reduced density matrix of any $\Delta - 1$ qubits of any state in the code space of C is maximally mixed. By choosing $n = L^{D-1}$ and considering

$$|\psi_L\rangle := |C(\chi_L)\rangle \otimes |0\rangle^{(L-1)L^{D-1}}, \tag{C1}$$

where $|C(\chi_L)\rangle$ is an arbitrary state vector in the code space of C, we see that for all L and all observables $A,\ B$ with disjoint support and whose joint support in the top hyperplane contains less than $\Delta = \Theta(L^{D-1})$ sites,

$$\langle\psi_L|\ AB\ |\psi_L\rangle - \langle\psi_L|\ A\ |\psi_L\rangle\langle\psi_L|\ B\ |\psi_L\rangle = 0. \tag{C2}$$

In particular, equation (C2) holds for local observables $A,\ B$. Clearly, states of the form (C1) obey a strong area law and since $k = \Theta(L^{D-1})$, we obtain a subspace of dimension $\exp(\Omega(L^{D-1}))$ of strong area law states with vanishing correlations of local observables. Corollary 13 now follows from theorem 8. □

References

[1] Eisert J, Cramer M and Plenio M B 2010 *Rev. Mod. Phys.* **82** 277

[2] Hastings M B 2007 *J. Stat. Mech.* P08024

[3] Arad I, Landau Z and Vazirani U 2012 *Phys. Rev.* B **85** 195145

[4] Arad I, Kitaev A, Landau Z and Vazirani U 2013 An area law and sub-exponential algorithm for 1d systems (arXiv:1301.1162)

[5] Brandão F G S L and Horodecki M 2013 *Nat. Phys.* **9** 721

[6] Huang Y 2014 Area law in one-dimension: degenerate ground states and renyi entanglement entropy (arXiv:1403.0327)

[7] Plenio M B, Eisert J, Dreissig J and Cramer M 2005 *Phys. Rev. Lett.* **94** 060503

[8] Cramer M and Eisert J 2006 *New J. Phys.* **8** 71

[9] Cramer M, Eisert J, Plenio M B and Dreissig J 2006 *Phys. Rev.* A **73** 012309

[10] Acoleyen K Van, Mariën M and Verstraete F 2013 *Phys. Rev. Lett.* **111** 170501

[11] Mariën M, Audenaert K M, Acoleyen K V and Verstraete F 2014 Entanglement rates and the stability of the area law for the entanglement entropy (arXiv:1411.0680)

[12] Brandao F G S L and Cramer M 2015 Entanglement area law from specific heat capacity *Phys. Rev.* B **92** 115134

[13] Hastings M B 2007 *Phys. Rev.* B **76** 035114
[14] Masanes L 2009 *Phys. Rev.* A **80** 052104
[15] de Beaudrap N, Ohliger M, Osborne T J and Eisert J 2010 *Phys. Rev. Lett.* **105** 060504
[16] Michalakis S 2012 Stability of the area law for the entropy of entanglement (arXiv:1206.6900)
[17] Schollwöck U 2011 *Ann. Phys.* **326** 96
[18] Verstraete F, Murg V and Cirac J I 2008 *Adv. Phys.* **57** 143
[19] Schuch N, Wolf M M, Verstraete F and Cirac J I 2008 *Phys. Rev. Lett.* **100** 030504
[20] Verstraete F and Cirac J I 2006 *Phys. Rev.* B **73** 94423
[21] Verstraete F and Cirac J I 2004 arXiv:cond-mat/0407066
[22] Vidal G 2007 *Phys. Rev. Lett.* **99** 220405
[23] Mora C, Briegel H and Kraus B 2007 *Int. J. Quant. Inf.* **05** 729
[24] Li M and Vitányi P M 2008 *An Introduction to Kolmogorov Complexity and Its Applications* 3rd edn (Berlin: Springer)
[25] Hayden P 2010 *Proc. Symp. in Applied Mathematics* **vol 68**
[26] Poulin D, Qarry A, Somma R and Verstraete F 2011 *Phys. Rev. Lett* **106** 170501
[27] Kliesch M, Barthel T, Gogolin C, Kastoryano M and Eisert J 2011 *Phys. Rev. Lett.* **107** 120501
[28] Aaronson S 2005 *Proc. R. Soc.* A **461** 3473
[29] Hastings M B and Koma T 2006 *Commun. Math. Phys.* **265** 781
[30] Hastings M B 2004 *Phys. Rev.* B **69** 104431
[31] Chen X, Gu Z-C and Wen X-G 2011 *Phys. Rev.* B **83** 035107
[32] Turner A M, Pollmann F and Berg E 2011 *Phys. Rev.* B **83** 075102
[33] Schuch N, Perez-Garcia D and Cirac I 2011 *Phys. Rev.* B **84** 165139
[34] Newman I and Szegedy M 1996 *Proc. 28th ACM Symp. on the Theory of Computing* (New York: ACM) pp 561–70
[35] Buhrman H, Cleve R, Watrous J and de Wolf R 2001 *Phys. Rev. Lett.* **87** 167902
[36] Gottesman D 1996 *Phys. Rev.* A **54** 1862

A.3 Faster ground state preparation and high-precision ground energy estimation with fewer qubits

Faster ground state preparation and high-precision ground energy estimation with fewer qubits

Yimin Ge, Jordi Tura, and J. Ignacio Cirac

In this work, we propose a general-purpose quantum algorithm for preparing ground states of quantum Hamiltonians from a given trial state. Compared with phase estimation, the runtime of our algorithm is exponentially better as a function of the allowed error, and at least quadratically better as a function of the overlap with the trial state. Moreover, we also show that our algorithm requires significantly fewer ancilla qubits than existing algorithms with comparable runtimes, and we show that it can also be used to determine an unknown ground energy to high precisions faster than with phase estimation.

The setup in this paper is as follows: for an $N \times N$ Hermitian matrix $\tilde{H}$ with spectrum contained in $[0,1]$ and known lower bound Δ to its spectral gap, the aim is to prepare a state which is ϵ-close to the ground state $|\lambda_0\rangle$ of $\tilde{H}$ from a trial state $|\phi\rangle$. The latter is given by a circuit which prepares $|\phi\rangle$ using Φ gates. We assume that its (generally unknown) overlap $\phi_0 = \langle\lambda_0|\phi\rangle$ with the ground state has a known lower bound $\chi = e^{-O(\log N)}$. We also assume the ability to efficiently perform Hamiltonian simulation of $\tilde{H}$ at a "base cost" of Λ gates.

This paper presents several versions of the algorithm for different setups. In the first case, if the ground energy λ_0 is known to a sufficiently good additive precision, Theorem 1 shows that an ϵ-close state to $|\lambda_0\rangle\langle\lambda_0|$ can be prepared with constant probability in a gate complexity of $\tilde{O}(\Lambda/(|\phi_0|\Delta) + \Phi/|\phi_0|)$, and using $O(\log N + \log\log\epsilon^{-1} + \log\Delta^{-1})$ qubits. The main idea of the algorithm, detailed in Section III, is to implement a suitable ground state projector using the "linear combination of unitaries" (LCU) Lemma. More precisely, we first transform $\tilde{H}$ into a Hamiltonian H such that $|\lambda_0\rangle$ is the unique eigenvector of H of eigenvalue ≈ 0, then approximate $\cos^M H$ as a linear combination of terms of the form e^{-iHt_k} which we then implement with some amplitude using Hamiltonian simulation and the LCU Lemma (an alternative approach using Chebyshev polynomials and quantum walks is presented in Appendix D). The final step is to use amplitude amplification to boost the overlap with the target state.

In Section IV, we present an adaptation of this algorithm to the case where λ_0 is not known beforehand. In that case, Theorem 2 shows that the same task can be achieved in a gate complexity of $\tilde{O}(\Lambda/(\chi\Delta^{3/2}) + \Phi/(\chi\sqrt{\Delta}))$ and the same number of qubits. The main new ingredient for this version is a subroutine which we call *minimum label finding* (Proposition 1). Roughly speaking, given a superposition of states entangled with a "label" register, that subroutine finds the smallest label value amongst the terms with at least a given amplitude. To implement the ground state projection, we first adapt the previous algorithm into a circuit controlled on an ancilla register $|E\rangle$, which runs the original algorithm assuming that the ground energy were E. We then divide $[0,1]$ into $L = \tilde{O}(\Delta^{-1})$ equally spaced values $E_0, \ldots, E_{L-1}$ and run the algorithm with $\sqrt{L^{-1}}\sum|E_j\rangle$ on the ancilla register. Using the minimum label finding subroutine, we then search for the smallest j for which the residual state of $|E_j\rangle$ has large norm.

Theorem 3 then shows that this algorithm can also be combined with phase estimation to obtain a better scaling in Δ at the expense of worsening the scaling in χ: by using phase estimation to first obtain a "crude" estimate of the ground energy, a runtime of $\tilde{O}(\Lambda/(\chi^3\Delta^\kappa) + \Lambda/(\chi\Delta^{(3-\kappa)/2}) + \Phi/(\chi\Delta^{(1-\kappa)/2}))$ is obtained, where $\kappa \in [0,1]$ can be chosen arbitrarily.

When the minimum label finding subroutine in the final step of Theorems 2 or 3 succeeds, then with high probability, the resulting E_j is a good approximation of the true ground energy.

This can be used to find the ground energy to a high precision, which is formally stated in Theorem 4.

The article also compares our results to previously known algorithms and we show that our algorithms exhibit the best scaling for both the runtime as well as the number of required qubits amongst all known ground space projection algorithms. To that end, in the appendices of this work, the runtime and qubit scalings for some previously known algorithms are analysed in greater detail than in the original works.

Statement of individual contribution

This work was motivated by several discussions between Jordi Tura, J. Ignacio Cirac, and myself. It was my idea to implement a suitable ground state projector using the LCU Lemma in order to extract the ground state from a trial state. Subsequently, with regular advice from J. Ignacio Cirac and occasional numerical support from Jordi Tura, I worked out the details of all variants of the algorithm, the minimum label finding subroutine, all proofs involved, as well as the comparisons to previously known algorithms. I was in charge of writing all parts of this article.

I, Yimin Ge, am the principal author of this article and was extensively involved in all parts of it.

Permission to include:

Yimin Ge, Jordi Tura, and J. Ignacio Cirac.
Faster ground state preparation and high-precision ground energy estimation with fewer qubits.
Journal of Mathematical Physics, 60, 022202 (2019).

Permission to Reuse Content

REUSING AIP PUBLISHING CONTENT

Permission from AIP Publishing is required to:

- republish content (e.g., excerpts, figures, tables) if you are not the author
- modify, adapt, or redraw materials for another publication
- systematically reproduce content
- store or distribute content electronically
- copy content for promotional purposes

To request permission to reuse AIP Publishing content, use RightsLink® for the fastest response or contact AIP Publishing directly at rights@aip.org and we will respond within one week:

For RightsLink, use Scitation to access the article you wish to license, and click on the Reprints and Permissions link under the TOOLS tab. (For assistance click the “Help” button in the top right corner of the RightsLink page.)

To send a permission request to rights@aip.org, please include the following:

- Citation information for the article containing the material you wish to reuse
- A description of the material you wish to reuse, including figure and/or table numbers
- The title, authors, name of the publisher, and expected publication date of the new work
- The format(s) the new work will appear in (e.g., print, electronic, CD-ROM)
- How the new work will be distributed and whether it will be offered for sale

Authors do **not** need permission from AIP Publishing to:

- quote from a publication (please include the material in quotation marks and provide the customary acknowledgment of the source)
- reuse any materials that are licensed under a Creative Commons CC BY license (please format your credit line: “Author names, Journal Titles, Vol.#, Article ID#, Year of Publication; licensed under a Creative Commons Attribution (CC BY) license.”)
- reuse your own AIP Publishing article in your thesis or dissertation (please format your credit line: “Reproduced from [FULL CITATION], with the permission of AIP Publishing”)
- reuse content that appears in an AIP Publishing journal for republication in another AIP Publishing journal (please format your credit line: “Reproduced from [FULL CITATION], with the permission of AIP Publishing”)
- make multiple copies of articles–although you must contact the Copyright Clearance Center (CCC) at www.copyright.com to do this

(…)

https://publishing.aip.org/resources/researchers/rights-and-permissions/permissions/ 03/12/2019

Journal of Mathematical Physics ARTICLE scitation.org/journal/jmp

Faster ground state preparation and high-precision ground energy estimation with fewer qubits

Cite as: J. Math. Phys. 60, 022202 (2019); doi: 10.1063/1.5027484
Submitted: 2 March 2018 • Accepted: 3 January 2019 •
Published Online: 6 February 2019

View Online Export Citation CrossMark

Yimin Ge, Jordi Tura, and J. Ignacio Cirac

AFFILIATIONS

Max-Planck-Institut für Quantenoptik, D-85748 Garching, Germany

ABSTRACT

We propose a general-purpose quantum algorithm for preparing ground states of quantum Hamiltonians from a given trial state. The algorithm is based on techniques recently developed in the context of solving the quantum linear system problem. We show that, compared to algorithms based on phase estimation, the runtime of our algorithm is exponentially better as a function of the allowed error, and at least quadratically better as a function of the overlap with the trial state. We also show that our algorithm requires fewer ancilla qubits than existing algorithms, making it attractive for early applications of small quantum computers. Additionally, it can be used to determine an unknown ground energy faster than with phase estimation if a very high precision is required.

I. INTRODUCTION

Quantum computers are expected to have a deep impact on the simulation of large quantum systems, as originally envisioned by Feynman.[1] Of particular interest is the potential ability to study both the dynamics and low energy properties of many-body quantum systems, which are usually inaccessible classically due to the exponential dimension of the underlying Hilbert space. Quantum computers do not suffer from this representability problem, as one can store states in a number of qubits that only scale logarithmically with that dimension. This fact can be used to develop very efficient algorithms to simulate the dynamics of quantum systems.[2–6] However, preparing certain physically relevant states, like the ground state of a many-body Hamiltonian, may be significantly more difficult. This can be seen as a consequence of ground state preparation likely being hard in full generality, as indeed many variations of ground state energy problems have been proven to be complete for the class QMA.[7] Nevertheless, preparing ground states of Hamiltonians has profound applications in several fields of science so that more efficient quantum algorithms than the ones existing[8–10] are highly desired. This could allow one, for instance, to prepare the initial states that are required to simulate quenches in quantum many-body systems, thus enabling the study of many intriguing and not fully understood phenomena, such as many-body localisation[11] or the presence of thermalisation in closed systems,[12] with quantum computers. The other applications include single-copy tomography[13] and the construction of QMA witnesses.[9] Furthermore, the ability to determine the ground energy of a Hamiltonian to a high precision also possesses many applications in the fields of physics and quantum chemistry,[14] and possibly even in quantum machine learning.[15]

Most existing quantum algorithms for ground state preparation are based on one of two methods. First, one could naively attempt to project a trial state $|\phi\rangle$ onto the ground state by measuring the energy of $|\phi\rangle$ using the phase estimation algorithm.[8] The probability of success is proportional to $|\phi_0|^2$, where ϕ_0 is the overlap of $|\phi\rangle$ with the ground state. Furthermore, straightforward application of phase estimation becomes expensive if a very high fidelity of the prepared state with the real ground state is required. A second class of algorithms is based on variants of the adiabatic algorithm.[16] Here, the target Hamiltonian $H(1)$ is connected to a trivial Hamiltonian $H(0)$ via a path $H(s)$, which is slowly changed from $H(0)$ to $H(1)$. The adiabatic theorem guarantees that if the initial state is the ground state of $H(0)$, which is assumed to be easily prepared, then for sufficiently long runtimes, the final state will be close to the ground state of $H(1)$. Rigorous bounds[17] on the runtime however depend inverse polynomially on the minimum spectral gap along the entire path $H(s)$, which is generally exponentially small and moreover extremely difficult to calculate or bound in practice. Thus, adiabatic algorithms are often only employed as *heuristic* methods to first obtain a state with (hopefully) non-trivial overlap with the ground state, which can then subsequently be used as the trial state in phase estimation.[18] This approach is expected to work significantly better than just using random trial states and is the current paradigm, e.g., for quantum chemistry applications.[14]

In this paper, we propose a quantum algorithm that significantly improves the part played by phase estimation in this approach. More generally, we consider the problem of preparing a good approximation of the ground state from a given trial state. We show that compared to using phase estimation, the runtime of our ground state preparation algorithm scales exponentially better in the allowed error to the real ground state, and polynomially better with the spectral gap and the overlap of the trial state with the ground state. We also show that, in case the ground energy is not known beforehand, the same algorithm can be used to obtain a good estimate of the ground energy to a high precision faster than is possible with phase estimation.

Unlike algorithms based on the adiabatic theorem, whose runtimes always depend on the minimum spectral gap along an entire path of Hamiltonians, all algorithms analysed in this paper only require a lower bound on the spectral gap of the target Hamiltonian. This is a significantly weaker assumption, and indeed, for many systems of interest such as in typical critical points, this gap is known to scale only inverse polynomially with the system size.

The outline of the remainder of the paper is as follows: in Sec. II, we give an overview of the results and a high-level overview of the ideas. In Sec. III, we give the technical details of the algorithm in case the ground energy is known beforehand. In Sec. IV, we present the technical details of the algorithm for both ground state preparation and high-precision ground energy estimation, in case the ground energy is unknown beforehand. We close the main part of the paper with some concluding remarks and open questions in Sec. V. Appendix A analyses the cost of finding the ground energy with phase estimation. In Appendix B, we demonstrate that if phase estimation is used for ground state preparation, an extremely precise estimate of the ground energy is required beforehand, and analyse the cost of doing so. In Appendix C, we analyse the filtering algorithm from Ref. 9. Finally, in Appendix D, we sketch an alternative approach (inspired by the "Chebyshev method" of Ref. 19) to the problem.

II. OVERVIEW OF RESULTS

Throughout this paper, let $\tilde{H}$ be an $N \times N$ Hermitian matrix such that its spectrum is contained in $[0, 1]$. We assume that we are given the ability to efficiently perform Hamiltonian simulation of $\tilde{H}$. More precisely, we require that $e^{\pm i\tilde{H}t}$ can be approximated to error ϵ' using $O(\Lambda t \mathrm{polylog}(N, 1/\epsilon'))$ elementary gates,[20] where Λ is the "base cost" of the simulation (e.g., if the simulation algorithm works in the oracle model[4] then Λ is the gate cost of the oracles). Let λ_0 be the lowest eigenvalue of $\tilde{H}$ and $|\lambda_0\rangle$ be the corresponding eigenstate. For simplicity of notation, we will assume that λ_0 is non-degenerate (all results in this paper trivially generalise to the case when λ_0 is degenerate, see Sec. V). Suppose that Δ is a known lower bound on the spectral gap of $\tilde{H}$.

Suppose moreover that we are given a circuit $\mathcal{C}_\phi$ using Φ elementary gates which prepares a trial state $|\phi\rangle$. Let $\phi_0 := \langle\lambda_0|\phi\rangle$ be its (generally unknown) overlap with the ground state and χ be a known lower bound on $|\phi_0|$. We will assume that $\chi = e^{-O(\log N)}$ throughout this paper. Notice that this is an extremely weak assumption, indeed, this is satisfied even for random states with high probability. The aim of this paper is to prepare a state ϵ-close to $|\lambda_0\rangle$ by (approximately) projecting $|\phi\rangle$ onto its ground state component.

Throughout this paper, we will use the computer science convention for the big-O notation. We furthermore use $\tilde{O}$ to denote the complexity up to polylogarithmic factors in N, Δ^{-1}, ϵ^{-1}, $|\phi_0|^{-1}$, and χ^{-1}. Our first result can now be stated as follows:

Theorem 1 (Ground state preparation for known ground energy). *Suppose that λ_0 is known to an additive precision of $O(\Delta/\log\frac{1}{\chi\epsilon})$, i.e., there exists a known real number $E > 0$ such that $|\lambda_0 - E| = O(\Delta/\log\frac{1}{\chi\epsilon})$ with certainty. Then, an ϵ-close state to $|\lambda_0\rangle\langle\lambda_0|$ can be prepared with constant probability in a gate complexity of*

$$\tilde{O}\left(\frac{\Lambda}{|\phi_0|\Delta} + \frac{\Phi}{|\phi_0|}\right) \tag{1}$$

and using

$$O\left(\log N + \log\log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right) \tag{2}$$

qubits. Moreover, a flag qubit indicates success.

Although other quantum algorithms for this or similar purposes have previously been proposed,[8–10] to the best of our knowledge, the algorithm in this paper, for the case of known ground energy, exhibits the best scaling for both the runtime and the number of qubits amongst all existing algorithms so far (see Table I). For example, the common approach of combining phase estimation with amplitude amplification[21] has a runtime that is exponentially worse in ϵ and moreover quadratically worse in $|\phi_0|$ (see Appendix B). In fact, an inverse polynomial dependence on ϵ is common to almost all algorithms that are based on phase estimation.[8,10] To the best of our knowledge, the only exception is a filtering method proposed by Poulin and Wocjan,[9] which was originally designed to quadratically improve the runtime dependence on $|\phi_0|$ to obtain a state with low expected energy, and which, as we prove in Appendix C, can also be used to obtain the ground state with a runtime scaling that is polylogarithmic in ϵ^{-1} with a suitable choice of parameters. This however comes at the cost of requiring significantly more ancilla qubits, which makes it challenging for early applications of small quantum computers.

The algorithm can also be adapted for the case when the ground energy is not known beforehand.

Theorem 2 (Ground state preparation for unknown ground energy). *If the ground energy is not known beforehand and $\chi = e^{-\mathrm{O}(\log \mathrm{N})}$, the same task as in Theorem 1 can be achieved in a gate complexity of*

$$\tilde{\mathrm{O}}\left(\frac{\Lambda}{\chi\Delta^{3/2}} + \frac{\Phi}{\chi\sqrt{\Delta}}\right) \tag{3}$$

and the same number (2) of qubits.

Provided Φ is not too large (which can be assumed in most practical scenarios), our algorithm for the case of unknown ground energy also has a better runtime scaling than naive phase estimation and uses significantly fewer qubits than an adaption of Poulin and Wocjan's filtering method[9] for this task [see Table II(a)].

Furthermore, for very small Δ, we show that alternatively, the scaling in Δ can be improved to $\sim 1/\Delta$ at the expense of worsening the scaling in χ by combining our algorithm with a prior run of phase estimation to first obtain an estimate of the ground energy.

Theorem 3 (Combined algorithm for ground state preparation). *If $\chi = e^{-\mathrm{O}(\log \mathrm{N})}$, the same task as in Theorem 2 can be achieved in a gate complexity of*

$$\tilde{\mathrm{O}}\left(\frac{\Lambda}{\chi^3\Delta^{\kappa}} + \frac{\Lambda}{\chi\Delta^{(3-\kappa)/2}} + \frac{\Phi}{\chi\Delta^{(1-\kappa)/2}}\right) \tag{4}$$

for any choice of $\kappa \in [0, 1]$, and the same number (2) of qubits.

In particular, choosing $\kappa = 1$ in Theorem 3 yields the optimal scaling in Δ of $\sim 1/\Delta$.

We show moreover that, with high probability, the algorithms for the case of unknown ground energy also find the ground energy to a precision of $\tilde{\mathrm{O}}(\Delta)$. Since Δ can be any reliable lower bound on the spectral gap, this yields a general algorithm for estimating the energy.

Theorem 4 (High-precision ground energy estimation). *Let $\xi = \tilde{\mathrm{O}}(\Delta)$. If $\chi = e^{-\mathrm{O}(\log \mathrm{N})}$, then we can find a real number E such that $|\mathrm{E} - \lambda_0| < \xi$ with constant probability in a gate complexity of*

$$\tilde{\mathrm{O}}\left(\frac{\Lambda}{\chi\xi^{3/2}} + \frac{\Phi}{\chi\sqrt{\xi}}\right), \tag{5}$$

TABLE I. Algorithms for ground state preparation for the case when the ground energy is known beforehand to the required precision.

Preparation (ground energy known)	Gates	Qubits	Required precision
This paper	$\tilde{\mathrm{O}}\left(\frac{\Lambda}{\lvert\phi_0\rvert\Delta} + \frac{\Phi}{\lvert\phi_0\rvert}\right)$	$\mathrm{O}\left(\log \mathrm{N} + \log\log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$	$\tilde{\mathrm{O}}(\Delta)$
Phase estimation + amp. amplif.	$\tilde{\mathrm{O}}\left(\frac{\Lambda}{\lvert\phi_0\rvert^2\Delta\epsilon} + \frac{\Phi}{\lvert\phi_0\rvert}\right)$	$\mathrm{O}\left(\log \mathrm{N} + \log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$	$\mathrm{O}(\lvert\phi_0\rvert\epsilon\Delta)$
Filtering (Poulin and Wocjan)	$\tilde{\mathrm{O}}\left(\frac{\Lambda}{\lvert\phi_0\rvert\Delta} + \frac{\Phi}{\lvert\phi_0\rvert}\right)$	$\mathrm{O}\left(\log \mathrm{N} + \log\frac{1}{\epsilon} + \frac{\log\frac{1}{\chi\epsilon}}{\log\log\frac{1}{\chi\epsilon}} \times \log\frac{1}{\Delta}\right)$	$\tilde{\mathrm{O}}(\Delta)$

TABLE II. Algorithms in case the ground energy is not known beforehand. (a) Algorithms for ground state preparation. (b) Algorithms for estimating the ground energy to a precision of $\xi \ll \Delta$. The minimum label finding algorithm is a subroutine that we describe in Sec. IV A. The combined approaches have been adjusted to yield the optimal scaling in Δ and ξ, respectively.

		Gates	Qubits
(a)	**Preparation (ground energy unknown)**		
	This paper	$\tilde{O}\left(\frac{\Lambda}{\chi\Delta^{3/2}} + \frac{\Phi}{\chi\sqrt{\Delta}}\right)$	$O\left(\log N + \log\log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$
	Phase estimation + min. label finding	$\tilde{O}\left(\frac{\Lambda}{\chi^4\Delta\epsilon} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$
	Filtering + min. label finding	$\tilde{O}\left(\frac{\Lambda}{\chi\Delta^{3/2}} + \frac{\Phi}{\chi\sqrt{\Delta}}\right)$	$O\left(\log N + \log\frac{1}{\epsilon} + \frac{\log\frac{1}{\chi\epsilon}}{\log\log\frac{1}{\chi\epsilon}} \times \log\frac{1}{\Delta}\right)$
	Combined approaches		
	This paper + phase estimation	$\tilde{O}\left(\frac{\Lambda}{\chi^3\Delta} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \log\frac{1}{\epsilon} + \log\frac{1}{\Delta}\right)$
	Filtering + phase estimation	$\tilde{O}\left(\frac{\Lambda}{\chi^3\Delta} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \log\frac{1}{\epsilon} + \frac{\log\frac{1}{\chi\epsilon}}{\log\log\frac{1}{\chi\epsilon}} \times \log\frac{1}{\Delta}\right)$
(b)	**Ground energy estimation**		
	This paper	$\tilde{O}\left(\frac{\Lambda}{\chi\xi^{3/2}} + \frac{\Phi}{\chi\sqrt{\xi}}\right)$	$O\left(\log N + \log\frac{1}{\xi}\right)$
	Phase estimation + min. label finding	$\tilde{O}\left(\frac{\Lambda}{\chi^3\xi} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \log\frac{1}{\xi}\right)$
	Filtering + min. label finding	$\tilde{O}\left(\frac{\Lambda}{\chi\xi^{3/2}} + \frac{\Phi}{\chi\sqrt{\xi}}\right)$	$O\left(\log N + \frac{\log\frac{1}{\chi}}{\log\log\frac{1}{\chi}} \times \log\frac{1}{\xi}\right)$
	Combined approaches		
	This paper + phase estimation	$\tilde{O}\left(\frac{\Lambda}{\chi^3\xi} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \log\frac{1}{\xi}\right)$
	Filtering + phase estimation	$\tilde{O}\left(\frac{\Lambda}{\chi^3\xi} + \frac{\Phi}{\chi}\right)$	$O\left(\log N + \frac{\log\frac{1}{\chi}}{\log\log\frac{1}{\chi}} \times \log\frac{1}{\xi}\right)$

using $O(\log N + \log \xi^{-1})$ *qubits. Alternatively, the combined approach of Theorem 3 achieves this task in a gate complexity of*

$$\tilde{O}\left(\frac{\Lambda}{\chi^3\xi^\kappa} + \frac{\Lambda}{\chi\xi^{(3-\kappa)/2}} + \frac{\Phi}{\chi\xi^{(1-\kappa)/2}}\right) \tag{6}$$

and the same number of qubits.

Provided that Φ is not too large, this also scales better than performing the same task with phase estimation and amplitude amplification [see Table II(b)].

In terms of query complexities, the coefficients of Φ and Λ in Tables I and II are, up to polylogarithmic factors, the number of calls to $\mathcal{C}_\phi$ and (unit time) Hamiltonian simulation, respectively. Note that the qubit requirements stated in Tables I and II exclude ancilla qubits required to perform Hamiltonian simulation.[22]

Our algorithms are inspired by classical power iteration methods.[23] They are based on techniques (termed the "Fourier method") that were recently developed for the quantum linear systems problem[19] and are based on the observation that the *Linear Combination of Unitaries*, or *LCU Lemma*,[4] can be used to implement other functions of a Hamiltonian.[24]

We now briefly outline the basic idea of the algorithms. It is easy to see that for positive-semidefinite H with non-degenerate lowest eigenvalue 0, high powers of $\cos H$ approximately project any given state into a state proportional to the unique ground state of H. In case the ground energy of $\tilde{H}$ is known, $\tilde{H}$ can be easily transformed into another Hamiltonian H such that $|\lambda_0\rangle$ is the unique eigenvector of H of eigenvalue ≈ 0. The outline of the algorithm in that case is as follows:

1. Approximate $\cos^M H$ as a linear combination of terms of the form e^{-iHt_k}.
2. Using the techniques in Ref. 19, we implement this linear combination with some amplitude using Hamiltonian simulation and the LCU lemma.
3. We use amplitude amplification to boost the overlap with the target state. Alternatively, the fixed point search algorithm[25] can be used for this step.

The outline of the algorithm in case the ground energy is unknown beforehand is as follows:

1. Adapt the previous algorithm into a circuit controlled on an ancilla register $|E\rangle$, which runs steps 1 and 2 of the previous algorithm, assuming that the ground energy were E.
2. Divide $[0, 1]$ into $L = \tilde{O}(\Delta^{-1})$ equally spaced values $E_0, \ldots, E_{L-1}$. Run the algorithm with $\sqrt{L^{-1}}\sum|E_j\rangle$ on the ancilla register.
3. Use the minimum label finding algorithm (Sec. IV A) to search for the smallest j such the residual state of $|E_j\rangle$ has large norm.
4. When this search succeeds, then with high probability the resulting E_j is within ξ of the true ground energy and the residual state is a good approximation of the ground state.

The outline of the combined approaches is as follows:

1. Use phase estimation and amplitude amplification to obtain a "crude" estimate of the ground energy. This provides an interval I which is known to contain the real ground energy.
2. Take $L \approx |I|/\xi$ equally spaced values $E_0, E_1, \ldots, E_{L-1}$ in I, and run the previous algorithm with these values of E_j.

III. ALGORITHM FOR THE CASE OF KNOWN GROUND ENERGY

In this section, we present the main technical analysis of our algorithm and prove Theorem 1. The method presented here is based on the observation that if the ground energy of $H \geq 0$ is close to 0, then high powers of $\cos H$ are approximately proportional to projectors onto the ground state:

Lemma 1. Let $E \in [0, \lambda_0]$ and $\tau \in [0, 1/2]$ be arbitrary, and let $H := \tilde{H} - (E - \tau)\mathbb{1}$. Then

$$\left\| \frac{\cos^M H|\phi\rangle}{\|\cos^M H|\phi\rangle\|} - |\lambda_0\rangle \right\| = O(\epsilon), \tag{7}$$

provided that

$$M = \Omega\left(\frac{1}{\Delta(\tau+\delta_E)}\log\frac{1}{|\phi_0|\epsilon}\right), \tag{8}$$

where $\delta_E := \lambda_0 - E$.

Proof. We have

$$\cos^M H|\phi\rangle = \phi_0 \cos^M(\tau+\delta_E)\left(|\lambda_0\rangle + \frac{1}{\phi_0}\frac{\cos^M H}{\cos^M(\tau+\delta_E)}|\lambda_0^\perp\rangle\right). \tag{9}$$

The norm of the second term is bounded by $|\phi_0|^{-1}e^{-\Omega(M(\tau+\delta_E)\Delta)}$. Indeed, since $\cos x$ is concave and decreasing on $[\tau, 1+\tau]$,

$$\left\| \frac{\cos^M H}{\cos^M(\tau+\delta_E)}|\lambda_0^\perp\rangle \right\| < \left(\frac{\cos(\tau+\delta_E) - \sin(\tau+\delta_E)\Delta}{\cos(\tau+\delta_E)}\right)^M, \tag{10}$$

$$= (1 - \tan(\tau+\delta_E)\Delta)^M, \tag{11}$$

$$= e^{-\Omega(M\tan(\tau+\delta_E)\Delta)} = e^{-\Omega(M(\tau+\delta_E)\Delta)}, \tag{12}$$

where in the last step we used $\tan x \geq x$ for $x \in [0, 1+\tau]$. Thus, Eq. (8) implies Eq. (7). □

Note that δ_E is not required to be small in Lemma 1.

Proof of Theorem 1. Suppose that the value of λ_0 is known to a precision of $\delta = O\big(\Delta/\log\frac{1}{\chi\epsilon}\big)$ with certainty, and let E be a known value such that $0 \le E \le \lambda_0$ and $\delta_E := \lambda_0 - E < \delta$. Define $H := \tilde{H} - (E - \tau)\mathbb{1}$ for some small value of τ chosen below. Then, $|\lambda_0\rangle$ is the (unique) ground state of H with eigenvalue $\tau + \delta_E$, and by assumption all other eigenvalues of H are $\ge \tau + \delta_E + \Delta$. Using Lemma 1,

$$\left\| \frac{\cos^M H|\phi\rangle}{\|\cos^M H|\phi\rangle\|} - |\lambda_0\rangle \right\| = O(\epsilon), \tag{13}$$

provided that

$$M = \Omega\bigg(\frac{1}{\Delta(\tau+\delta_E)}\log\frac{1}{|\phi_0|\epsilon}\bigg). \tag{14}$$

On the other hand, using that $\cos x > 1 - x^2/2$,

$$\cos^M(\tau+\delta_E) > \left(1 - \frac{(\tau+\delta_E)^2}{2}\right)^M \tag{15}$$

$$= e^{-O((\tau+\delta_E)^2 M)}. \tag{16}$$

Thus,

$$\|\cos^M H|\phi\rangle\| = \Omega(|\phi_0|), \tag{17}$$

provided that $\tau + \delta_E = O(1/\sqrt{M})$. Hence, since by assumption $\delta_E < \delta$, choosing

$$\tau = \Theta\left(\frac{\Delta}{\log\frac{1}{\chi\varepsilon}}\right) \tag{18}$$

and

$$M = \Theta\left(\frac{1}{\Delta^2}\log^2\frac{1}{\chi\epsilon}\right) \tag{19}$$

satisfies both (13) and (17).

Our aim in the following is to prepare $\cos^M H|\phi\rangle$. The strategy we employ is as follows: First, we approximate $\cos^M H$ as a linear combination of few unitaries of the form e^{-iHt_k} for suitable values of t_k. Second, we implement this linear combination with some amplitude using Hamiltonian simulation and the LCU lemma.[4] Third, we use amplitude amplification or fixed point search to boost the overlap with the target state.

In the following, assume for simplicity that $M = 2m$ is even (the algorithm can be adapted to odd M with minor modifications). Observe that

$$\cos^{2m} x = \left(\frac{e^{ix}+e^{-ix}}{2}\right)^{2m} = 2^{-2m}\sum_{k=-m}^{m}\binom{2m}{m+k}e^{2ikx}. \tag{20}$$

Note that

$$2^{-2m}\sum_{k=m_0+1}^{m}\binom{2m}{m+k} \le e^{-m_0^2/4m}. \tag{21}$$

Indeed, the LHS is the probability of seeing more than $m + m_0$ heads when flipping $2m$ coins, and (21) follows from the Chernoff bound. Thus,

$$\cos^{2m} H = \sum_{k=-m_0}^{m_0}\alpha_k e^{-2iHk} + O(\chi\epsilon), \tag{22}$$

where

$$\alpha_k := 2^{-2m}\binom{2m}{m+k} \tag{23}$$

and

$$m_0 = \Theta\left(\sqrt{M\log\frac{1}{\chi\epsilon}}\right) = \Theta\left(\frac{1}{\Delta}\log^{3/2}\frac{1}{\chi\epsilon}\right). \tag{24}$$

Next, e^{-2iHk} can be implemented using Hamiltonian simulation algorithms. To implement the RHS of (22), we employ the LCU lemma: let B be a circuit on $b := \lceil \log_2(2m_0 + 1) \rceil$ qubits that maps $|0\rangle^{\otimes b}$ to

$$\mathrm{B}|0\rangle^{\otimes b} := \frac{1}{\sqrt{\alpha}} \sum_{k=-m_0}^{m_0} \sqrt{\alpha_k}|k\rangle, \tag{25}$$

where $\alpha = \sum_{k=-m_0}^{m_0} \alpha_k$ and let U be the controlled Hamiltonian simulation $\mathrm{U}|k\rangle|\phi\rangle = |k\rangle e^{-2i\mathrm{H}k}|\phi\rangle$. Then,

$$(\mathrm{B}^\dagger \otimes \mathbb{1})\mathrm{U}(\mathrm{B} \otimes \mathbb{1})|\phi\rangle = \frac{1}{\alpha}|0\rangle^{\otimes b} \sum_{k=-m_0}^{m_0} \alpha_k e^{-2i\mathrm{H}k}|\phi\rangle + |\mathrm{R}\rangle, \tag{26}$$

where $(|0\rangle\langle 0|^{\otimes b} \otimes \mathbb{1})|\mathrm{R}\rangle = 0$.

The final step of the algorithm is to boost the overlap with amplitude amplification or fixed point search. Measuring the ancillas will then project the state onto

$$|\lambda_0'\rangle := \frac{\sum_{k=-m_0}^{m_0} \alpha_k e^{-2i\mathrm{H}k}|\phi\rangle}{\| \sum_{k=-m_0}^{m_0} \alpha_k e^{-2i\mathrm{H}k}|\phi\rangle \|} \tag{27}$$

with probability close to 1. From (22),

$$\frac{\sum_{k=-m_0}^{m_0} \alpha_k e^{-2i\mathrm{H}k}|\phi\rangle}{\| \sum_{k=-m_0}^{m_0} \alpha_k e^{-2i\mathrm{H}k}|\phi\rangle \|} = \frac{\cos^{2m}\mathrm{H}|\phi\rangle}{\| \cos^{2m}\mathrm{H}|\phi\rangle \|} + \mathrm{O}(\epsilon). \tag{28}$$

Thus, (13) implies

$$|\lambda_0'\rangle = |\lambda_0\rangle + \mathrm{O}(\epsilon), \tag{29}$$

as required. Eq. (17) implies that the number of repetitions is $\mathrm{O}\left(\alpha/\| \sum_{k=-m_0}^{m_0} \alpha_k e^{-2i\mathrm{H}k}|\phi\rangle \|\right) = \mathrm{O}(\alpha/|\phi_0|)$.

We now calculate the gate count of the entire algorithm. First note that B can be implemented with $\mathrm{O}(2^b) = \mathrm{O}(m_0)$ elementary gates.[26] Next, the gate cost to implement $e^{\pm 2i\mathrm{H}}$ to accuracy ϵ' in operator norm is $\mathrm{O}(\Lambda\,\mathrm{polylog}(\mathrm{N}, \frac{1}{\epsilon'}))$, depending on the precise model and Hamiltonian simulation algorithm used (see Table 1 of Ref. 6 for an overview). Here, we require $\epsilon' = \mathrm{O}(\epsilon|\phi_0|/m_0)$. Thus, the gate cost of U is $\mathrm{O}(m_0\Lambda\,\mathrm{polylog}(\mathrm{N}, \frac{m_0}{\epsilon|\phi_0|}))$ (Lemma 8 of Ref. 19). Note that Hamiltonian simulation with respect to H can be trivially obtained from Hamiltonian simulation with respect to $\tilde{\mathrm{H}}$, either by a phase shift or absorbing these phases into the values of the α_k. Finally, note that $\alpha = \mathrm{O}(1)$, and each iteration of amplitude amplification or fixed point search requires O(1) uses of $\mathcal{C}_\phi$, B, and U. The final gate complexity is thus

$$\mathrm{O}\left(\frac{1}{|\phi_0|}\left(m_0\Lambda\,\mathrm{polylog}\left(\mathrm{N}, \frac{m_0}{\epsilon|\phi_0|}\right) + \Phi\right)\right) = \mathrm{O}\left(\frac{\Lambda}{|\phi_0|\Delta}\,\mathrm{polylog}\left(\mathrm{N}, \frac{1}{\Delta}, \frac{1}{|\phi_0|\epsilon}\right) + \frac{\Phi}{|\phi_0|}\right). \tag{30}$$

Note that if fixed point search is used for the final step, we also require a good lower bound of $|\phi_0|$. However, we can simply run the algorithm with $\mathrm{O}(1/\chi')$ repetitions in fixed point search for $\chi' = 1, \frac{1}{2}, \frac{1}{4}, \ldots$ until we successfully project the ancillas into $|0\rangle^{\otimes(b+q)}$. Indeed, this only results in an overall multiplicative overhead of $\mathrm{O}(\log 1/|\phi_0|)$, yielding (1). Moreover, whenever we succeed, the resulting state is $|\lambda_0'\rangle$ independently of the value of χ' that was used. The number of ancilla qubits required is b plus any ancillas necessary for performing Hamiltonian simulation. Note that the number of ancilla qubits required for implementing U is essentially the same as the number of ancilla qubits required for a single run of $e^{\pm 2i\tilde{\mathrm{H}}}$, since the latter can be re-used. Thus, the total number of qubits required is given by (2), plus any ancilla qubits required for Hamiltonian simulation (see also Table 1 of Ref. 6). This proves Theorem 1. □

IV. ALGORITHM FOR THE CASE OF UNKNOWN GROUND ENERGY

The algorithm of Theorem 1 presented in Sec. III requires an estimate E of λ_0 to a precision of $\mathrm{O}(1/\sqrt{\mathrm{M}})$. Since E can simply be viewed as an input parameter, the algorithm can in principle also be run with different values of E. It is easy to see that for any $\mathrm{E} \in [0, \lambda_0]$, the algorithm would, if successful, produce a good approximation of $|\lambda_0\rangle$ but may have an exponentially small probability of success. Thus, if the ground energy is not known beforehand, one could simply run the algorithm for increasing values of E, using a step size $\mathrm{O}(1/\sqrt{\mathrm{M}})$ and stop when first successful. It is moreover easy to see that the value of E at which we first succeed is with high probability a good estimate of λ_0. Clearly, the runtime of this algorithm would result in an overall factor of $\mathrm{O}(\sqrt{\mathrm{M}}) = \tilde{\mathrm{O}}(1/\Delta)$ compared to Eq. (1).

It turns out that this "classical search" for the correct value of E can be replaced by a "quantum search" that quadratically improves the overhead from $O(\sqrt{M})$ to $\tilde{O}(\sqrt[4]{M})$. We call this search the *minimum label finding* algorithm, which we describe in Sec. IV A as a general subroutine, and which may be of independent interest. We then apply this algorithm to the ground state preparation and ground energy estimation problem in Sec. IV B.

A. Minimum label finding

In this section, we describe a general subroutine to find the minimum "label" in an ancilla register amongst terms with at least some given amplitude in a given superposition. To motivate this result, consider the following scenario. Suppose we have L unitaries $U_0, \dots, U_{L-1}$ that prepare the states $U_i|0\rangle|0\rangle = |0\rangle|\Phi_i\rangle + |R_i\rangle$, where $(|0\rangle\langle 0| \otimes \mathbb{1})|R_i\rangle = 0$. Let $\chi \in (0, 1)$ be a known real number. Suppose we want to approximately find the smallest i (or alternatively, prepare the corresponding $|\Phi_i\rangle$) such that $\| |\Phi_i\rangle \| \geq \chi$. The naive way to do this is to use amplitude estimation (or fixed point search) to increase the amplitude of $|0\rangle$ on the first register of $U_i|0\rangle|0\rangle$ for $i = 0, 1, \dots$, each time using only $O(1/\chi)$ repetitions of U_i, until we first succeed. This requires $\tilde{O}(L/\chi)$ calls to the individual U_i's in total. Below, we show that a quadratic improvement in L to $\tilde{O}(\sqrt{L}/\chi)$ can be achieved, provided that performing the controlled version $U = \sum_i |i\rangle\langle i| \otimes U_i$ can be done with essentially the same cost as the individual U_i's. The algorithm is based on a simple binary search technique to successively find the binary digits of the smallest "label" i amongst the terms in the superposition $\sqrt{L^{-1}}|0\rangle \sum_i |i\rangle|\Phi_i\rangle + |R\rangle$ with a norm of at least $\chi/\sqrt{L}$.

Proposition 1 (Minimum label finding). *Let $\mathcal{C}_\Phi$ be a circuit on $q + n + m$ qubits that prepares the state*

$$|\Phi\rangle := \mathcal{C}_\Phi|0\rangle^{\otimes(q+n+m)} = |0\rangle^{\otimes q} \sum_{i=0}^{2^n-1} |i\rangle|\Phi_i\rangle + |R\rangle, \tag{31}$$

where $|\Phi_i\rangle$ are non-normalised m-qubit states and $|R\rangle$ has no overlap with $|0\rangle^{\otimes q}$ on the first q qubits. Let $\zeta \in (0, 1)$ and $\delta \in (0, 1/4)$ be known real numbers, and let $\tilde{J}, J \in \{0, \dots, 2^n - 1\}$ be (unknown) integers such that $\| |\Phi_J\rangle \| \geq \zeta$ and

$$\sum_{i=0}^{\tilde{J}} \| |\Phi_i\rangle \|^2 < \zeta^2 \frac{\delta}{8n \log_2 \frac{n}{\delta} \ln^2\left(\frac{4n}{\delta} \log_2 \frac{n}{\delta}\right)}. \tag{32}$$

Then, there exists a quantum algorithm on $q + n + m + 1$ qubits which, with probability at least $1 - 4\delta$, prepares the state $|j\rangle|\Phi_j\rangle/\| |\Phi_j\rangle \|$ for some $j \in [\tilde{J}, J]$, and which uses

$$O\left(\frac{n}{\zeta} \log^2 \frac{n}{\delta}\right) \tag{33}$$

calls to $\mathcal{C}_\Phi$ and

$$O\left(\frac{\mathrm{poly}(q, n, m)}{\zeta} \log^2 \frac{n}{\delta}\right) \tag{34}$$

additional elementary gates.

Note that at the end of the algorithm, the register containing j is in a computational basis state and that the state $|\Phi_j\rangle/\| |\Phi_j\rangle \|$ is prepared exactly.

The minimum label finding algorithm uses the fixed-point search algorithm,[25] which can be thought of as a variation of amplitude amplification, with the additional features that it is not possible to "overshoot" the target state, and that only a lower bound on the initial overlap is required to be known. More precisely, let $\mathcal{C}$ be a circuit on n' qubits that prepares the state $\mathcal{C}|0\rangle^{\otimes n'} = \lambda|T\rangle + \sqrt{1 - \lambda^2}|\bar{T}\rangle$ with $\langle T|\bar{T}\rangle = 0$, and U be a unitary that satisfies $U|T\rangle|b\rangle = |T\rangle|1 - b\rangle$ and $U|\bar{T}\rangle|b\rangle = |\bar{T}\rangle|b\rangle$ for $b \in \{0, 1\}$. Then, given input parameters $\lambda', \delta \in (0, 1)$, the fixed point search $\mathrm{FPS}(\mathcal{C}, U, \lambda', \delta)$ is a circuit on $n' + 1$ qubits using $O(\log(1/\delta)/\lambda')$ calls to $\mathcal{C}, \mathcal{C}^\dagger, U$ and $O(n'^2 \log(1/\delta)/\lambda')$ elementary gates such that the following hold:[27]

Lemma 2. (i) *If $\lambda \geq \lambda'$, then $|\langle T, 0|\mathrm{FPS}(\mathcal{C}, U, \lambda', \delta)|0\rangle^{\otimes(n'+1)}|^2 \geq 1 - \delta^2$.*
(ii) *If $\lambda \leq \lambda'$, then $|\langle T, 0|\mathrm{FPS}(\mathcal{C}, U, \lambda', \delta)|0\rangle^{\otimes(n'+1)}| \leq 2\frac{\lambda}{\lambda'} \ln \frac{2}{\delta}$.*

Proof. Part (i) is the central result proven in Ref. 25. To prove part (ii), let $t := \lceil \frac{1}{\lambda'} \ln \frac{2}{\delta} \rceil$ be the number of calls to $\mathcal{C}$. From Eq. (1) of Ref. 25, the success probability P can be expressed in terms of (generalised) Chebyshev polynomials $\mathcal{T}_t(x)$ of the first kind,

$$P := |\langle T, 0|\mathrm{FPS}(\mathcal{C}, U, \lambda', \delta)|0\rangle^{\otimes(n'+1)}|^2 = 1 - \delta^2 \mathcal{T}_t\left(\mathcal{T}_{1/t}(1/\delta)\sqrt{1 - \lambda^2}\right)^2. \tag{35}$$

Without loss of generality, assume that $t \geq 5$, so $t(t+1) \leq 2\left(\frac{1}{\lambda'}\ln\frac{2}{\delta}\right)^2$. It thus suffices to prove that $P \leq 2t(t+1)\lambda^2$. Write $\lambda = \sin\theta$ and $\tau := \mathcal{T}_{1/t}(1/\delta) = \cosh(t^{-1}\operatorname{arccosh}(1/\delta)) \geq 1$. Note that $\mathcal{T}_t(\tau) = 1/\delta$. Then, using the mean value theorem on the function $\mathcal{T}_t(x)^2$, we obtain

$$P = 1 - \frac{\mathcal{T}_t(\tau\cos\theta)^2}{\mathcal{T}_t(\tau)^2}, \tag{36}$$

$$= \tau(1-\cos\theta)\frac{2\mathcal{T}_t(\xi)\mathcal{T}'_t(\xi)}{\mathcal{T}_t(\tau)^2}, \tag{37}$$

$$= t\tau(1-\cos\theta)\frac{2\mathcal{T}_t(\xi)\mathcal{U}_{t-1}(\xi)}{\mathcal{T}_t(\tau)^2} \tag{38}$$

for some $\xi \in [\tau\cos\theta, \tau]$, where we used that $\mathcal{T}'_t(x) = t\mathcal{U}_{t-1}(x)$ and $\mathcal{U}_t(x)$ are the (generalised) Chebyshev polynomials of the second kind. Note that since $\tau \geq 1$ and $\xi \leq \tau$, $|\mathcal{T}_t(\xi)| \leq \mathcal{T}_t(\tau)$ and $|\mathcal{U}_{t-1}(\xi)| \leq \mathcal{U}_{t-1}(\tau)$. Indeed, both $\mathcal{T}_t(x)$ and $\mathcal{U}_{t-1}(x)$ attain their maximum moduli on $[-1, 1]$ at $x = 1$ and are monotonically increasing on $[1, \infty)$. Finally, using the relations

$$\mathcal{U}_t(\tau) = \begin{cases} 1 + 2\sum_{k=1}^{t/2}\mathcal{T}_{2k}(\tau), & t \text{ even}, \\ \\ 2\sum_{k=0}^{(t-1)/2}\mathcal{T}_{2k+1}(\tau), & t \text{ odd} \end{cases} \tag{39}$$

and $0 < \mathcal{T}_{k'}(\tau) \leq \mathcal{T}_k(\tau)$ for $k' \leq k$ and $\tau \geq 1$, we obtain

$$2\tau\mathcal{U}_{t-1}(\tau) = \mathcal{U}_t(\tau) + \mathcal{U}_{t-2}(\tau) \leq 2\mathcal{U}_t(\tau) \leq 2(t+1)\mathcal{T}_t(\tau). \tag{40}$$

Hence,

$$P \leq 2t(t+1)(1-\cos\theta) \leq 2t(t+1)\sin^2\theta = 2t(t+1)\lambda^2, \tag{41}$$

as claimed. □

Proof of Proposition 1. The algorithm proceeds by successively attempting to find the binary digits $a_1, \ldots, a_n \in \{0, 1\}$ of an integer such that $2^{n-1}a_1 + \cdots + a_n \in [\tilde{J}, J]$. The algorithm runs in two stages. Broadly, the first stage of the algorithm tries to find a sufficient number of leading digits $a_1, \ldots, a_k$, while the second stage then attempts to prepare the state $|j\rangle|\Phi_j\rangle$ for some $j \in [\tilde{J}, J]$. Concretely, the algorithm runs as follows: in the first stage, suppose that $a_1, \ldots, a_{k-1}$ have already been found. To obtain a_k, we use fixed point search[25] to search for $|0\rangle^{\otimes q}|a_1 \ldots a_{k-1}0\rangle$ on the first $q + k$ qubits, using at most $O(\frac{1}{\zeta}\log\frac{1}{\delta'})$ repetitions of $\mathcal{C}_\Phi$. We repeat this search K times. We choose $\delta' = \delta/(2n\log_2(n/\delta))$ and $K = \lceil\log_2(n/\delta)\rceil$. If it succeeds all K times, we set $a_k = 0$ and move on to the next digit a_{k+1} (unless $k = n$, in which case the algorithm terminates). Otherwise, we search K times for $|0\rangle^{\otimes q}|a_1 \ldots a_{k-1}1\rangle$ on the first $q + k$ qubits. If we succeed all K times, we set $a_k = 1$ and move on to the next digit a_{k+1} (unless $k = n$, in which case the algorithm terminates). If we fail at least once, we say that the result is "inconclusive," and we abort the first stage of the algorithm and move on. In the second stage of the algorithm, we successively search for $|0\rangle^{\otimes q}|a_1 \ldots a_{k-1}\rangle$, $|0\rangle^{\otimes q}|a_1 \ldots a_{k-2}\rangle$, etc., where each search is repeated K times, using $O(\frac{1}{\zeta}\log\frac{1}{\delta'})$ calls to $\mathcal{C}_\Phi$, until we succeed at least once. Once successful, we measure the remaining ancillas and the algorithm terminates.

We now show that this algorithm produces the required results. Let $J = 2^{n-1}b_1 + 2^{n-2}b_2 + \cdots + b_n$ and $\tilde{J} = 2^{n-1}c_1 + 2^{n-2}c_2 + \cdots + c_n$ be the binary representations of $J, \tilde{J}$, respectively. Let i_0 be the maximum integer such that $b_l = c_l$ for all $l = 1, \ldots, i_0$. Since $\tilde{J} < J$, it follows that $i_0 < n$ and $b_{i_0+1} = 1, c_{i_0+1} = 0$. Moreover, we say that binary digits $a_1, \ldots, a_k$ are "consistent" if they are the dominating k binary digits of at least one integer in $[\tilde{J}, J]$.

Let $a_1, \ldots, a_k$ be the digits that the first stage of the algorithm finds, and let $i_1 \leq k$ be the largest integer such that $a_l = b_l$ for $l = 1, \ldots, i_1$. We first show that with probability at least $1 - 2\delta$, we find $k \geq i_0 + 1$ consistent digits (implying $i_1 \geq i_0$) and either $k > i_1$ or $k = n$. For a given k', the probability of finding $a_{k'}$ such that $a_1, \ldots, a_{k'}$ are not consistent, given that $a_1, \ldots, a_{k'-1}$ are consistent, is at most δ/n. Indeed, finding too large a value means that $a_l = b_l$ for $l = 1, \ldots, k' - 1$, $b_{k'} = 0$ and finding $a_{k'} = 1$. The probability of this happening is bounded by the probability of the algorithm failing to find $|0\rangle^{\otimes q}|b_1 \ldots b_{k'-1}0\rangle$ at least once out of the K trials, which is at most $\delta'^2 K < \delta/n$.[25] On the other hand, from Lemma 2(ii), using $\lambda' = \zeta$, the probability of finding too small a value for a single trial of the search is upper bounded by

$$\frac{4\ln^2\frac{2}{\delta'}}{\zeta^2}\sum_{j=0}^{\tilde{J}}\| |\Phi_j\rangle \|^2 < \delta. \tag{42}$$

Thus, the probability of finding too small a value is at most $\delta^K < \delta/n$. Note that since $a_1, \ldots, a_{k'-1}$ are assumed to be consistent, this can only happen if $b_{k'} = 1$.

Moreover, if $a_l = b_l$ for $l = 1, \ldots, k' - 1$, the probability of finding an inconclusive result for $a_{k'}$ is at most $\delta'^2 K < \delta/n$. Thus, with probability at least

$$1 - n(\max(\delta'^2 K, \delta^K) + \delta'^2 K) \geq 1 - 2\delta, \tag{43}$$

the first stage finds consistent digits $a_1, \ldots, a_k$ with the following property: either the algorithm that never finds an inconclusive result (i.e., $k = n$) or the algorithm that finds $a_l = b_l$ for $l = 1, \ldots, i_1$ for some $i_1 \geq i_0$, does not find an inconclusive result on a_{i_1+1} (which implies $k > i_1$) and later finds an inconclusive result at a_{k+1}. Indeed, in the LHS of Eq. (43), the two terms inside the max correspond to finding a wrong digit leading to an inconclusive result, and the last term corresponds to finding an inconclusive result at $k \leq i_0$ or $k = i_1 + 1$.

In the following, we bound the probability of the algorithm failing in the second stage. This can happen in two ways. First, fixed point search could successfully find $|0\rangle^{\otimes q}|a_1 \ldots a_l\rangle$ for some $l > i_1$, but, upon measuring the remaining ancillas, we get a value of $j \notin [\tilde{J}, J]$. Second, fixed point search could fail to find $|0\rangle^{\otimes q}|a_1 \ldots a_{i_1+1}\rangle$.

We now address the first way of the second stage failing. Suppose that in the first stage, we found $k \geq i_0 + 1$ consistent digits and then found an inconclusive result. We can thus assume $k > i_1 \geq i_0$. Clearly, $a_{i_1+1} = 0$ and $b_{i_1+1} = 1$, since $a_1, \ldots, a_k$ are consistent. Thus, for all $l > i_1$, if the algorithm successfully finds $|0\rangle^{\otimes q}|a_1 \ldots a_l\rangle$, then upon measuring the other ancillas, we obtain a value $j < J$.

We now show that for a single trial, the probability of first successfully finding $|0\rangle^{\otimes q}|a_1 \ldots a_l\rangle$ and then finding $j < \tilde{J}$ upon measuring the remaining $n - l$ ancillas is at most δ/nK. Let A_l be the event that a single trial of fixed point search successfully finds $|0\rangle^{\otimes q}|a_1 \ldots a_l\rangle$, and let B_l be the event that measuring the remaining ancillas then yields $j < \tilde{J}$. Write $\tilde{a}_l := 2^{n-1}a_1 + 2^{n-2}a_{n-2} + \cdots + 2^{n-l}a_l$ and

$$\lambda_l^2 := \sum_{j=\tilde{a}_l}^{\tilde{a}_l + 2^{n-l}-1} \| |\Phi_j\rangle \|^2. \tag{44}$$

Suppose first that $\lambda_l \leq \zeta$. Then, using Lemma 2(ii) with $\lambda' = \zeta$, we have $\mathbb{P}(A_l) \leq 4\frac{\lambda_l^2}{\zeta^2}\ln^2\frac{2}{\delta'}$, and

$$\mathbb{P}(B_l|A_l) = \frac{1}{\lambda_l^2}\sum_{j=\tilde{a}_l}^{\tilde{J}-1} \| |\Phi_j\rangle \|^2. \tag{45}$$

Hence,

$$\mathbb{P}(A_l \cap B_l) \leq \frac{4\ln^2\frac{2}{\delta'}}{\zeta^2}\sum_{j=\tilde{a}_l}^{\tilde{J}-1} \| |\Phi_j\rangle \|^2 < \frac{\delta}{nK}, \tag{46}$$

as claimed. On the other hand, if $\lambda_l > \zeta$, then

$$\mathbb{P}(A_l \cap B_l) \leq \mathbb{P}(B_l|A_l) < \frac{1}{\zeta^2}\sum_{j=\tilde{a}_l}^{\tilde{J}-1} \| |\Phi_j\rangle \|^2 < \frac{\delta}{nK}, \tag{47}$$

as claimed. Since there are at most nK points in the algorithm where the event $A_l \cap B_l$ can occur (K times for each value of l), the probability of finding $j < \tilde{J}$ in the second stage for some $l > i_1$ is at most δ.

The last possibility for the algorithm to fail is if the algorithm fails to find $|a_1 \ldots a_{i_1+1}\rangle$ in the second stage. Note that this is only possible if the algorithm does the following: in the first stage, we find $a_1, \ldots, a_k$ for some $k > i_1$, and then in the second stage, we fail to find $|0\rangle^{\otimes q}|a_1 \ldots a_k\rangle$ on K trials, then fail to find $|0\rangle^{\otimes q}|a_1 \ldots a_{k-1}\rangle$ on K trials, etc., until ultimately we also fail to find $|0\rangle^{\otimes q}|a_1 \ldots a_{i_1+1}\rangle$ on K trials. Call this event C_{i_1}. Let p_{i_1} be the probability of a single run of the search finding $|0\rangle^{\otimes q}|a_1 \ldots a_{i_1+1}\rangle$. Note that C_{i_1} includes both successfully finding $a_1, \ldots, a_{i_1+1}$ K times and failing to find $a_1, \ldots, a_{i_1+1}$ K times. Thus,

$$\mathbb{P}(C_{i_1}) \leq p_{i_1}^K (1 - p_{i_1})^K \leq 4^{-K}, \tag{48}$$

where we used that $p(1-p) \leq 1/4$ for $p \in [0,1]$. Thus, the probability of the algorithm failing in the second stage is at most

$$\delta + \sum_{i_1=i_0}^{n-1} \mathbb{P}(C_{i_1}) \leq \delta + n4^{-K} \leq 2\delta, \tag{49}$$

where the first δ comes from the upper bound on the probability of finding $j < \tilde{J}$ in the second stage upon measuring the remaining $n - l$ ancillas after successfully finding $a_1, \ldots, a_l$ for some $l > i_1$. Thus, the total probability of the algorithm failing is at most (43) plus (49), as claimed. The number of calls to $\mathcal{C}_\Phi$ and additional elementary gates is

$$O\left(nK\frac{1}{\zeta}\log\frac{1}{\delta'}\right) = O\left(\frac{n}{\zeta}\log^2\frac{n}{\delta}\right), \tag{50}$$

as claimed. □

Notice that if only the first $n' < n$ dominant binary digits of an integer in $[\tilde{J}, J]$ are needed, n can be replaced with n' in (32) and (33). Moreover, it is clear that the ζ^2 dependence in (32) is optimal.

B. Proof of Theorems 2–4

With M and τ defined as in Sec. III, let $L = \Theta(\sqrt{M})$ be a power of 2 and define $E_j := j/L$. Clearly, $E_{j+1} - E_j \in O(1/\sqrt{M})$. Define $\delta_j := \lambda_0 - E_j$ and $H_j := \tilde{H} - (E_j - \tau)\mathbb{1}$. Let $J \in \{0, \ldots, L-1\}$ be the largest integer such that $E_J \leq \lambda_0$. Clearly, $\delta_J \in O(1/\sqrt{M})$. Using the construction from Sec. III, we have unitaries V_j such that

$$V_j|0\rangle^{\otimes(b+q)}|\phi\rangle = |0\rangle^{\otimes(b+q)}\mathcal{G}_j|\phi\rangle + |R\rangle, \tag{51}$$

where

$$\mathcal{G}_j = \frac{1}{\alpha}\sum_{k=-m_0}^{m_0}\alpha_k e^{-2iH_jk} \tag{52}$$

and $(|0\rangle\langle 0|^{\otimes(b+q)} \otimes \mathbb{1})|R\rangle = 0$. The gate cost of a single V_j is given by

$$\tilde{O}\left(\frac{\Lambda}{\Delta}\right). \tag{53}$$

We now introduce an additional ancilla system $\mathcal{L}$ on $l := \lceil \log_2 L \rceil$ qubits. Let V be the V_j's controlled on $\mathcal{L}$, i.e., $V = \sum_{j=0}^{L-1}|j\rangle\langle j|_{\mathcal{L}} \otimes V_j$. Then,

$$V|+\rangle_{\mathcal{L}}^{\otimes l}|0\rangle^{\otimes(b+q)}|\phi\rangle = \frac{1}{\sqrt{L}}\sum_{j=0}^{L-1}|j\rangle_{\mathcal{L}}|0\rangle^{\otimes(b+q)}\mathcal{G}_j|\phi\rangle + |R\rangle, \tag{54}$$

where $(\mathbb{1}_{\mathcal{L}} \otimes |0\rangle\langle 0|^{\otimes(b+q)} \otimes \mathbb{1})|R\rangle = 0$. From Lemma 1 and (22),

$$\left\| \frac{\mathcal{G}_j|\phi\rangle}{\|\mathcal{G}_j|\phi\rangle\|} - |\lambda_0\rangle \right\| = O(\epsilon), \tag{55}$$

whenever $E_j \leq \lambda_0$, and $\|\mathcal{G}_j|\phi\rangle\| = \Theta\left(|\phi_0|\cos(\tau + \delta_j)^M\right)$. In particular, $\|\mathcal{G}_J|\phi\rangle\| = \Omega(|\phi_0|)$.

We now show that we can use the minimum label finding algorithm to project (54) onto the state $|j\rangle_{\mathcal{L}}\mathcal{G}_j|\phi\rangle/\|\mathcal{G}_j|\phi\rangle\|$ for some $j \in [\tilde{J}, J]$ and suitable $\tilde{J}$. Indeed, for any integer $\tilde{J}$,

$$\begin{aligned}
\sum_{j=0}^{\tilde{J}}\|\mathcal{G}_j|\phi\rangle\|^2 &= O\left(\sum_{j=0}^{\tilde{J}}|\phi_0|^2\cos(\tau+\delta_j)^{2M}\right) && (56)\\
&= O\left(|\phi_0|^2\sum_{j=0}^{\tilde{J}}e^{-(\tau+\delta_j)^2M}\right) && (57)\\
&= O\left(|\phi_0|^2\sum_{j=0}^{\tilde{J}}e^{-2\delta_j\tau M}\right) && (58)\\
&= O\left(|\phi_0|^2\sum_{j=0}^{\tilde{J}}e^{-2(\lambda_0-j/L)\tau M}\right) && (59)\\
&= O\left(|\phi_0|^2e^{-2(\lambda_0-\tilde{J}/L)\tau M}\right) && (60)\\
&= O\left(|\phi_0|^2e^{-2\delta_{\tilde{J}}\tau M}\right) && (61)\\
&= O\left(|\phi_0|^2e^{-\Theta(\delta_{\tilde{J}}\sqrt{M})}\right), && (62)
\end{aligned}$$

where the last step follows from (18). Thus, since χ is a known lower bound on $|\phi_0|$ satisfying $\log(|\phi_0|/\chi) = \mathrm{O}(\log \mathrm{N})$, we can ensure that

$$\sum_{j=0}^{\tilde{J}} \|\mathcal{G}_j|\phi\rangle\|^2 = \mathrm{O}\left(\frac{\chi^2\epsilon}{L\log L\log\frac{\log L}{\epsilon}\log^2\left(\frac{\log L}{\epsilon}\log\frac{\log L}{\epsilon}\right)}\right), \tag{63}$$

provided that

$$\delta_{\tilde{J}} = \Theta\left(\frac{1}{\sqrt{M}}\log\left(\frac{|\phi_0|^2}{\chi^2\epsilon}L\log L\log\frac{\log L}{\epsilon}\log^2\left(\frac{\log L}{\epsilon}\log\frac{\log L}{\epsilon}\right)\right)\right) = \tilde{\Theta}(\Delta). \tag{64}$$

To prove Theorem 2, observe that using Proposition 1 with $\zeta = \chi/\sqrt{L}$, $n = \log L$, and $\delta = \epsilon/4$, we can, with probability at least $1-\epsilon$, prepare the state $|j\rangle_{\mathcal{L}}\mathcal{G}_j|\phi\rangle/\|\mathcal{G}_j|\phi\rangle\|$ for some $j \in [\tilde{J}, J]$ using $\tilde{\mathrm{O}}(\sqrt{L}/\chi)$ repetitions of V and $\mathcal{C}_\phi$. The value in the $\mathcal{L}$ register gives an estimate of the ground energy to a precision of (64). Since this process succeeds with probability at least $1-\epsilon$, (55) implies that the second register then contains a (mixed) state that is $\mathrm{O}(\epsilon)$-close to $|\lambda_0\rangle\langle\lambda_0|$.

Notice that V can be implemented with essentially the same cost as a single V_j. Indeed, the only explicit dependence of V_j on j is in the call to Hamiltonian simulation, $e^{\pm 2iH_j} = e^{\pm 2i(\tau - E_j)}e^{\pm 2i\tilde{H}}$. Thus, given access to $e^{\pm 2i\tilde{H}}$, it is easy to implement

$$\sum_{j=0}^{L-1}|j\rangle\langle j| \otimes e^{\pm 2iH_j} = \sum_{j=0}^{L-1}|j\rangle\langle j| \otimes e^{\pm 2i(\tau - E_j)}e^{\pm 2i\tilde{H}} \tag{65}$$

with a single call to $e^{\pm 2i\tilde{H}}$ and $\mathrm{O}(\log L) = \tilde{\mathrm{O}}(1)$ additional elementary gates implementing the phases $e^{\pm 2i(\tau - E_j)}$. Hence, the overall gate cost of this algorithm is

$$\tilde{\mathrm{O}}\left(\frac{\sqrt{L}}{\chi}\left(\frac{\Lambda}{\Delta} + \Phi\right)\right) = \tilde{\mathrm{O}}\left(\frac{\Lambda}{\chi\Delta^{3/2}} + \frac{\Phi}{\chi\sqrt{\Delta}}\right), \tag{66}$$

as claimed. This proves Theorem 2.

To prove Theorem 3, one can alternatively combine the algorithm of Theorem 2 with the prior use of phase estimation. This approach improves the scaling with Δ at the cost of a worse scaling in χ and is hence useful if Δ is very small.

First we use Proposition 2 from Appendix A with $\delta = \epsilon$ to obtain a "crude" estimate of the ground energy, to a precision of $\xi = \mathrm{O}(\Delta^\kappa)$ for some $\kappa \in [0, 1]$ to be chosen later. The runtime of this is

$$\tilde{\mathrm{O}}\left(\frac{\Lambda}{\chi^3\Delta^\kappa} + \frac{\Phi}{\chi}\right) \tag{67}$$

gates (see Appendix A). With probability at least $1-\epsilon$, this provides us with an interval $[a, b] \ni \lambda_0$ with $b - a = \mathrm{O}(\Delta^\kappa)$. Let $E'_j = a + (b-a)j/L'$, where $L' = \Theta(\sqrt{M}\Delta^\kappa)$. Writing $H'_j = \tilde{H} - (E'_j - \tau)\mathbb{1}$ and

$$\mathcal{G}'_j = \frac{1}{\alpha}\sum_{k=-m_0}^{m_0}\alpha_k e^{-2iH'_j k}, \tag{68}$$

we run the previous algorithm but with

$$V'|+\rangle_{\mathcal{L}'}^{\otimes l'}|0\rangle^{\otimes(b+q)}|\phi\rangle = \frac{1}{\sqrt{L'}}\sum_{j=0}^{L'-1}|j\rangle_{\mathcal{L}'}|0\rangle^{\otimes(b+q)}\mathcal{G}'_j|\phi\rangle + |R\rangle \tag{69}$$

instead of (54), and an ancilla system $\mathcal{L}'$ on $l' = \lceil\log_2 L'\rceil$ qubits instead of $\mathcal{L}$. Since the use of Proposition 2 succeeds with probability at least $1-\epsilon$, an additional error of only $\mathrm{O}(\epsilon)$ is added to the final (mixed) state.

The algorithm now requires $\tilde{\mathrm{O}}(\sqrt{L'}/\chi)$ repetitions of V' and $\mathcal{C}_\phi$. As before, the cost of V' is given by (53). Hence, the number of gates for projecting (69) onto the target state is

$$\tilde{\mathrm{O}}\left(\frac{\sqrt{L'}}{\chi}\left(\frac{\Lambda}{\Delta} + \Phi\right)\right) = \tilde{\mathrm{O}}\left(\frac{\Lambda}{\chi\Delta^{(3-\kappa)/2}} + \frac{\Phi}{\chi\Delta^{(1-\kappa)/2}}\right). \tag{70}$$

The total number of gates for the algorithm is thus (70) plus the gate cost (67) from the prior use of phase estimation, i.e.,

$$\tilde{O}\left(\frac{\Lambda}{\chi^3\Delta^\kappa}+\frac{\Lambda}{\chi\Delta^{(3-\kappa)/2}}+\frac{\Phi}{\chi\Delta^{(1-\kappa)/2}}\right). \tag{71}$$

This proves Theorem 3.

Finally, to prove Theorem 4, it is easy to see that both algorithms can be used to estimate the ground energy to an arbitrarily small precision ξ. Indeed, note that both algorithms, with constant probability, find λ_0 to a precision of $\delta_{\bar{j}}$ given by Eq. (64), and the latter depends on the input parameter Δ. However, for the algorithms to function, Δ is only required to be a lower bound on the spectral gap rather than the spectral gap itself. Hence, if we run the algorithm with $\Delta' \ll \Delta$ instead of Δ as the input parameter, we obtain an estimate of the ground state to a precision of $\delta(\Delta')$, where $\delta(\Delta')$ is given by Eq. (64), but with Δ replaced by Δ'. The algorithm can thus be used to estimate the ground energy to an arbitrary precision ξ in a gate complexity of

$$\tilde{O}\left(\frac{\Lambda}{\chi\xi'^{3/2}}+\frac{\Phi}{\chi\sqrt{\xi'}}\right) \tag{72}$$

for the first algorithm and

$$\tilde{O}\left(\frac{\Lambda}{\chi^3\xi'^\kappa}+\frac{\Lambda}{\chi\xi'^{(3-\kappa)/2}}+\frac{\Phi}{\chi\xi'^{(1-\kappa)/2}}\right) \tag{73}$$

for the combined approach, where $\xi' = \min(\xi, \delta_{\bar{j}})$. This proves Theorem 4. □

Note that choosing $\kappa = 1$ in the combined algorithm gives the optimal scaling of $\sim 1/\Delta$ with Δ. However, it is obviously also possible to choose different values of κ that also take the other parameters into account.

V. CONCLUSION

In this paper, we presented quantum algorithms to prepare the ground state of a Hamiltonian faster and with fewer qubits than with previous methods, both in the case of known and unknown ground energy. In the latter scenario, the algorithm also provides a high precision estimate of the ground energy in a complexity faster than with phase estimation and amplitude amplification.

Perhaps surprisingly, the straightforward use of phase estimation and amplitude amplification has a significantly worse scaling in the overlap of the trial state with the ground state than what one would naively expect. In Appendix B, we show that straightforward phase estimation requires

$$\tilde{O}\left(\frac{\Lambda}{|\phi_0|^2\Delta\epsilon}+\frac{\Phi}{|\phi_0|}\right) \tag{74}$$

gates to prepare the ground state, provided the ground energy is known to a precision of $O(|\phi_0|\epsilon\Delta)$ beforehand. Notice in particular the $1/|\phi_0|^2$ scaling, even after using amplitude amplification. The scaling becomes even worse if the ground energy is not known beforehand [see Table II(a)].

Previous improvements by Poulin and Wocjan[9] to the phase estimation approach quadratically improved the dependence on $|\phi_0|$. Moreover, we prove in Appendix C that for suitable choices of parameters, this algorithm can prepare the ground state in the same runtime as our algorithm. The algorithms in this paper however use significantly fewer qubits, which make them more attractive for early applications of quantum computing.

In the case of known ground energy, no non-trivial prior knowledge of the value of $|\phi_0|$ is required for any algorithm discussed in this paper (although for the filtering method, the number of qubits required becomes worse if no non-trivial lower bound is known). However, in the case of unknown ground energy, the runtime dependences on $|\phi_0|$ are replaced by dependences on the known lower bound χ of $|\phi_0|$. There appears to be a systematic reason for this which seems difficult to overcome: broadly speaking, all methods discussed in this paper produce a state of the form $\sum_j |E_j\rangle|\Phi_j\rangle$, where $|\Phi_j\rangle$ is approximately proportional to the ground state and $\||\Phi_j\rangle\| \approx |\phi_0|$ if $E_j \approx \lambda_0$, and $\||\Phi_j\rangle\| \ll |\phi_0|$ if $E_j \ll \lambda_0$. Suppose that $\lambda_0^{(1)}, \lambda_0^{(2)}, \phi_0^{(1)}, \phi_0^{(2)}$ are real numbers in [0, 1] satisfying $\lambda_0^{(1)} \ll \lambda_0^{(2)}$ and $\phi_0^{(1)} \ll \phi_0^{(2)}$. Any search for the "correct" value of j needs to be able to distinguish the following two cases: (i) $\lambda_0 = \lambda_0^{(1)}$ and $\phi_0 = \phi_0^{(1)}$, and (ii) $\lambda_0 = \lambda_0^{(2)}$ and $\phi_0 = \phi_0^{(2)}$. Any use of amplitude amplification to distinguish these cases would require many, i.e., $O\left(1/\phi_0^{(1)}\right)$ repetitions to "see" if (i) is the case, which means that if in fact (ii) is the case, significantly more repetitions than $O\left(1/\phi_0^{(2)}\right)$ have been performed. On the other hand, if only $O\left(1/\phi_0^{(2)}\right)$ repetitions are performed, one would accidentally find a much larger value (e.g., $\lambda_0^{(2)}$) than the true ground energy $\lambda_0^{(1)}$ (and hence also prepare the wrong state) if in fact (i) is the case. We currently do not see a way to get around this problem.

For simplicity of notation, we assumed throughout this paper that the ground energy of $\tilde{H}$ is non-degenerate. The algorithm however trivially generalises to degenerate ground spaces. Indeed, the only thing that changes in this case is that the resulting state $|\lambda_0'\rangle$ is ϵ-close to $|\lambda_0\rangle = P|\phi\rangle/\|P|\phi\rangle\|$, where P is the projector onto the ground space of $\tilde{H}$. The algorithm also generalises to the case of only approximately degenerate ground spaces, i.e., when the spectrum of $\tilde{H}$ is contained in $[\lambda_0, \lambda_0 + \varepsilon] \cup [\lambda_0 + \Delta, 1]$ with $\varepsilon \ll \Delta$.

Finally, given the close relation of this work to Ref. 19, it is also natural to expect that the alternative approach of Ref. 19 using Chebyshev polynomials and quantum walks can be used to get a different algorithm with the same runtime scaling. This is indeed possible, as shown in Appendix D. These algorithms however are restricted to the case where $\tilde{H}$ is a sparse Hamiltonian with quantum oracle access. By contrast, the advantage of the approach based on Hamiltonian simulation is that it can be applied outside the framework of given oracle access to $\tilde{H}$, as efficient Hamiltonian simulation algorithms exist for other models of accessing the Hamiltonian.[3,6]

ACKNOWLEDGMENTS

Y.G. thanks H. Buhrman, V. Dunjko, A. Gilyén, D. Gosset, R. Kothari, G. Low, A. Molnár, A. Montanaro, and N. Schuch for helpful discussions. This work was supported by the ERC grant QUENOCOBA, ERC-2016-ADG (Grant No. 742102). J.T. acknowledges funding from the EU's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Grant Agreement No. 748549.

APPENDIX A: FINDING THE GROUND ENERGY USING PHASE ESTIMATION

We now demonstrate how to combine phase estimation and the minimum label finding algorithm to find the unknown ground energy of a quantum Hamiltonian to a precision of ξ in a gate complexity of

$$\tilde{O}\left(\frac{\Lambda}{\chi^3 \xi} + \frac{\Phi}{\chi}\right) \tag{A1}$$

with probability at least $1 - \delta$, where $\delta \in (0, 1)$ is a real number and we assume throughout this section that polylogarithmic factors in δ^{-1} are suppressed by the $\tilde{O}$ notation. Notice that straightforward amplitude amplification cannot be used to amplify the ground energy since the latter is not known.

Let U be an $N \times N$ unitary with eigenvectors $|\lambda_i\rangle$ and eigenvalues $e^{2\pi i \lambda_i}$ with $\lambda_i \in [0, 1)$. We are interested in finding a good approximation of the minimum value λ_0, say, to $n = \lceil \log_2 1/\xi \rceil$ binary digits of precision.

Let $|\phi\rangle = \sum_i \phi_i |\lambda_i\rangle$ be an arbitrary trial state. Suppose that we are given a circuit $\mathcal{C}_\phi$ which prepares $|\phi\rangle$ using Φ elementary gates. Recall that the phase estimation algorithm[8] takes $|\phi\rangle|0\rangle^{\otimes k}$ for some $k > n$ to

$$|\Phi\rangle := \sum_i \phi_i |\lambda_i\rangle|\tilde{\varphi}_i\rangle \tag{A2}$$

using controlled $U, U^2, \ldots, U^{2^{k-1}}$ and an inverse quantum Fourier transform on k qubits, where $|\tilde{\varphi}_i\rangle$ can be shown to have a large overlap with the computational basis state that encodes the first n binary digits of λ_i.

Let

$$|\tilde{\varphi}_i\rangle = \sum_{x=0}^{2^k-1} \gamma_{ix}|x\rangle \tag{A3}$$

be its expansion in the computational basis. Then,

$$|\Phi\rangle = \sum_{x=0}^{2^k-1} |\Phi_x\rangle|x\rangle, \tag{A4}$$

where

$$|\Phi_x\rangle = \sum_i \phi_i \gamma_{ix}|\lambda_i\rangle \tag{A5}$$

are non-normalised states.

Naively, one would expect that simply measuring the ancillas would give a good approximation $x \approx 2^k \lambda_0$ of the ground energy with probability $\approx |\phi_0|^2$ so that $O(1/|\phi_0|^2)$ repetitions would be needed to find the ground energy. One would also expect that this could be quadratically reduced to $O(1/|\phi_0|)$ using suitable amplitude amplification techniques. However, we show below that the overall gate cost is much higher if the ground energy is not known beforehand. Indeed, due to the finite precision and error in the phase estimation algorithm, $\||\Phi_x\rangle\|^2$ could in principle be large even when $x \ll 2^k\lambda_0$. This means that the algorithm could fail by accidentally finding a value that is much smaller than the true ground energy. We show below that in order to guarantee that this does not happen, one needs to choose $2^k = \tilde{O}(2^n/|\phi_0|^2)$. Since the runtime of phase estimation scales linearly with 2^k, and only a lower bound χ on $|\phi_0|$ is assumed to be known, this would lead to an overall runtime that scales as $\sim 1/\chi^3$. We also demonstrate at the end of this section that this dependence is essentially tight.

Proposition 2 (Ground energy estimation with phase estimation). *Let $\delta \in (0, 1)$ be a real number. Then, in a gate complexity of (A1) and using $O(\log N + \log 1/\xi)$ qubits, phase estimation can be used to find a real number $E > 0$ such that with probability at least $1 - \delta$, $|E - \lambda_0| < \xi$.*

Proof. Recall first the well-known[29] relations

$$\gamma_{ix} = \frac{1}{2^k}\left(\frac{1 - e^{2\pi i(2^k\lambda_i - x)}}{1 - e^{2\pi i(\lambda_i - x/2^k)}}\right) \tag{A6}$$

and

$$|\gamma_{ix}| \le \frac{1}{2|2^k\lambda_i - x|}. \tag{A7}$$

Let $D := 2^{k-n}$. Then, using (A7), we have

$$\sum_{x<2^k\lambda_0 - D} \||\Phi_x\rangle\|^2 = \sum_{\substack{i\\ x<2^k\lambda_0 - D}} |\phi_i|^2|\gamma_{ix}|^2 \tag{A8}$$

$$\le \sum_{\substack{i\\ x<2^k\lambda_0 - D}} |\phi_i|^2\left|\frac{1}{2|2^k\lambda_i - x|}\right|^2 \tag{A9}$$

$$\le \sum_{\substack{i\\ x<2^k\lambda_0 - D}} |\phi_i|^2\left|\frac{1}{2|2^k\lambda_0 - x|}\right|^2 \tag{A10}$$

$$\le \sum_{x<2^k\lambda_0 - D}\left|\frac{1}{2|2^k\lambda_0 - x|}\right|^2 \tag{A11}$$

$$< \sum_{y>D}\frac{1}{y^2} < \frac{1}{D}. \tag{A12}$$

Moreover, when x satisfies $|\lambda_i - x/2^k| < 1/2^{k+1}$,

$$|\gamma_{ix}| \ge \frac{2}{\pi}, \tag{A13}$$

using that

$$|\theta| \ge |1 - e^{i\theta}| \ge 2|\theta|/\pi \tag{A14}$$

for $\theta \in [-\pi, \pi]$. Thus,

$$\left\||\Phi_{\lfloor 2^k\lambda_0\rceil}\rangle\right\| \ge \frac{2}{\pi}|\phi_0|. \tag{A15}$$

Let χ be a known lower bound on $|\phi_0|$. Using Proposition 1, we can thus find an integer $x \in [\lfloor 2^k\lambda_0\rceil - D, \lfloor 2^k\lambda_0\rceil]$ with probability at least $1 - \delta$, provided that $1/D = \tilde{O}(\chi^2)$. The number of calls to phase estimation is $\tilde{O}(1/\chi)$. The number of gates required for each run of phase estimation is $\tilde{O}(2^k\Lambda)$. Thus, taking $U = e^{-2\pi i\tilde{H}}$ and $\xi = 2^{-n}$, we arrive at a total gate count of (A1). Moreover, the number of qubits required is $O(\log N + k) = O(\log N + \log 1/\xi)$. □

It is not difficult to show that for suitable $\tilde{H}$ and $|\phi\rangle$, the dependence of this algorithm on χ is optimal. Indeed, suppose that $\tilde{H} = \frac{1}{2^k}(H' + c\mathbb{1})$, where $c \in (0, 1/2)$ is a constant and H' is any Hamiltonian with integer spectrum in $\{2^{k-2}, \ldots, 2^{k-1} - 1\}$. Then from (A14),

$$|\gamma_{ix}| \ge \frac{2c}{\pi|2^k\lambda_i - x|}. \tag{A16}$$

Suppose that our trial state $|\phi\rangle$ is such that $|\phi_1|^2 \ge 1/2$. Then,

$$\begin{aligned}\sum_{x<2^k\lambda_0-D} \| |\Phi_x\rangle\|^2 &= \sum_{\substack{i \\ x<2^k\lambda_0-D}} |\phi_i|^2|\gamma_{ix}|^2 && \text{(A17)}\\ &\geq \frac{c}{\pi}\sum_{x<2^k\lambda_0-D}\frac{1}{|2^k\lambda_1 - x|^2} && \text{(A18)}\\ &= \Omega\left(\int_{D+2^k\Delta}^{2^k\lambda_1}\frac{1}{x^2}dx\right) && \text{(A19)}\\ &= \Omega\left(\frac{1}{2^k}\left(\frac{1}{\xi+\Delta}-\frac{1}{\lambda_1}\right)\right). && \text{(A20)}\end{aligned}$$

Suppose now that $\Delta = O(\xi)$ and $\xi = o(1)$. Then

$$\sum_{x<2^k\lambda_0-D} \| |\Phi_x\rangle\|^2 = \Omega\left(\frac{1}{2^k\xi}\right), \tag{A21}$$

which is much larger than χ^2 unless $2^k = \Omega\left(\frac{1}{\chi^2\xi}\right)$, and the claim follows.

APPENDIX B: PREPARING THE GROUND STATES USING PHASE ESTIMATION

Naively, one would expect that upon successfully projecting the ancillas of (A2) into the first n (or even k) digits of λ_0, the residual state should be a good approximation of $|\lambda_0\rangle$, provided that $\xi = \tilde{O}(\Delta)$. Unfortunately, this is not true because the "imperfections" in $|\tilde{\varphi}_i\rangle$ build up to a non-negligible error in the residual state. This phenomenon was also observed in Ref. 9. Below, we analyse these errors and show a non-negligible bound on the minimum precision to which the ground energy needs to be known beforehand.

Proposition 3 (Ground state preparation with phase estimation for known ground energy). *Suppose that the value of λ_0 is known to a precision of $O(|\phi_0|\epsilon\Delta)$, i.e., there exists a known real number $E > 0$ such that $|E - \lambda_0| = O(|\phi_0|\epsilon\Delta)$ with certainty. Then phase estimation can be used to prepare a state ϵ-close to the ground state with constant probability in a gate complexity of*

$$\tilde{O}\left(\frac{\Lambda}{|\phi_0|^2\Delta\epsilon}+\frac{\Phi}{|\phi_0|}\right) \tag{B1}$$

and using $O(\log N + \log 1/\epsilon + \log 1/\Delta)$ qubits.

Proof. Suppose that $z := \lfloor 2^k\lambda_0 \rceil$, i.e., z encodes the first k binary digits of λ_0 (k will be specified below) and we apply phase estimation on k qubits. Then, post-selecting the ancillas of (A2) to be in $|z\rangle$, the residual state is

$$|\lambda\rangle := \frac{\sum_i \phi_i\gamma_{iz}|\lambda_i\rangle}{\|\sum_i \phi_i\gamma_{iz}|\lambda_i\rangle\|}. \tag{B2}$$

Thus, the error in the residual state is

$$\| |\lambda\rangle - |\lambda_0\rangle\| = \Theta\left(\frac{\|\sum_{i\neq 0}\phi_i\gamma_{iz}|\lambda_i\rangle\|}{\|\sum_i \phi_i\gamma_{iz}|\lambda_i\rangle\|}\right). \tag{B3}$$

For $i \neq 0$, (A7) implies that

$$|\gamma_{iz}| \leq \frac{1}{2^{k+1}\Delta}. \tag{B4}$$

Moreover, (A13) implies $|\gamma_{0z}| \geq 2/\pi$. Hence,

$$\begin{aligned}\left(\frac{\|\sum_{i\neq 0}\phi_i\gamma_{iz}|\lambda_i\rangle\|}{\|\sum_i \phi_i\gamma_{iz}|\lambda_i\rangle\|}\right)^2 &= \frac{\sum_{i\neq 0}|\phi_i|^2|\gamma_{iz}|^2}{\sum_i |\phi_i|^2|\gamma_{iz}|^2} && \text{(B5)}\\ &\leq \left(\frac{1}{\pi|\phi_0|2^k\Delta}\right)^2. && \text{(B6)}\end{aligned}$$

Thus, it is sufficient if k satisfies

$$2^k = O\left(\frac{1}{|\phi_0|\epsilon\Delta}\right) \tag{B7}$$

for an error $\| |\lambda\rangle - |\lambda_0\rangle \| = O(\epsilon)$. Notice in particular the scaling with $|\phi_0|$.

Suppose that the value of z, i.e., the ground energy to a precision of k binary digits, is known, where k is known to satisfy (B7). The cost of a single run of phase estimation is $\tilde{O}(2^k\Lambda)$, up to polylogarithmic factors. Fixed point search requires $O(1/|\phi_0|)$ applications of phase estimation and $\mathcal{C}_\phi$. We thus arrive at a total gate count of

$$\tilde{O}\left(\frac{2^k\Lambda + \Phi}{|\phi_0|}\right) = \tilde{O}\left(\frac{\Lambda}{|\phi_0|^2\Delta\epsilon} + \frac{\Phi}{|\phi_0|}\right), \tag{B8}$$

as claimed. □

Suppose next that the ground energy is not known beforehand. If phase estimation and minimum label finding (Proposition 1) are first used to find the ground energy, we require a precision of $O(|\phi_0|\epsilon\Delta)$. Since $|\phi_0|$ is not assumed to be known, we need to run phase estimation to find the energy to a precision of $\xi = O(\chi\epsilon\Delta)$. Thus, from Proposition 2, the number of gates to first find the ground energy to the required precision takes

$$\tilde{O}\left(\frac{\Lambda}{\chi^4\Delta\epsilon} + \frac{\Phi}{\chi}\right) \tag{B9}$$

gates.

Corollary 1 (Ground state preparation with phase estimation for unknown ground energy). *If the ground energy is not known beforehand, phase estimation can be used to prepare a state ϵ-close to $|\lambda_0\rangle\langle\lambda_0|$ with constant probability in a gate complexity of*

$$\tilde{O}\left(\frac{\Lambda}{\chi^4\Delta\epsilon} + \frac{\Phi}{\chi}\right) \tag{B10}$$

and using $O(\log N + \log 1/\epsilon + \log 1/\Delta)$ qubits.

We now argue that the dependence of 2^k on $|\phi_0|$ in (B7), and thus the quadratic dependence on $|\phi_0|$ in (B1), is essentially tight for this algorithm. Suppose that

$$\frac{\sum_{i\neq 0}|\phi_i|^2|\gamma_{iz}|^2}{\sum_i |\phi_i|^2|\gamma_{iz}|^2} \le \epsilon^2. \tag{B11}$$

Then,

$$\begin{aligned}
\epsilon^2 &\ge \frac{\sum_{i\neq 0}|\phi_i|^2|\gamma_{iz}|^2}{\sum_i |\phi_i|^2|\gamma_{iz}|^2} && \text{(B12)}\\
&= \frac{\sum_{i\neq 0}|\phi_i|^2|\gamma_{iz}|^2}{|\phi_0|^2|\gamma_{0z}|^2 + \sum_{i\neq 0}|\phi_i|^2|\gamma_{iz}|^2} && \text{(B13)}\\
&\ge (1-\epsilon^2)\frac{\sum_{i\neq 0}|\phi_i|^2|\gamma_{iz}|^2}{|\phi_0|^2|\gamma_{0z}|^2} && \text{(B14)}\\
&\ge (1-\epsilon^2)\frac{1}{|\phi_0|^2}\sum_{i\neq 0}|\phi_i|^2|\gamma_{iz}|^2. && \text{(B15)}
\end{aligned}$$

Suppose that for all $i \neq 0$,

$$r(2^k\lambda_i - z) \ge c, \tag{B16}$$

where $r(x) := |x - \lfloor x \rceil| \in [0, 1/2]$, for some constant $c > 0$. Notice that this can be achieved for any Hamiltonian of the form $H = \frac{1}{2^k}(\tilde{H} + c\mathbb{1})$, where $\tilde{H}$ is any Hamiltonian with the integer spectrum. Then, using Eq. (A14),

$$|\gamma_{iz}| = \frac{1}{2^k}\left(\frac{1 - e^{2\pi i r(2^k\lambda_i - z)}}{1 - e^{2\pi i(\lambda_i - z/2^k)}}\right) \ge \frac{c}{2^{k-1}}. \tag{B17}$$

Thus,

$$\epsilon^2 \ge (1-\epsilon^2)\frac{c^2(1-|\phi_0|^2)}{4^{k-1}|\phi_0|^2}. \tag{B18}$$

Therefore, $2^k = \Omega\left(\frac{1}{|\phi_0|\epsilon}\right)$, and the claim follows.

One can also show that the linear dependence on $1/\Delta$ is tight: if our trial state is $|\phi\rangle = \phi_0|\lambda_0\rangle + \phi_1|\lambda_1\rangle$ with $\lambda_1 = \lambda_0 + \Delta$, then (B17) can be replaced by

$$|\gamma_{iz}| = \frac{1}{2^k}\left(\frac{1 - e^{2\pi i r(2^k\lambda_1 - z)}}{1 - e^{2\pi i(\lambda_1 - z/2^k)}}\right) \geq \frac{c}{\pi 2^{k-1}\Delta}, \tag{B19}$$

which follows from Eq. (A14). Thus, $2^k = \Omega\left(\frac{1}{|\phi_0|\epsilon\Delta}\right)$, and the claim follows.

APPENDIX C: FILTERING METHOD BY POULIN AND WOCJAN

Previously, Poulin and Wocjan proposed a filtering method[9] as an improvement to phase estimation that only has an inverse dependence on the overlap $|\phi_0|$. We briefly review this algorithm here and show that it can be adapted to achieve a runtime that scales only polylogarithmically in the allowed error (the analysis provided in Ref. 9 only yields a state with low expected energy, and only an error analysis for the expected energy rather than the residual state is given there). We also show that this method can be combined with the minimum label finding algorithm in case the ground energy is not known beforehand to first find the ground energy to the required precision. We remark that the filtering method can also be formulated as a majority voting scheme.[19,28]

The Poulin-Wocjan algorithm is based on the following idea: let $\mathcal{A}$ be the circuit of phase estimation with k ancilla qubits, but without the inverse Fourier transform. Then $\mathcal{A}|\lambda_i\rangle|0\rangle^{\otimes k} = |\lambda_i\rangle|\varphi_i\rangle$, where

$$|\varphi_i\rangle := \frac{1}{\sqrt{2^k}}\sum_{x=0}^{2^k-1} e^{2\pi i\lambda_i x}|x\rangle \tag{C1}$$

is a momentum state encoding of λ_i. Since $\mathcal{A}$ maps $|\lambda_i\rangle|0\rangle^{\otimes k}$ to $|\lambda_i\rangle|\varphi_i\rangle$, then for any state $|\mu\rangle$ on k qubits, $\mathcal{A}^\dagger$ maps $|\phi\rangle|\mu\rangle$ to

$$\sum_i \phi_i\langle\varphi_i\rangle\mu|\lambda_i\rangle|0\rangle^{\otimes k} + |R\rangle, \tag{C2}$$

where $|R\rangle$ has no overlap with $|0\rangle^{\otimes k}$ on the ancillas. Hence, starting with η copies of $|\mu\rangle$ on ηk ancillas (η will be specified later), applying η copies of $\mathcal{A}^\dagger$ that maps $|\phi\rangle|\mu\rangle^{\otimes\eta}$ to

$$\sum_i \phi_i\langle\varphi_i\rangle\mu^\eta|\lambda_i\rangle|0\rangle^{\otimes\eta k} + |R'\rangle, \tag{C3}$$

where $|R'\rangle$ has no overlap with $|0\rangle^{\otimes\eta k}$ on the ancillas. The central idea of Ref. 9 is that for suitable choices of η and $|\mu\rangle$, $|\langle\varphi_i|\mu\rangle|^\eta$ is a "filter function" that is centered around λ_0 and falls off quickly, thus suppressing all terms in (C3) except for the contribution from $|\lambda_0\rangle$. We now show that this idea can be used to obtain a ground state preparation algorithm whose runtime also only scales polylogarithmically with ϵ.

Proposition 4 (Ground state preparation with the filtering method for known ground energy). *Suppose that the value of λ_0 is known to a precision of*

$$O\left(\frac{\Delta}{\sqrt{\log^3\frac{1}{|\phi_0|\epsilon}\log\log\frac{1}{|\phi_0|\epsilon}}}\right) \tag{C4}$$

with certainty, i.e., there exists a known real number $E > 0$ such that $|E - \lambda_0|$ is at most (C4). Then, the filtering method can prepare a state ϵ-close to $|\lambda_0\rangle\langle\lambda_0|$ with constant probability in a gate complexity of

$$\tilde{O}\left(\frac{\Lambda}{|\phi_0|\Delta} + \frac{\Phi}{|\phi_0|}\right) \tag{C5}$$

and using

$$O\left(\log N + \log\frac{1}{\epsilon} + \frac{\log\frac{1}{\chi\epsilon}}{\log\log\frac{1}{\chi\epsilon}} \times \log\frac{1}{\Delta}\right) \tag{C6}$$

qubits.

Proof. Suppose that we know a value of $\mu \in [0, 1)$ such that

$$|\mu - \lambda_0| < \frac{1}{2^{k+1}\pi\sqrt{\eta}}, \tag{C7}$$

where k and η will be specified later. Note that this is the same as assuming that we know λ_0 up to $k + l$ binary digits, where $l := \lceil \log_2(2\pi\sqrt{\eta}) \rceil$, and we can assume, without loss of generality, that $2^{k+l}\mu \in \mathbb{Z}$. Choose

$$|\mu\rangle := \frac{1}{\sqrt{2^k}} \sum_{x=0}^{2^k-1} e^{2\pi i \mu x} |x\rangle. \tag{C8}$$

Note that $|\mu\rangle$ can be efficiently prepared from the computational basis state $|2^{k+l}\mu\rangle$ by first applying the quantum Fourier transform on $k + l$ qubits, then applying Hadamard gates on the last l qubits, and finally discarding the last l qubits. The circuit $\mathcal{C}_\mu$ preparing $|\mu\rangle$ from $|0\rangle^{\otimes(k+l)}$ thus only requires $O((k + l)^2)$ gates.[29] Similarly to (A7) and (B4), for $i \neq 0$,

$$|\langle \varphi_i |\mu\rangle| \leq \frac{1}{2^{k+1}|\lambda_i - \mu|} \leq \frac{1}{2^{k+1}\Delta}. \tag{C9}$$

Moreover, it can be shown[9] that (C7) implies

$$|\langle \varphi_0 |\mu\rangle|^\eta \geq \frac{1}{2}. \tag{C10}$$

Thus, using amplitude amplification or fixed point search to search for $|0\rangle^{\otimes \eta k}$ on (C3), we obtain with constant probability the state

$$|\sigma\rangle := \frac{\sum_i \phi_i \langle \varphi_i\rangle \mu^\eta |\lambda_i\rangle}{\|\sum_i \phi_i \langle \varphi_i\rangle \mu^\eta |\lambda_i\rangle\|}, \tag{C11}$$

with $O(\eta/|\phi_0|)$ uses of $\mathcal{A}^\dagger$ and $\mathcal{C}_\mu$, and $O(1/|\phi_0|)$ uses of $\mathcal{C}_\phi$.

We now show that $|\sigma\rangle$ is a good approximation of $|\lambda_0\rangle$ for appropriate choices of k and η. We have that

$$\frac{1}{2}\| |\sigma\rangle - |\lambda_0\rangle \|^2 \leq \frac{\| \sum_{i\neq 0} \phi_i \langle \varphi_i|\mu\rangle^\eta |\lambda_i\rangle \|^2}{\| \sum_i \phi_i \langle \varphi_i|\mu\rangle^\eta |\lambda_i\rangle \|^2} \tag{C12}$$

$$= \frac{\sum_{i\neq 0} |\phi_i|^2 |\langle \varphi_i|\mu\rangle|^{2\eta}}{\sum_i |\phi_i|^2 |\langle \varphi_i|\mu\rangle|^{2\eta}} \tag{C13}$$

$$\leq 4\left(\frac{1}{2^{k+1}\Delta}\right)^{2\eta} \frac{1}{|\phi_0|^2}. \tag{C14}$$

Thus, in order to obtain

$$\| |\sigma\rangle - |\lambda_0\rangle \| < \epsilon, \tag{C15}$$

it suffices if k and η satisfy

$$k = \left\lceil \log_2 \frac{1}{\Delta} \right\rceil + O\left(\log\log \frac{1}{|\phi_0|\epsilon}\right) \tag{C16}$$

and

$$\eta = O\left(\frac{\log \frac{1}{|\phi_0|\epsilon}}{\log\log \frac{1}{|\phi_0|\epsilon}}\right). \tag{C17}$$

But since η is a parameter of the algorithm that needs to be chosen beforehand, and $|\phi_0|$ is unknown, we need to choose

$$\eta = O\left(\frac{\log \frac{1}{\chi\epsilon}}{\log\log \frac{1}{\chi\epsilon}}\right) \tag{C18}$$

to ensure the algorithm works. The full algorithm thus requires

$$\tilde{O}\left(\frac{\eta 2^k \Lambda + \Phi}{|\phi_0|}\right) = \tilde{O}\left(\frac{\Lambda}{|\phi_0|\Delta} + \frac{\Phi}{|\phi_0|}\right) \tag{C19}$$

gates and

$$O(\log N) + \eta k = O\left(\log N + \log\frac{1}{\epsilon} + \frac{\log \frac{1}{\chi\epsilon}}{\log\log \frac{1}{\chi\epsilon}} \times \log\frac{1}{\Delta}\right) \tag{C20}$$

qubits. □

There does not appear to be an obvious way, such as a recycling scheme, to reduce the number of qubits required.

Suppose now that the value of μ is not known beforehand. We show now that in this case, one can combine the filtering method with the minimum label finding algorithm to determine a suitable value of μ beforehand.

Proposition 5 (Ground state preparation with the filtering method for unknown ground energy). *If the ground energy is not known beforehand, the same task as in Proposition 4 can be achieved in a gate complexity of*

$$\tilde{O}\left(\frac{\Lambda}{\chi\Delta^{3/2}} + \frac{\Phi}{\chi\sqrt{\Delta}}\right) \tag{C21}$$

and the same number (C6) *of qubits.*

Proof. Let k and η be defined as in the Proof of Proposition 4. Let $\mu_j = j/2^{k_1}$, where $k_1 \geq k$ will be chosen later. It is easy to prepare the state

$$\frac{1}{\sqrt{2^{k_1}}}\sum_{j=0}^{2^{k_1}-1} |j\rangle|\mu_j\rangle^{\otimes\eta_1}|\phi\rangle, \tag{C22}$$

where η_1 will be chosen later, and

$$|\mu_j\rangle = \frac{1}{\sqrt{2^{k_1}}}\sum_{x=0}^{2^k-1} e^{2\pi i\mu_j x}|x\rangle. \tag{C23}$$

We now run a controlled version of the filtering algorithm with $\eta_1 \times k_1$ ancilla qubits. This produces the state

$$\sum_{j=0}^{2^{k_1}-1} |j\rangle|\Phi_j\rangle, \tag{C24}$$

where

$$|\Phi_j\rangle = \frac{1}{\sqrt{2^{k_1}}}\sum_i \phi_i\langle\varphi_i|\mu_j\rangle^{\eta_1}|\lambda_i\rangle. \tag{C25}$$

Let J be the smallest integer such that $|\mu_J - \lambda_0| < \frac{1}{2^{k+2}\pi\sqrt{\eta}}$. Then from (C10), $\||\Phi_J\rangle\| \geq \frac{|\phi_0|}{2\sqrt{2^{k_1}}} \geq \frac{\chi}{2\sqrt{2^{k_1}}}$. Let $\tilde{J} < J$ be an integer such that

$$\sum_{j=0}^{\tilde{J}} \||\Phi_j\rangle\|^2 = O\left(\frac{\chi^2}{2^{k_1}}\right). \tag{C26}$$

Then, the minimum label finding algorithm with $\delta = \epsilon/4$ finds an integer $j \in [\tilde{J}, J]$ with probability at least $1 - \epsilon$. To obtain a value of j that gives rise to a good approximation of λ_0, we need to ensure that $\mu_J - \mu_{\tilde{J}} < \frac{1}{2^{k+2}\pi\sqrt{\eta}}$, since the latter then implies $|\mu_j - \lambda_0| < \frac{1}{2^{k+1}\pi\sqrt{\eta}}$ for all $j \in [\tilde{J}, J]$.

Let $D = J - \tilde{J}$. We have

$$\sum_{j=0}^{\tilde{J}} \||\Phi_j\rangle\|^2 = \frac{1}{2^{k_1}}\sum_{j=0}^{\tilde{J}}\sum_i |\phi_i|^2|\langle\varphi_i|\mu_j\rangle|^{2\eta_1} \tag{C27}$$

$$\leq \frac{1}{2^{k_1}}\sum_{j=0}^{\tilde{J}}\sum_i |\phi_i|^2\frac{1}{(2^{k_1+1}|\lambda_i - \mu_j|)^{2\eta_1}} \tag{C28}$$

$$< \frac{1}{2^{k_1}}\sum_{j=0}^{\tilde{J}}\frac{1}{(2^{k_1+1}|\lambda_0 - \mu_j|)^{2\eta_1}} \tag{C29}$$

$$< \frac{1}{2^{k_1}}\sum_{j>D}\frac{1}{j^{2\eta_1}} \tag{C30}$$

$$< \frac{1}{2^{k_1}}\frac{1}{D^{2\eta_1-1}}. \tag{C31}$$

We thus require

$$\frac{1}{D^{2\eta_1-1}} < \chi^2, \tag{C32}$$

with $D = O(2^{k_1}\xi_F)$, where $\xi_F = 1/(2^{k+1}\pi\sqrt{\eta})$ is the required precision (C7). Thus, it suffices to choose

$$2^{k_1} = O\left(\frac{1}{\xi_F}\log\frac{1}{\chi}\right) = O\left(\frac{1}{\Delta}\log^{3/2}\frac{1}{\chi\epsilon}\sqrt{\log\log\frac{1}{\chi\epsilon}}\log\frac{1}{\chi}\right) \tag{C33}$$

and

$$\eta_1 = O\left(\frac{\log\frac{1}{\chi}}{\log\log\frac{1}{\chi}}\right). \tag{C34}$$

This will, with probability at least $1-\epsilon$, provide an estimate of λ_0 that can be used for the state preparation. The probability of not finding such an estimate of λ_0 only induces an error of $O(\epsilon)$ in the final (mixed) state. The number of gates for this estimation is

$$\tilde{O}\left(\frac{\sqrt{2^{k_1}}}{\chi}\left(2^{k_1}\Lambda + \Phi\right)\right) = \tilde{O}\left(\frac{\Lambda}{\chi\Delta^{3/2}} + \frac{\Phi}{\chi\Delta^{1/2}}\right), \tag{C35}$$

as claimed. □

Note that in analogy to Sec. IV B, the algorithm can be used to estimate the ground energy to an arbitrary precision $\xi = \tilde{O}(\Delta)$, by simply running the algorithm with a smaller value of Δ.

Corollary 2 (Ground energy estimation with the filtering method). *Let $\xi = \tilde{O}(\Delta)$. Then, the filtering method can be used to find a real number $E > 0$ such that $|E - \lambda_0| < \xi$ with constant probability in a gate complexity of*

$$\tilde{O}\left(\frac{\Lambda}{\chi\xi^{3/2}} + \frac{\Phi}{\chi\sqrt{\xi}}\right) \tag{C36}$$

and using

$$O\left(\log N + \frac{\log\frac{1}{\chi}}{\log\log\frac{1}{\chi}} \times \log\frac{1}{\xi}\right) \tag{C37}$$

qubits.

Alternatively, one can also use a combined approach by using a prior run of phase estimation to first get a "crude" estimate of λ_0, similarly to the method in Sec. IV of the main text. This approach is useful if Δ is very small.

Proposition 6 (Combining the filtering method with phase estimation). *By combining the filtering method with phase estimation,*

(i) *The same task as in Proposition 5 can be achieved in a gate complexity of*

$$\tilde{O}\left(\frac{\Lambda}{\chi^3\Delta^\kappa} + \frac{\Lambda}{\chi\Delta^{(3-\kappa)/2}} + \frac{\Phi}{\chi\Delta^{(1-\kappa)/2}}\right) \tag{C38}$$

and the same number (C6) of qubits.

(ii) *The same task as in Corollary 2 can be achieved in a gate complexity of*

$$\tilde{O}\left(\frac{\Lambda}{\chi^3\xi^\kappa} + \frac{\Lambda}{\chi\xi^{(3-\kappa)/2}} + \frac{\Phi}{\chi\xi^{(1-\kappa)/2}}\right) \tag{C39}$$

and the same number (C37) of qubits, where $\kappa \in [0, 1]$ is arbitrary.

Proof. To prove (i), first use Proposition 2 from Appendix A with $\delta = \epsilon$ to obtain the ground energy to a precision of $\xi' = \Delta^\kappa$ for some $\kappa \in [0, 1]$ chosen below. With probability at least $1 - \epsilon$, this provides us with an interval $[a, b] \ni \lambda_0$ with $b - a = O(\Delta^\kappa)$. Let $\mu'_j = a + (b - a)j/L$ with $L = \Theta(\Delta^\kappa/\xi)$. Note that $\mu'_{j+1} - \mu'_j < \xi$. We now run the algorithm from Proposition 5, but replacing (C22) with

$$\frac{1}{\sqrt{L}}\sum_{j=0}^{L}|j\rangle|\mu'_j\rangle^{\otimes\eta}\,|\phi\rangle. \tag{C40}$$

Then, the total number of gates is

$$\tilde{O}\left(\frac{\Lambda}{\chi^3\xi'} + \frac{\sqrt{L}}{\chi}(2^{k_1}\Lambda + \Phi)\right) = \tilde{O}\left(\frac{\Lambda}{\chi^3\Delta^\kappa} + \frac{\Lambda}{\chi\Delta^{(3-\kappa)/2}} + \frac{\Phi}{\chi\Delta^{(1-\kappa)/2}}\right). \tag{C41}$$

This proves part (i). Part (ii) follows from the same argument as Corollary 2. □

Note that choosing $\kappa = 1$ gives the optimal inverse scaling in Δ and ξ, respectively, for this algorithm. Similarly as in Sec. IV, however, other values of κ can be chosen to take the other parameters into account.

APPENDIX D: CHEBYSHEV METHOD

We now show that the alternative approach of Ref. 19 of using quantum walks and Chebyshev polynomials can be used to obtain an algorithm with essentially the same runtime. We assume in this section that $\tilde{H}$ has at most $d = O(\log N)$ non-zero entries in each row/column.[30] We also assume that the spectrum of $\tilde{H}$ is contained in [0, 1/2] for simplicity.[31] Moreover, as in previous work,[4,6,19] we assume that we are given quantum oracle access to the positions and values of the non-zero elements of $\tilde{H}$. Specifically, we assume that we are given unitaries $\mathcal{O}_1, \mathcal{O}_2$ such that $\mathcal{O}_1|j, l\rangle = |j, \nu(j, l)\rangle$ and $\mathcal{O}_2|j, k, z\rangle = |j, k, z \oplus \tilde{H}_{jk}\rangle$, where $\nu(j, l)$ is the column index of the lth nonzero entry in the jth row of $\tilde{H}$, and the third register on which $\mathcal{O}_2$ acts encodes a bit string representation of an entry of $\tilde{H}$. In this section, let Λ denote the gate complexity of the oracles.

Suppose that the value of λ_0 is known to a precision of $\delta = O\left(\Delta/\log\frac{1}{\chi\epsilon}\right)$. Let E be a known real number such that $0 \le E \le \lambda_0$ and $\delta_E := \lambda_0 - E < \delta$. Define $H := \tilde{H} - (E - \tau)\mathbb{1}$. Then $|\lambda_0\rangle$ is the unique eigenvector of H with minimum eigenvalue $\tau + \delta_E$ and by assumption, all other eigenvalues of H are $\ge\tau + \delta_E + \Delta$. This method is based on the observation that a high power of $\mathbb{1} - (H/d)^2$ is approximately proportional to a projector onto $|\lambda_0\rangle$. More precisely, for any trial state $|\phi\rangle = \phi_0|\lambda_0\rangle + |\lambda_0^\perp\rangle$,

$$\left(\mathbb{1} - \left(\frac{H}{d}\right)^2\right)^M|\phi\rangle = \phi_0\left(1 - \left(\frac{\tau + \delta_E}{d}\right)^2\right)^M\left(|\lambda_0\rangle + \frac{1}{\phi_0}\left(\frac{\mathbb{1} - (H/d)^2}{1 - ((\tau + \delta_E)/d)^2}\right)^M|\lambda_0^\perp\rangle\right). \tag{D1}$$

The norm of the second term in the brackets is bounded by $|\phi_0|^{-1}e^{-\Omega(M(\tau+\delta_E)\Delta)}$. Indeed, for small Δ,

$$\left\|\left(\mathbb{1} - \left(\frac{H}{d}\right)^2\right)^M|\lambda_0^\perp\rangle\right\| < \left(\frac{1 - \left(\frac{\Delta+\tau+\delta_E}{d}\right)^2}{1 - \left(\frac{\tau+\delta_E}{d}\right)^2}\right)^M \tag{D2}$$

$$= \left(1 - \frac{2\Delta(\tau + \delta_E) + \Delta^2}{d^2\left(1 - \left(\frac{\tau+\delta_E}{d}\right)^2\right)}\right)^M \tag{D3}$$

$$= e^{-\Omega(M(\tau+\delta_E)\Delta/d^2)}. \tag{D4}$$

Thus,

$$\left\|\frac{\left(\mathbb{1} - (H/d)^2\right)^M|\phi\rangle}{\left\|\left(\mathbb{1} - (H/d)^2\right)^M|\phi\rangle\right\|} - |\lambda_0\rangle\right\| = O(\epsilon), \tag{D5}$$

provided that

$$M = \Omega\left(\frac{d^2}{\Delta(\tau + \delta_E)}\log\frac{1}{|\phi_0|\epsilon}\right). \tag{D6}$$

On the other hand,

$$\left(1-\left(\frac{\tau+\delta_E}{d}\right)^2\right)^M = e^{-O((\tau+\delta_E)^2 M/d^2)}. \tag{D7}$$

Thus,

$$\left\|\left(\mathbb{1}-\left(\frac{H}{d}\right)^2\right)^M |\phi\rangle\right\| = \Omega(|\phi_0|), \tag{D8}$$

provided that $\tau+\delta_E = O(d/\sqrt{M})$. Hence, since by assumption $\delta_E < \delta$, choosing

$$\tau = \Theta\left(\frac{\Delta}{\log\frac{1}{\chi\varepsilon}}\right) \tag{D9}$$

and

$$M = \Theta\left(\frac{d^2}{\Delta^2}\log^2\frac{1}{\chi\epsilon}\right) \tag{D10}$$

satisfies both (D5) and (D8).

Our aim in the following is to prepare $(\mathbb{1}-(H/d)^2)^M|\phi\rangle$. The strategy we employ is as follows: first, we approximate $(\mathbb{1}-(H/d)^2)^M$ as a linear combination of low order Chebyshev polynomials in H/d. Second, we implement this linear combination with some amplitude using quantum walks and the non-unitary LCU lemma (Lemma 7 of Ref. 19). Third, we use amplitude amplification or fixed point search to boost the overlap with the target state.

In the following, assume for simplicity that $M = 2m$ is even (the algorithm can also be adapted to odd M with minor modifications). Let $\mathcal{T}_k(x)$ and $\mathcal{U}_k(x)$ be the kth Chebyshev polynomials of the first and second kind, respectively. It is well-known that

$$(1-x^2)^M = \sum_{k=0}^{M}\alpha_k\mathcal{T}_{2k}(x), \tag{D11}$$

where

$$\alpha_k = \begin{cases} 2^{1-2M}\binom{2M}{M} - \dfrac{(2M-1)!!}{2^M M!}, & k=0,\\ (-1)^k 2^{1-2M}\binom{2M}{M+k}, & k\ge 1. \end{cases} \tag{D12}$$

Since $\|H\| \le 1$ and $|\mathcal{T}_k(x)| \le 1$ for $|x| \le 1$, (21) implies that

$$\left(\mathbb{1}-\left(\frac{H}{d}\right)^2\right)^{2m} = \sum_{k=0}^{m_0}\alpha_k\mathcal{T}_{2k}\left(\frac{H}{d}\right) + O(\chi\epsilon), \tag{D13}$$

provided that

$$m_0 = \Theta\left(\sqrt{M\log\frac{1}{\chi\epsilon}}\right) = \Theta\left(\frac{d}{\Delta}\log^{3/2}\frac{1}{\chi\epsilon}\right). \tag{D14}$$

Next, recall that $\mathcal{T}_k(H/d)$ can be implemented with some amplitude using quantum walks (cf. Sec. 4.1 of Ref. 19) on a larger Hilbert space, which is obtained by first adding an ancilla qubit and then doubling the entire system: for $j \in [N] := \{1,\ldots,N\}$, define $|\psi_j\rangle \in \mathbb{C}^{2N}\otimes\mathbb{C}^{2N}$ as

$$|\psi_j\rangle := |j0\rangle \otimes \frac{1}{\sqrt{d}}\sum_{\substack{l\in[N]\\ H_{jl}\neq 0}}|l\rangle\left(\sqrt{H_{jl}^*}|0\rangle + \sqrt{1-|H_{jl}|}|1\rangle\right) \tag{D15}$$

and

$$T := \sum_{j\in[N]}|\psi_j\rangle\langle j|. \tag{D16}$$

Note that the entries of H have modulus at most 1. Let S be the swap operator on $\mathbb{C}^{2N}\otimes\mathbb{C}^{2N}$, i.e., $S|jb_1\rangle\,|lb_2\rangle = |lb_2\rangle\,|jb_1\rangle$ and $W = S(2TT^\dagger - \mathbb{1})$. By Lemma 16 of Ref. 19, within the invariant subspace $\text{span}\{T|j\rangle, ST|j\rangle : j\in[N]\}$, W^k has the form

$$W^k = \begin{pmatrix} \mathcal{T}_k(H/d) & -\sqrt{1-(H/d)^2}\,\mathcal{U}_{k-1}(H/d) \\ \sqrt{1-(H/d)^2}\,\mathcal{U}_{k-1}(H/d) & \mathcal{T}_k(H/d) \end{pmatrix}, \tag{D17}$$

where the first block corresponds to the space span$\{T|j\rangle : j \in [N]\}$. Hence, using k steps of the walk, we can implement the transformation

$$W_k|0\rangle^{\otimes q}|\phi\rangle = |0\rangle^{\otimes q}\mathcal{T}_k(H/d)|\phi\rangle + |R'_k\rangle, \tag{D18}$$

where $q := \lceil \log_2 N \rceil + 3$ and $(|0\rangle\langle 0|^{\otimes q} \otimes \mathbb{1})|R'_k\rangle = 0$.

To implement the RHS of (D13), we employ the non-unitary LCU lemma: let B be a circuit on $b = \lceil \log_2(m_0 + 1) \rceil$ qubits that map $|0\rangle^{\otimes b}$ to

$$B|0\rangle^{\otimes b} := \frac{1}{\sqrt{\alpha}} \sum_{k=0}^{m_0} \sqrt{\alpha_k}|k\rangle, \tag{D19}$$

where $\alpha = \sum_{k=0}^{m_0} \alpha_k$. Let $U = \sum_{k=0}^{m_0} |k\rangle\langle k| \otimes W_{2k}$ be the controlled quantum walk. Then,

$$(B^\dagger \otimes \mathbb{1})U(B \otimes \mathbb{1})|0\rangle^{\otimes(b+q)}|\phi\rangle = \frac{1}{\alpha}|0\rangle^{\otimes b} \sum_{k=0}^{m_0} \alpha_k(|0\rangle^{\otimes q}\mathcal{T}_{2k}(H/d)|\phi\rangle + |R'_{2k}\rangle) + |R'\rangle \tag{D20}$$

$$= \frac{1}{\alpha}|0\rangle^{\otimes(b+q)} \sum_{k=0}^{m_0} \alpha_k \mathcal{T}_{2k}(H/d)|\phi\rangle + |R\rangle, \tag{D21}$$

where $(|0\rangle\langle 0|^{\otimes b} \otimes \mathbb{1})|R'\rangle = 0$ and $(|0\rangle\langle 0|^{\otimes(b+q)} \otimes \mathbb{1})|R\rangle = 0$.

The final step of the algorithm is to boost the overlap with amplitude amplification or fixed point search. Note that amplitude amplification can be used without prior knowledge of the overlap.[21] Alternatively, fixed point search[25] can be used for this step. Measuring the ancillas will then project the state onto

$$|\lambda'_0\rangle := \frac{\sum_{k=0}^{m_0} \alpha_k \mathcal{T}_{2k}(H/d)|\phi\rangle}{\| \sum_{k=0}^{m_0} \alpha_k \mathcal{T}_{2k}(H/d)|\phi\rangle \|}, \tag{D22}$$

provided we successfully obtain $|0\rangle^{\otimes(b+q)}$ on the ancillas. From (D13),

$$|\lambda'_0\rangle = \frac{(\mathbb{1} - (H/d)^2)^{2m}|\phi\rangle}{\|(\mathbb{1} - (H/d)^2)^{2m}|\phi\rangle\|} + O(\epsilon), \tag{D23}$$

and thus (D5) implies

$$|\lambda'_0\rangle = |\lambda_0\rangle + O(\epsilon), \tag{D24}$$

as required. The probability of success is close to 1, provided that the number of repetitions is

$$O\left(\frac{\alpha}{\| \sum_{k=0}^{m_0} \alpha_k \mathcal{T}_{2k}(H/d)|\phi\rangle \|}\right) = O(\alpha/|\phi_0|), \tag{D25}$$

where Eq. (D25) follows from (D8).

We now calculate the gate count of the entire algorithm. First note that B can be implemented with $O(2^b) = O(m_0)$ elementary gates.[26] Next, note that the oracle to H can be obtained from the oracle to $\tilde{H}$ with $O(\log M)$ additional gates and qubits. The gate cost to implement W to accuracy ϵ' is $O(\Lambda + \log M + \log N + \log^{5/2}(1/\epsilon'))$.[4] Here, we require $\epsilon' = O(\epsilon|\phi_0|/m_0 d)$. Thus, the gate cost of U is $O(m_0(\Lambda + \log M + \log N + \log^{5/2}(m_0 d/\epsilon|\phi_0|))$ (Lemma 8 of Ref. 19). Note that $\alpha = O(1)$, and each iteration of amplitude amplification or fixed point search requires $O(1)$ uses of $\mathcal{C}_\phi$, B, and U. The final gate complexity is thus

$$O\left(\frac{1}{|\phi_0|}\left(m_0\left(\Lambda + \log M + \log N + \log^{5/2}\frac{m_0 d}{\epsilon|\phi_0|}\right) + \Phi\right)\right) = O\left(\frac{\Lambda}{|\phi_0|\Delta}\operatorname{polylog}\left(N, \frac{1}{\Delta}, \frac{1}{|\phi_0|\epsilon}\right) + \frac{\Phi}{|\phi_0|}\right). \tag{D26}$$

The total number of qubits required is $O(\log N + \log M + \log m_0)$. □

It is moreover easy to see that, analogously to Sec. IV B, this approach can also be used for ground state preparation in the case of unknown ground energy, and for estimating the ground energy.

REFERENCES

[1] R. Feynman, Int. J. Theor. Phys. **21**, 467 (1982).

[2] S. Lloyd, Science **273**, 1073 (1996).

[3] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma, Phys. Rev. Lett. **114**, 090502 (2015).

[4] D. W. Berry, A. M. Childs, and R. Kothari, in *2015 IEEE 56th Annual Symposium on Foundations of Computer Science* (IEEE, 2015), pp. 792–809.

[5] G. H. Low and I. L. Chuang, Phys. Rev. Lett. **118**, 010501 (2017).

[6] G. H. Low and I. L. Chuang, preprint arXiv:1610.06546 [quant-ph] (2016).

[7] S. Gharibian, Y. Huang, Z. Landau, and S. W. Shin, Found. Trends Theor. Comput. Sci. **10**, 159 (2015).

[8] A. Y. Kitaev, preprint arXiv:quant-ph/9511026 [quant-ph] (1995).

[9] D. Poulin and P. Wocjan, Phys. Rev. Lett. **102**, 130503 (2009).

[10] D. S. Abrams and S. Lloyd, Phys. Rev. Lett. **83**, 5162 (1999).

[11] D. A. Abanin and Z. Papi, Ann. Phys. **529**, 1700169 (2017).

[12] A. Polkovnikov, K. Sengupta, A. Silva, and M. Vengalattore, Rev. Mod. Phys. **83**, 863 (2011). L. D'Alessio, Y. Kafri, A. Polkovnikov, and M. Rigol, Adv. Phys. **65**, 239 (2016).

[13] E. Farhi, D. Gosset, A. Hassidim, A. Lutomirski, D. Nagaj, and P. Shor, Phys. Rev. Lett. **105**, 190503 (2010).

[14] M.-H. Yung, J. D. Whitfield, S. Boixo, D. G. Tempel, and A. Aspuru-Guzik, "Introduction to quantum algorithms for physics and chemistry," in *Quantum Information and Computation for Chemistry* (John Wiley & Sons, Inc., 2014), pp. 67–106.

[15] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Nature **549**, 195 (2017).

[16] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, preprint arXiv:quant-ph/0001106 [quant-ph] (2000).

[17] S. Jansen, M.-B. Ruskai, and R. Seiler, J. Math. Phys. **48**, 102111 (2007).

[18] S. Oh, Phys. Rev. A **77**, 012326 (2008).

[19] A. Childs, R. Kothari, and R. Somma, SIAM J. Comput. **46**, 1920 (2017).

[20] Examples of such algorithms include Refs. 3–6. Notice that algorithms based on Trotter product formulas such as Ref. 2 do not meet this requirement.

[21] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, *Quantum Computation and Information*, AMS Contemporary Mathematics Series (AMS, 2002), Vol. 305.

[22] In fact, the Hamiltonian simulation algorithms[4–6] only require few ancilla qubits and leave the $O(\cdot)$ expressions in Tables I and II unchanged. Reference 3 requires $O\left(\frac{\log(d)\log(\beta\Delta^{-1}\log(\chi^{-1})/\epsilon)}{\log\log(\beta\Delta^{-1}\log(\chi^{-1})/\epsilon)}\right)$ additional qubits, where $\tilde{H} = \sum_{j=1}^{d}\beta_j U_j$ with unitaries U_j costing $O(\Lambda)$ elementary gates and $\beta = \sum_j |\beta_j|$ (see Table 1 of Ref. 6 for an overview).

[23] R. V. Mises and H. Pollaczek-Geiringer, Z. Angew. Math. Mech. **9**, 152 (1929).

[24] J. van Apeldoorn, A. Gilyén, S. Gribling, and R. de Wolf, *IEEE 58th Annual Symposium on Foundations of Computer Science* (IEEE, 2017), pp. 403–414.

[25] T. J. Yoder, G. H. Low, and I. L. Chuang, Phys. Rev. Lett. **113**, 210501 (2014).

[26] V. V. Shende, S. S. Bullock, and I. L. Markov, IEEE Trans. Comput. -Aided Des. Integr. Circuits Syst. **25**, 1000 (2006).

[27] Note that unlike Ref. 26, where $\sqrt{\lambda}$ denotes the overlap, here we write the overlap as λ.

[28] R. Kothari, private communication (2017).

[29] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, 10th ed. (Cambridge University Press, New York, NY, USA, 2011).

[30] Notice that this includes many-body Hamiltonians, as Hamiltonians consisting of n terms acting on at most k qubits are sparse with $d = 2^k n$.

[31] In fact, it is sufficient to assume that the spectrum of $\tilde{H}$ is contained in $[0, 1 - \tau]$, where τ is defined in (D9). This ensures that H, as defined below, has entries with modulus at most 1.

B Further articles as principal author currently under review

B.1 A hybrid algorithm framework for small quantum computers with application to finding Hamiltonian cycles

A hybrid algorithm framework for small quantum computers with application to finding Hamiltonian cycles

Yimin Ge and Vedran Dunjko

In this work, we generalise the approach of Contributed Article VI by developing a general framework for hybrid quantum-classical algorithms which only use quantum computers significantly smaller than the problem size. We prove general criteria such that, given an arbitrarily small ratio of the number of available qubits to the instance size, a polynomial speedup for classical divide-and-conquer algorithms can be achieved. We also demonstrate the applicability of this framework by speeding up Eppstein's algorithm for the cubic Hamiltonian cycle problem.

Section III of this work develops the so-called *divide-and-conquer hybrid approach*, which is a general framework for speeding up classical divide-and-conquer algorithms. The main idea is to replace the recursive call with a call to a suitable quantum algorithm once the instance size becomes sufficiently small for the number of available qubits. Theorem 1 formalises this idea, and explicitly makes the trade-off between the number of available qubits, the resource requirement of the quantum subroutine, and the speedup obtained. We also explicitly demonstrate how the results of Contributed Article VI fit into this framework (Example 3).

Although the basic idea of the divide-and-conquer hybrid approach is simple, in order to obtain a polynomial speedup over the original classical algorithm, the underlying quantum subroutine needs to fulfil strict criteria for space-efficiency whilst also being (at least) polynomially faster than the original algorithm. In many cases, the main contribution to the space requirement of the quantum subroutine comes from the necessity of generating and storing large sets. Indeed, a naive encoding of large sets as ordered lists is demonstrated to usually lead to a less-than-polynomial speedup (Example 2). On the other hand, straightforward encodings as genuine sets would naively lead to problems with reversibility, which is naturally required for quantum algorithms. Turning non-reversible computations into reversible ones in the canonical way either introduces many ancillas, or can incur exponential overheads, both of which would prevent any reasonable speedup. Indeed, combining the seemingly competing requirements of reversibility, space-efficiency and time-efficiency for our purposes is non-trivial.

In Section IV, we show how to overcome this problem. In particular, Theorem 2 provides a classical reversible set generation routine which can be used to obtain quantum algorithms that are compatible with a polynomial speedup in the divide-and-conquer hybrid approach. For this, specialised data-structures to encode large sets are first introduced in Section IV A, which will later allow the required trade-off between space-efficiency and efficient uncomputation. The main idea here is to split the large set into smaller subsets of suitable sizes, thus adding just enough ordering information to be able to uncompute only the smaller subsets individually. This allows us to keep the overall number of ancillas used low. Afterwards, in Section IV B, we provide the general algorithm to generate an efficient encoding of a large set, given only access to operations which generate single elements.

Section IV then provides a novel example of how these tools can be applied in practice. Eppstein's algorithm, which decides if a cubic graph of n vertices has a Hamiltonian cycle in runtime $O^*(2^{n/3})$, is a classical divide-and-conquer algorithm that naturally fits the framework of Theorem 1. The main ingredient for a speedup then becomes a polynomially faster quantum algorithm that solves the "forced" version of this problem using sufficiently few qubits. Theorem 3 proves the existence of such a quantum algorithm. The proof of Theorem 3 heavily utilises the set generation routine of Theorem 2, which reduces the task to implementing a small number of problem-specific, i.e. graph-theoretic, operations. Theorem 3, together with Theorem 1, then immediately imply a polynomial speedup of Eppstein's algorithm of runtime $O^*(2^{(1/3-f(c))n})$ using only $M = cn$ qubits, where $c > 0$ is an arbitrary constant and $f(c) > 0$. This is formally stated in Theorem 4.

Statement of individual contribution

This work was motivated by several discussions between Vedran Dunjko and myself. I had the idea of speeding up Eppstein's algorithm using similar methods to the ones we developed in Contributed Article VI, and subsequently worked out the details of the algorithm and the proofs involved. Afterwards, with regular advice from Vedran Dunjko, I formulated the more general framework for developing hybrid algorithms. I was in charge of writing all parts of this article.

I, Yimin Ge, am the principal author of this article and was extensively involved in all parts of it.

Permission to include:

Yimin Ge and Vedran Dunjko.
A hybrid algorithm framework for small quantum computers with application to finding Hamiltonian cycles.
arXiv:1907.01258 [quant-ph], 2019
Submitted to *Journal of Mathematical Physics*, July 2019.

Permission to Reuse Content

REUSING AIP PUBLISHING CONTENT

Permission from AIP Publishing is required to:

- republish content (e.g., excerpts, figures, tables) if you are not the author
- modify, adapt, or redraw materials for another publication
- systematically reproduce content
- store or distribute content electronically
- copy content for promotional purposes

To request permission to reuse AIP Publishing content, use RightsLink® for the fastest response or contact AIP Publishing directly at rights@aip.org and we will respond within one week:

For RightsLink, use Scitation to access the article you wish to license, and click on the Reprints and Permissions link under the TOOLS tab. (For assistance click the "Help" button in the top right corner of the RightsLink page.)

To send a permission request to rights@aip.org, please include the following:

- Citation information for the article containing the material you wish to reuse
- A description of the material you wish to reuse, including figure and/or table numbers
- The title, authors, name of the publisher, and expected publication date of the new work
- The format(s) the new work will appear in (e.g., print, electronic, CD-ROM)
- How the new work will be distributed and whether it will be offered for sale

Authors do **not** need permission from AIP Publishing to:

- quote from a publication (please include the material in quotation marks and provide the customary acknowledgment of the source)
- reuse any materials that are licensed under a Creative Commons CC BY license (please format your credit line: "Author names, Journal Titles, Vol.#, Article ID#, Year of Publication; licensed under a Creative Commons Attribution (CC BY) license.")
- reuse your own AIP Publishing article in your thesis or dissertation (please format your credit line: "Reproduced from [FULL CITATION], with the permission of AIP Publishing")
- reuse content that appears in an AIP Publishing journal for republication in another AIP Publishing journal (please format your credit line: "Reproduced from [FULL CITATION], with the permission of AIP Publishing")
- make multiple copies of articles–although you must contact the Copyright Clearance Center (CCC) at www.copyright.com to do this

(…)

https://publishing.aip.org/resources/researchers/rights-and-permissions/permissions/ 03/12/2019

A hybrid algorithm framework for small quantum computers with application to finding Hamiltonian cycles

Yimin Ge[1, *] and Vedran Dunjko[2, †]

[1] *Max-Planck-Institut für Quantenoptik, Hans-Kopfermann-Str. 1, 85748 Garching, Germany*
[2] *LIACS, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, Netherlands*

Recent works have shown that quantum computers can polynomially speed up certain SAT-solving algorithms even when the number of available qubits is significantly smaller than the number of variables. Here we generalise this approach. We present a framework for hybrid quantum-classical algorithms which utilise quantum computers significantly smaller than the problem size. Given an arbitrarily small ratio of the quantum computer to the instance size, we achieve polynomial speedups for classical divide-and-conquer algorithms, provided that certain criteria on the time- and space-efficiency are met. We demonstrate how this approach can be used to enhance Eppstein's algorithm for the cubic Hamiltonian cycle problem, and achieve a polynomial speedup for any ratio of the number of qubits to the size of the graph.

I. INTRODUCTION

Although fully scalable quantum computers may be far off, small quantum computers may be achievable in the forseeable future. Such devices may able to provide solutions to toy or specialised problems of small size (e.g. in quantum chemistry), it was however until recently unclear whether they could also be utilised for speeding up more general and common computations of much larger problem instances. Indeed, quantum and classical algorithms usually exploit global structures inherent to the problem, and it is generally difficult to utilise small quantum computers without breaking that structure. For example, the ability to factor $n/10$-digit integers is unlikely to be of much help for the task of factoring n-digit integers. One would therefore naively expect that for structured problems, small quantum computers would only be useful for small problem sizes.

Recently in [1], it was show that this is not generally true: given a quantum computer with only M qubits, it was shown that one can obtain a significant speedup of Schöning's algorithm for solving 3SAT involving $n \gg M$ variables. More precisely, the speedup can be expressed in terms of the ratio $c = M/n$ of available qubits to the problem size, and it was shown that an asymptotic polynomial speedup can be achieved for arbitrarily small values of c. The latter is non-trivial, since it was also shown that employing a naive approach to speed up the classical algorithm breaks the exploited problem structure, thus resulting in no improvement unless c is quite large.

One of the main insights of [1] was that classical divide-and-conquer algorithms inherently don't suffer from this "threshold effect" since they naturally maintain the structure of the problem despite breaking it into smaller subproblems, and are thus well-suited for being enhanced using small quantum computers. Yet, to achieve genuine speedups, the quantum subroutines employed must meet stringent criteria for space- and time-efficiency, which in general are non-trivial to fulfil and, in the case of [1], required specialised data-structures and careful memory management. Moreover, [1] exclusively considers the example of Schöning's 3SAT algorithm, and while it demonstrated how divide-and-conquer structures can in principle be exploited, it left open a formal characterisation of the criterion for when this approach works, and whether other examples beyond Schöning's 3SAT algorithm exist where similar speedups can be obtained [2]. Given the ubiquity of classical divide-and-conquer algorithms, a general framework for developing such hybrid algorithms to obtain speedups using only small quantum computers would thus be highly desirable.

In this work, we formalise and generalise the criteria for the hybrid approach of [1], and show that indeed, there are other problems and algorithms that can be enhanced with that approach, using only small quantum computers. Specifically, we develop a general framework for constructing hybrid algorithms that speed up certain classical divide-and-conquer algorithms using significantly fewer qubits than the problem size, which also makes precise the relation between the speedup obtained on the one hand, and the space requirements and runtime of the underlying quantum subroutines on the other. We then apply this formalism to finding Hamiltonian cycles on cubic graphs, which is another fundamental NP-complete problem. This provides the first example of the applicablity of these techniques beyond the example of Schöning's 3SAT algorithm given in [1]. Our framework operates on the algorithmic level and is distinct from the circuit-level techniques of [3, 4], which aim to simulate general quantum circuits on fewer qubits.

[*] yimin.ge@mpq.mpg.de
[†] v.dunjko@liacs.leidenuniv.nl

The efficiency of the latter depends on sparseness or decomposability assumptions on the original circuits which will in general prevent speedups for the algorithms we consider here.

We provide our formalism for constructing hybrid algorithms from classical algorithms in the form of a "toolkit" comprising two parts. The first part, which we term the *divide-and-conquer hybrid approach*, shows general criteria for the kind of classical algorithms one can speed up using our techniques, and relates the speedup to the number of available qubits. The second part, which we term *efficient reversible set generation*, comprises specialised data-structures. These are specifically designed to bridge the gap between two seemingly irreconciliable properties required of the quantum algorithm for a polynomial speedup in the divide-and-conquer hybrid approach: reversibility on the one hand, and being extremely space-efficient on the other. The set generation procedure we develop makes the task of developing quantum algorithms suitable for our hybrid approach significantly easier: we show that it suffices to find space-efficient implementations for a few problem-specific quantum operations.

Finally, we illustrate how this framework is applied to speed up Eppstein's algorithm [5] for the cubic Hamiltonian cycle problem with only a small quantum computer. The cubic Hamiltonian cycle problem asks if a given cubic graph of n vertices has a *Hamiltonian cycle*, i.e. a cycle visiting every vertex exactly once. This problem is NP-complete, and a special case of the general Hamiltonian cycle problem (where no restrictions on the maximum degree of the graph is assumed), which in turn is closely linked to the travelling salesman problem. Brute-force search requires $O(n! \operatorname{poly}(n))$ time, there is however also a trivial path-search algorithm with runtime $O(2^n \operatorname{poly}(n))$. In 2004, Eppstein [5] gave a divide-and-conquer algorithm of runtime $O(2^{n/3} \operatorname{poly}(n)) = O(1.2599^n \operatorname{poly}(n))$, which heavily exploits the cubic structure of the graph. Quantum speedups for Eppstein's algorithm have previously been obtained using arbitrarily-sized quantum computers [6]. In this work, we obtain a polynomial speedup using only $M = cn$ qubits for arbitrarily small $c > 0$.

The outline of the remainder of this paper is as follows. In Section II, we give a brief overview of the results and clarify some notation. In Section III, we formulate the divide-and-conquer hybrid approach for a general class of classical algorithms. In Section IV, we provide the details of the efficient and reversible set generation procedure. In Section V, we apply these tools to Eppstein's algorithm for the cubic Hamiltonian cycle problem. Finally, we close the paper with some concluding remarks and open questions in Section VI.

II. OVERVIEW

A. Overview of results

We briefly summarise the results and main ideas of this paper.

Sections III and IV set up the general framework for designing hybrid algorithms for quantum computers significantly smaller than the problem size. Specifically, Section III introduces the divide-and-conquer hybrid approach, and outlines the general criteria for the kind of classical algorithms which our framework is applicable to. Theorem 1 then formalises the trade-off between the number of available qubits, the space-requirement of the underlying quantum subroutine, and the speedup obtained. The main idea of this hybrid approach is to take a classical divide-and-conquer algorithm, which calls itself on ever smaller problem instances, and to replace the recursive call with a suitable quantum algorithm once the problem instance is sufficiently small to fit the number of available qubits. While the basic idea is simple, in order to obtain a polynomial speedup over the original classical algorithm, the replacing quantum algorithm has to fulfill strict criteria for space-efficiency whilst also being polynomially faster. In many cases, the main contribution to the quantum algorithm's space-requirement comes from the necessity of generating and storing large sets.

Section IV then shows how to do the latter efficiently. In particular, Theorem 2 provides a (classical) reversible set generation routine which can be used to obtain quantum algorithms that are compatible with a polynomial speedup when used in Theorem 1. To that end, specialised set encodings are first introduced which are designed to overcome the main challenge of such an implementation: the ability to uncompute encodings of previously generated sets without resulting in either large computational overheads or large memory requirements.

Section V then provides an example of how these tools can be applied in practice. Eppstein's algorithm for finding Hamiltonian cycles on cubic graphs is a classical divide-and-conquer algorithm that naturally fits the framework of Theorem 1. The main ingredient for a speedup then becomes a polynomially faster quantum algorithm that solves this problem using sufficiently few qubits. Theorem 3 proves the existence of such a quantum algorithm. The proof of Theorem 3 heavily utilises the set generation routine of Theorem 2, which reduces the task to implementing a small number of problem-specific, i.e. graph-theoretic, operations. Theorem 3, together with Theorem 1, then immediately imply a polynomial speedup of Eppstein's algorithm using only significantly fewer qubits than the size of the graph, which is formally stated in Theorem 4.

B. Related work

The present work is closely linked to the results obtained in [1], which however exclusively focuses on speeding up Schöning's algorithm for 3SAT. By contrast, the results presented in Sections III and IV of this work provide a general framework which is not tied to any specific algorithm. In Section III, we provide general criteria for when similar speedups as the one in [1] can be obtained for other classical algorithms, and moreover explicitly relate the degree of the speedup to the resource requirements of the underlying quantum subroutine and the number of qubits available. The main ideas of the efficient and reversible set generation procedure given in Section IV can also be found in [1]. However, the presentation in [1] is specific to speeding up Schöning's algorithm, and potentially more general structures are often strongly interwoven with 3SAT-specific constructions there. By contrast, in this work and specifically in Theorem 2, we provide a clean formulation that explicitly separates the more abstract structures from problem-specific implementations, thus obtaining a general framework that can be more easily applied in different scenarios. For example, the general formulation of the set generation procedure vastly simplifies the construction of a suitable quantum subroutine for polynomially speeding Eppstein's algorithm later in Section V.

C. Notation

Throughout this paper, we will use standard bra/ket notation for quantum states. We will also use bra/ket notation in the context of classical reversible circuits, since they can be seen as special cases of quantum circuits.

Moreover, for simplicity of notation, we will often make several notational simplifications. First, we often simply write $|0\rangle$ for $|0\rangle^{\otimes L}$ for any known $L \in \mathbb{N}$. The value of L will always be clear from context. Second, for operators acting on some registers of a multi-register state, we will normally not explitly write the complementing identiy operators (e.g., we will simply write $A\,|a\rangle\,|b\rangle\,|c\rangle\,|d\rangle$ instead of $(\mathbb{1} \otimes A \otimes \mathbb{1})\,|a\rangle\,|b\rangle\,|c\rangle\,|d\rangle$ if A acts on the middle two registers). It will always be clear from context which registers which operators act on.

III. THE DIVIDE-AND-CONQUER HYBRID APPROACH

In this section, we formalise the *divide-and-conquer hybrid approach*, generalising the techniques of [1], for designing hybrid algorithms using only quantum computers significantly smaller than the problem size. The main idea is to take a classical divide-and-conquer [7] algorithm that calls itself on ever smaller (effective) problem sizes, and replace the recursive calls with a quantum algorithm once the problem size becomes sufficiently small.

Let $\mathcal{P}$ be a countable set, $\mathcal{A} : \mathcal{P} \to \{0, 1\}$ be a decision problem [8], and $n : \mathcal{P} \to \mathbb{N}$ be a problem parameter. We refer to $n(P)$ as the *problem size* of P.

```
procedure Alg($P$)
    if Trivial($P$) $= 1$
        return $f(P)$
    else
        return $g(\text{Alg}(R_1(P)), \ldots, \text{Alg}(R_l(P)))$
end procedure
```

ALG 1: General algorithm for the divide-and-conquer hybrid approach

Suppose that $\mathcal{A}(P)$ can be decided by a classical recursive algorithm [9] of the form given in Alg. 1, where $l \geq 2$ is an integer, $R_1, \ldots, R_l : \mathcal{P} \to \mathcal{P}$, $g : \{0,1\}^l \to \{0,1\}$, and $f, \textsc{Trivial} : \mathcal{P} \to \{0,1\}$. We assume that $\textsc{Trivial}(P), R_1(P), \ldots, R_l(P)$ can be calculated in $O(\text{poly}(n(P)))$ time, and that $f(P)$ can be calculated in $O(\text{poly}(n(P)))$ time if $\textsc{Trivial}(P) = 1$. The maps $R_1, \ldots, R_l$ can be thought of as "reduction operations", mapping the problem to a smaller instance, whereas $\textsc{Trivial}(P)$ signifies if P is sufficiently simple to be solved directly. We assume that for all $P \in \mathcal{P}$ and $i = 1, \ldots, l$, $n(R_i(P)) \leq n(P)$.

The runtime of such divide-and-conquer algorithms can often be bounded by introducing an *effective problem size* $s : \mathcal{P} \to \mathbb{N}$. In general, $s(P)$ and $n(P)$ can be different, we assume however that for all $P \in \mathcal{P}$, $s(P) \leq n(P)$. We assume that both $n(P)$ and $s(P)$ can be calculated in time $O(\text{poly}(n(P)))$. We moreover assume that there is a universal constant $C \in \mathbb{N}$ such that $s(P) \leq C$ implies $\textsc{Trivial}(P) = 1$.

To ensure that $\textsc{Alg}(P)$ has a runtime of the form $O(2^{\gamma s(P)}\,\text{poly}(n(P)))$ for some constant $\gamma > 0$, we assume that there exist integers $k > 0$, $C_{ij} > 0$ for $1 \leq i \leq l, 1 \leq j \leq k$, and $l_1, \ldots, l_k \in \{1, \ldots, l\}$ such that for all $P \in \mathcal{P}$ with

$\textsc{Trivial}(P) = 0$, there exists some $j \in \{1, \dots, k\}$ such that for all $i = 1, \dots, l$, we have

$$s(R_i(P)) \leq s(P) - C_{ij} \quad \text{or} \quad \textsc{Trivial}(R_i(P)) = 1 \tag{1}$$

for $i = 1, \dots, l_j$, and $\textsc{Trivial}(R_i(P)) = 1$ for $i = l_j + 1, \dots, l$. Here, $j \in \{1, \dots, k\}$ labels one of k possible cases for effective problem size reductions, and $l_j \leq l$ the effective number of recursive branches of that case, whereas the positive integers C_{ij} are lower bounds on the decrease of the effective problem size, guaranteeing that the algorithm terminates.

Under these assumptions, it is easy to derive the stated upper bound on the runtime of running $\textsc{Alg}(P)$. Indeed, (1) implies a recursive runtime bound $T(s(P))$ depending only on $s(P)$ given by

$$T(s') \leq \max_{j=1,\dots,k} (T(s' - C_{1j}) + \dots + T(s' - C_{l_j j})) + O(\text{poly}(n(P))) \tag{2}$$

and $T(s') = O(\text{poly}\, n(P))$ for $s' \leq C$. Using standard methods for solving recurrence relations [10], this leads to a runtime of $T(s') = O(2^{\gamma s'} \text{poly}(n(P)))$ for some $\gamma > 0$. Thus, $\textsc{Alg}(P)$ has a runtime of $O(2^{\gamma s(P)} \text{poly}(n(P)))$.

Our aim is to provide criteria for when a reduction of the value of γ, i.e. a polynomial speedup, can be achieved.

Theorem 1 (Divide-and-conquer hybrid approach)**.** *Suppose that there is a quantum algorithm* $\textsc{QAlg}$ *that decides* $\mathcal{A}(P)$ *using at most* $G(s(P), n(P))$ *qubits in time* $O(2^{\gamma_Q s(P)} \text{poly}(n(P)))$ *for some constant* $\gamma_Q \in [0, \gamma)$*. Suppose that* $G : [0, \infty) \times [0, \infty) \to [0, \infty)$ *has the property that for all nonnegative integers* $s \leq n' \leq n$, $G(s, n') \leq G(s, n)$*. Suppose moreover that there exists some* $\tilde{\lambda} \in (0, 1)$ *such that for all* $\lambda \in [0, \tilde{\lambda}]$ *and* $n \in \mathbb{N}$,

$$G(\lambda n, n) = nF(\lambda) + O(\log n) \tag{3}$$

for some strictly monotonically increasing and continuously differentiable $F : [0, \tilde{\lambda}] \to [0, \infty)$ *such that* $F(0) = 0$ *and* F' *is bounded away from* 0.

Let $c \in (0, F(\tilde{\lambda}))$ *be an arbitrary constant. Then, given a quantum computer with* $M = cn(P)$ *qubits, there exists a hybrid quantum-classical algorithm that decides* $\mathcal{A}(P)$ *in a runtime of* $O(\max(2^{\gamma s(P) - f(c) n(P)}, 2^{\gamma_Q s(P)}) \text{poly}(n(P)))$*, where* $f(c) = (\gamma - \gamma_Q) F^{-1}(c) > 0$*. In particular,* $\mathcal{A}(P)$ *can be decided in* $O(2^{(\gamma - f(c)) n(P)} \text{poly}(n(P)))$*.*

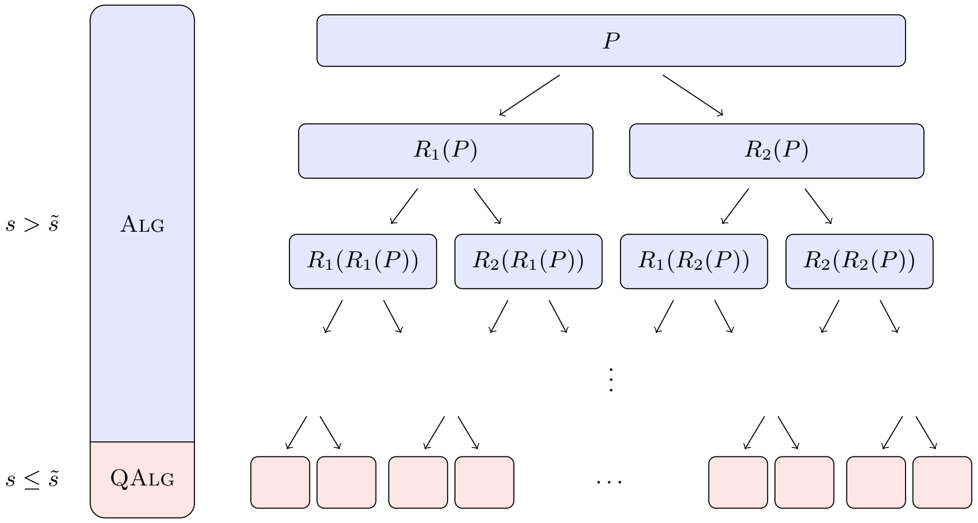


FIG. 1: Schematic view of the hybrid algorithm for $l = 2$. Recursive calls to $\textsc{Alg}$ are replaced with calls to $\textsc{QAlg}$ once the effective problem size becomes sufficiently small.

Proof. The proof is based on the ideas developed in [1] in the context of 3SAT. The main idea of the hybrid algorithm is to call $\textsc{QAlg}(P')$ instead of $\textsc{Alg}(P')$ in the recursive step of $\textsc{Alg}$ when $s(P')$ is sufficiently small (see Fig. 1).

By assumption, running $\textsc{QAlg}(P')$ for any $P' \in \mathcal{P}$ with $n(P') \leq n(P)$ requires at most $G(s(P'), n(P')) \leq G(s(P'), n(P)) \leq n(P) F(s(P')/n(P)) + a \ln n(P)$ qubits for some constant $a > 0$. Hence, if $n(P') \leq n(P)$, then

$$s(P') \leq n(P) F^{-1} \left(c - \frac{a \ln n(P)}{n(P)} \right) =: \tilde{s} \tag{4}$$

is a sufficient condition for being able to run QAlg(P') with $M = cn(P)$ qubits. Note that since F is strictly increasing and continuously differentiable with its derivative bounded away from 0, the same applies to F^{-1}. Thus, by the mean value theorem, $\tilde{s} = F^{-1}(c)n(P) - O(\log n(P))$.

Let HybridAlg(P') be the algorithm which calls QAlg(P') if $s(P') \le \tilde{s}$ and Alg$'$(P') otherwise, where Alg$'$ is the same algorithm as Alg except that in line 5 of Alg, the calls to Alg are replaced by calls to HybridAlg. Note that since $n(R_i(P')) \le n(P')$ for all $P' \in \mathcal{P}$ and $i = 1, \dots, l$, $M = cn(P)$ qubits suffice to run HybridAlg(P).

We now analyse the runtime of HybridAlg. Note that its runtime can be bounded by the function $T_H(s(P))$ depending only on $s(P)$, where T_H is given recursively by

$$T_H(s') = O(2^{\gamma_Q s'} \operatorname{poly}(n(P))) \tag{5}$$

for $s' \le \tilde{s}$ and

$$T_H(s') \le \max_{j=1,\dots,k} (T_H(s' - C_{1j}) + \cdots + T_H(s' - C_{l_j j})) + O(\operatorname{poly}(n(P))) \tag{6}$$

for $s' > \tilde{s}$. Using standard recurrence relation techniques, we therefore obtain

$$T_H(s') = O(2^{\gamma(s'-\tilde{s})+\gamma_Q \tilde{s}} \operatorname{poly}(n(P))) = O(2^{\gamma s' - (\gamma - \gamma_Q)\tilde{s}} \operatorname{poly}(n(P))) = O(2^{\gamma s' - f(c)n(P)} \operatorname{poly}(n(P))), \tag{7}$$

for $s' > \tilde{s}$, which proves the claim. □

We remark that the assumptions of Theorem 1 can be relaxed in various ways. First, it is sufficient for QAlg to only decide $\mathcal{A}(P)$ for $P \in \bigcup_{i=1}^{l} R_i(\mathcal{P})$. Second, F being continuously differentiable can be relaxed to F^{-1} being locally Lipschitz-continuous.

Note moreover that the assumption that F' is bounded away from 0 ensures that the degree of the polynomial overhead of the hybrid algorithm can be bounded by a constant independent of c. More precisely, if $F'(\lambda) \ge \kappa > 0$ for all $\lambda \in [0, \tilde{\lambda}]$, then the hybrid algorithm decides $\mathcal{A}(P)$ in a runtime of $O(\max(2^{\gamma s(P) - f(c)n(P)}, 2^{\gamma_Q s(P)}) n(P)^{O(1/\kappa)})$.

Example 1. Suppose that in Theorem 1, running QAlg(P) requires $O(s \log(n/s) + s + \log n)$ qubits, where here we just write s, n instead of $s(P), n(P)$ for simplicity. In that case, $G(s, n) = As \ln(n/s) + Bs + O(\log n)$ for some constants $A, B > 0$. Then, $F(\lambda) = A\lambda \ln(1/\lambda) + B\lambda$, which is monotonically increasing on $(0, e^{B/A-1})$. It can be shown that $F^{-1}(c) = -c/(AW_{-1}(-ce^{-B/A}/A))$, where W_{-1} is the -1 branch of the Lambert W function. It is easy to see that for small values of c, $F^{-1}(c) = \Theta(c/\log(1/c))$.

The curious expression $O(s \log(n/s) + s)$ stems from the information-theoretic cost of encoding a subset of $\{1, \dots, O(n)\}$ of size $O(s)$. It is moreover the scaling obtained in Ref [1] (see Example 3 below) and also later in Section IV and V. ■

Example 2. Suppse that QAlg requires $\ge s \log_2 n$ qubits instead. Note that this does not satisfy the requirements of Theorem 1. Then, with the same notation as in the proof of Theorem 1, $\tilde{s} = cn/\log_2 n$, and hence

$$T_H(n) = O(2^{(\gamma - \frac{c}{\log_2 n})n} \operatorname{poly}(n))). \tag{8}$$

Note that this does not yield a polynomial speedup over Alg, since the value of γ is not reduced by a constant.

The importance of this example lies in that while a qubit scaling of $O(s \log n)$ is, in many cases, easy to achieve (e.g. through storing an ordered list of $O(s)$ numbers in $\{1, \dots, O(\operatorname{poly} n)\}$), it however does not lead to a polynomial (albeit still asymptotic) speedup. A similar result can also be seen for a scaling of $O(s \log s)$. ■

Note that the strictness of the space requirement for QAlg to obtain a polynomial speedup comes from the premise of only having a quantum computer of size $M = cn$. Note that if in Example 2, we were given a quantum computer with $M = cn \log n$ qubits instead, a polynomial speedup would still be obtained. The strength of the speedup therefore critically depends what is considered a "natural" scaling of M relative to n. In many of the typical applications, the search space of typical classical algorithms (e.g. brute-force search) can be enumerated using $O(n)$ classical bits. Using amplitude amplification, one can then often obtain a quantum algorithm using $O(n)$ qubits that is usually polynomially (and often quadratically) faster than the corresponding classical algorithm [11]. In these cases, $M = cn$ is the natural scaling because if the scaling were to be relaxed to $M = cq(n)$ qubits with $q(n)$ being superlinear in n, then even for arbitrarily small c, the above quantum algorithm would require asymptotically fewer qubits than the hybrid algorithm, which would be inconsistent with the notion that M should be significantly smaller than the number of qubits required by a full quantum algorithm.

Example 3. We now show that the results of [1] also fit in this paradigm. We use the nomenclature of [1], and refer the interested reader to [1, 12] for further details.

In [1], Alg was taken to be the algorithm from [12] for the Promise-Ball-SAT problem. There, $P = (F, \mathbf{x}, r)$ comprises an n-variable 3SAT formula F, a trial assignment $\mathbf{x} \in \{0,1\}^n$ and a radius $r \in \{1, \ldots, n\}$. The effective problem size $s(P)$ was simply taken to be its radius r of P.

In [12], it was shown that there are two possible cases ($k = 2$) in the recursive algorithm. In the first case, two subproblems of radius $r - 1$ are created. In the second case, at most $t^2 2^\Delta$ subproblems of radius $r - \Delta$ are created, where $\Delta \in \mathbb{N}$ is a constant chosen below and $t = 3\Delta$.

Then, with the notation of this section, $l = t^2 2^\Delta$, $k = 2$, $l_1 = 2$, $l_2 = l$, and

$$C = \begin{pmatrix} 1 & \Delta \\ 1 & \Delta \\ 0 & \Delta \\ \vdots & \vdots \\ 0 & \Delta \end{pmatrix} . \tag{9}$$

This leads to a recursive runtime bound of

$$T(r) \leq \max\left(2T(r-1), t^2 2^\Delta T(r-\Delta)\right), \tag{10}$$

leading to a runtime of $T(r) = O((2t^{2/\Delta})^r \operatorname{poly}(n)) = O(2^{(1+\varepsilon)r} \operatorname{poly}(n))$, where $\varepsilon = 2\Delta^{-1}\log_2(3\Delta)$. Note that $\varepsilon \to 0$ as $\Delta \to \infty$.

Ref [1] then constructs a quantum algorithm solving the Promise-Ball-SAT problem in time $O(2^{\gamma_Q r} \operatorname{poly}(n))$ with $\gamma_Q = \log_2(3)/2 < 1$, and using at most $O(r\log(n/r) + r + \log n)$ qubits. Thus, Theorem 1 and the observation in Example 1 implies that Promise-Ball-SAT can be solved in time $O(\max(2^{(1+\varepsilon)r - \tilde{f}(c)n}, 2^{\gamma_Q r}) \operatorname{poly}(n))$ with $\tilde{f}(c) = \Theta(c/\log(1/c))$. Note that Δ can be chosen such that $\varepsilon < \tilde{f}(c)/2$. The results from [13], which reduce 3SAT to Promise-Ball-SAT, then imply that given a quantum computer with $M = cn$ qubits, n-variable 3SAT can be solved in time $O(2^{(\gamma - f(c))n} \operatorname{poly}(n))$, where $f(c) = \tilde{f}(c)/2$ and $\gamma = \log_2(4/3)$. This is a polynomial speedup of the 3SAT algorithm obtained in [12] (which in turn can be seen as a derandomised version of Schöning's algorithm [14]), which is the central result of [1]. ■

IV. EFFICIENT AND REVERSIBLE SET GENERATION

The previous section highlights the importance of the space-efficiency of the quantum algorithm used in Theorem 1. In many instances, this quantum algorithm require the storing and manipulation large sets, e.g. to keep track of changes to P. As observed in [1], this is in general a non-trivial task when constrained by limited memory.

In this section, we formulate a general process to space- and time-efficiently generate an encoding of a set in a reversible manner, designed to be compatible with the use of Theorem 1 to obtain a polynomial speedup using small quantum computers.

Note that most of the required subroutines can trivially be implemented space-efficiently if one assumes that ancillary memory registers can be erased at will. However, we naturally require our computations to be reversible. This is in general an issue, since turning non-reversible computations into reversible ones in the canonical fashion either introduces many ancillas, or can incur exponential overheads (see e.g. [15]), both of which are non-starters for our needs. Combining the seemingly competing requirements of reversibility, small memory requirements and computational efficiency for our purposes is non-trivial. To illustrate the problem (see also [1]), note that if sets $S = \{x_1, \ldots, x_r\} \subset \{1, \ldots, N\}$ are simply stored as ordered lists $|x_1\rangle \ldots |x_r\rangle$, the memory requirement of $O(r \log N)$ qubits is too large for a polynomial speedup if $r = O(s(P))$ and $N = O(n(P))$ (see Example 2). On the other hand, if sets are encoded as genuine sets (i.e., without storing any ordering of the elements in the set), the operation $|\{x_1, \ldots, x_{i-1}\}\rangle |x_i\rangle \mapsto |\{x_1, \ldots, x_i\}\rangle$ is non-reversible, because the information on which element was added last is lost. The naive way to make this reversible would be to first implement the operation $|\{x_1, \ldots, x_{i-1}\}\rangle |x_i\rangle \mapsto |\{x_1, \ldots, x_{i-1}\}\rangle |x_i\rangle |\{x_1, \ldots, x_i\}\rangle$ and then to uncompute the $|\{x_1, \ldots, x_{i-1}\}\rangle |x_i\rangle$ registers by applying the inverse of the circuit up to that point. It is easy to see however that this incurs a computational overhead of $O(2^i)$, which is too large. Indeed, suppose that $\mathsf{SetGen}_i |0\rangle = |\{x_1, \ldots, x_i\}\rangle$ and $\mathsf{Calculate}_i |0\rangle = |x_i\rangle$. Then, the naive (recursive) implementation of SetGen_i would be to first apply $\mathsf{Calculate}_i \mathsf{SetGen}_{i-1}$ to generate $|\{x_1, \ldots, x_{i-1}\}\rangle |x_i\rangle$, then implementing $|\{x_1, \ldots, x_{i-1}\}\rangle |x_i\rangle \mapsto |\{x_1, \ldots, x_{i-1}\}\rangle |x_i\rangle |\{x_1, \ldots, x_i\}\rangle$, and finally applying $(\mathsf{Calculate}_i \mathsf{SetGen}_{i-1})^{-1}$ to uncompute the ancillas. The resulting recursive runtime would be $|\mathsf{SetGen}_i| > 2|\mathsf{SetGen}_{i-1}|$, leading to an exponential gate count of $O(2^i)$.

In this section, we describe a general formalism to overcome this problem based on the ideas of [1]. Specifically, we will show how the above task can in fact be implemented with $O(r\log(N/r)+r+\log N)$ memory and $O(\mathrm{poly}(N))$ runtime. Of course, one of the key aspects of our implementation is the continuous uncomputation of any ancillas we introduce along the way once they are no longer needed. Once they are uncomputed (i.e., reset to a known initial state, say $|0\rangle$), they can be re-used for later computational steps. This allows us to keep the overall number of ancillas used low. The primary challenge is to do this in a way which avoids the exponential overhead mentioned above.

In Section IV A, we first introduce the data-structures which allow this suitable trade-off between space-efficiency and computational overhead of uncomputation. Specifically, we develop a space-efficient encoding of large sets which adds just enough ordering information to allow for efficient uncomputation. Afterwards, in Section IV B, we describe the general algorithm for generating such an encoding of a set, given only access to operations which generate single elements of the set.

All algorithms considered in the remainder of this section are classical and will be written as reversible circuits. For convenience, we also introduce the following notion of reversible implementation and note the subsequent trivial observation.

Definition 1. Let $q,l,g\in\mathbb{N}$, $\mathcal{X}\subset\{0,1\}^q$ and $f:\mathcal{X}\to\{0,1\}^q$ be injective. We say that f can be *implemented reversibly using l ancillas and g gates* if there exists a classical reversible circuit of at most g elementary gates which for all $x\in\mathcal{X}$ implements the operation

$$|x\rangle\,|0\rangle^{\otimes l}\mapsto|f(x)\rangle\,|0\rangle^{\otimes l}\,. \tag{11}$$

Proposition 1. *Let $q,t\in\mathbb{N}$, $\mathcal{X}_1,\dots,\mathcal{X}_t\subset\{0,1\}^q$ and $f_i:\mathcal{X}_i\to\{0,1\}^q$, $i=1,\dots,t$, be injective such that f_i can be implemented reversibly using l_i ancillas and g_i gates. Suppose that $f_i(\mathcal{X}_i)\subset X_{i+1}$ for $i=1,\dots,t-1$. Then, $f_t\circ f_{t-1}\circ\dots\circ f_1$ can be implemented reversibly using* $\max_{i=1,\dots,t} l_i$ *ancillas and $g_1+\dots+g_t$ gates.*

A. Efficient set encodings

In this section we describe how to efficiently encode sets in a way which allows for efficient uncomputation whilst maintaining reversibility. We first describe "basic" set encodings, which use little memory but by themselves do not allow for efficient uncomputation. After that, we describe an efficient encoding composed of multiple basic encodings that allows for efficient uncomputation.

Definition 2. Let $N,k\in\mathbb{N}$ be positive integers with $k\le N$, and let $S\subset\{1,\dots,N\}$ with $|S|=k$. Define the *basic encoding* $|\mathrm{Enc}_N\,S\rangle$ of S to be a sequence of $\lfloor k\log_2(N/k+1)\rfloor+2k$ trits set to

$$|\mathrm{Enc}_N\,S\rangle:=|(y_1)_2\rangle\,|2\rangle\,|(y_2-y_1)_2\rangle\,|2\rangle\dots|(y_k-y_{k-1})_2\rangle\,|2\rangle\,|0\rangle\dots|0\rangle\,, \tag{12}$$

where $S=\{y_1,\dots,y_k\}$ with $y_1<\dots<y_k$, and for a positive integer y, $|(y)_2\rangle$ denotes a sequence of $\lceil\log_2(y+1)\rceil$ trits encoding the binary representation of y on the $\{0,1\}$ subspace [16].

Example 4. Suppose $N=20$, $k=5$, and $S=\{6,7,10,15,17\}$. Then, $\lfloor k\log_2(N/k+1)\rfloor+2k=21$, and $y_1=6=110_2$, $y_2-y_1=1=1_2$, $y_3-y_2=3=11_2$, $y_4-y_3=5=101_2$, and $y_5-y_4=2=10_2$. Thus,

$$|\mathrm{Enc}_N\,S\rangle=|110212112101210200000\rangle\,. \tag{13}$$

■

To see that $\lfloor k\log_2(N/k+1)\rfloor+2k$ indeed suffice for $|\mathrm{Enc}_N\,S\rangle$, note that the number of trits required is

$$\begin{aligned}
&\lceil\log_2(y_1+1)\rceil+\lceil\log_2(y_2-y_1+1)\rceil+\dots+\lceil\log_2(y_k-y_{k-1}+1)\rceil+k &(14)\\
&\quad\le\log_2(y_1+1)+\log_2(y_2-y_1+1)+\dots+\log_2(y_k-y_{k-1}+1)+2k &(15)\\
&\quad\le k\log_2((y_k+k)/k)+2k &(16)\\
&\quad\le k\log_2(N/k+1)+2k, &(17)
\end{aligned}$$

where (16) follows from Jensen's inequality. Note that this is significantly less than the naive encoding of S as an ordered list, which uses $O(k\log N)$ bits.

In [1], it was shown how to perform basic set operations on $|\mathrm{Enc}_N\,S\rangle$.

Definition 3. Let N be a positive integer.

(i) For any positive integer $k \leq N$, let $\mathsf{Contains}_{N,k}$ be the operation that performs

$$\mathsf{Contains}_{N,k} \left|\mathrm{Enc}_N\, S\right\rangle \left|x\right\rangle \left|0\right\rangle = \left|\mathrm{Enc}_N\, S\right\rangle \left|x\right\rangle \left|x \in S?\right\rangle \tag{18}$$

for any $S \subset \{1, \dots, N\}$ with $|S| = k$, where the last bit on the right-hand side of (18) is 1 if $x \in S$ and 0 otherwise.

(ii) Let $\mathsf{Convert}_N$ be the operation that performs

$$\mathsf{Convert}_N \left|x\right\rangle \left|0\right\rangle = \left|x\right\rangle \left|\mathrm{Enc}_N \{x\}\right\rangle \tag{19}$$

for any $x \in \{1, \dots, N\}$.

(iii) For any positive integers k_1, k_2 with $k_1 + k_2 \leq N$, let $\mathsf{Union}_{N,k_1,k_2}$ be the operation that performs

$$\mathsf{Union}_{N,k_1,k_2} \left|\mathrm{Enc}_N\, S_1\right\rangle \left|\mathrm{Enc}_N\, S_2\right\rangle \left|0\right\rangle = \left|\mathrm{Enc}_N\, S_1\right\rangle \left|\mathrm{Enc}_N\, S_2\right\rangle \left|\mathrm{Enc}_N\, S_1 \cup S_2\right\rangle \tag{20}$$

for any disjoint $S_1, S_2 \subset \{1, \dots, N\}$ with $|S_1| = k_1$ and $|S_2| = k_2$.

Proposition 2 ([1], Lemma 2,5,6 in Supplemental Material)**.** *Let N be a positive integer. Then,*

(i) for any positive integer $k \leq N$, $\mathsf{Contains}_{N,k}$ can be implemented reversibly using $O(\log N)$ ancillas and $O(\mathrm{poly}(N))$ gates.

(ii) $\mathsf{Convert}_N$ can be implemented reversibly using $O(\log N)$ ancillas and $O(\mathrm{poly}(N))$ gates [17].

(iii) for any positive integers k_1, k_2 with $K = k_1 + k_2 \leq N$, $\mathsf{Union}_{N,k_1,k_2}$ can be implemented reversibly using $O(K \log(N/K) + K + \log N)$ ancillas and $O(\mathrm{poly}(N))$ gates.

Note in particular that in Proposition 2(i) and (iii), the runtime bound (i.e., the degree of the polynomial) does not depend on k or k_1, k_2, respectively. In fact, none of the operations depend on the set-sizes in any relevant way.

Although $\left|\mathrm{Enc}_N\, S\right\rangle$ is itself space-efficient, it does not allow for a set generation procedure that is simultaneously space- and time-efficient as well as reversible, for the reasons explained at the beginning of this section. We now define a memory-structure that allows for this task.

Definition 4. Let N, k be positive integers with $k \leq N$. Let $S \subset \{1, \dots, N\}$ with $|S| = k$, and let $Z = (x_1, \dots, x_k)$ be a permutation of the elements of S. Then, the *efficient encoding* $\left|\mathrm{EffEnc}_N\, Z\right\rangle$ of Z is definded as follows: suppose that k has binary representation $k = 2^{a_1} + \cdots + 2^{a_s}$ with integers $\lfloor \log_2 k \rfloor = a_1 > a_2 > \cdots > a_s \geq 0$. For $j = 1, \dots, s$, let $k_j := 2^{a_1} + \cdots + 2^{a_j}$. Then, $\left|\mathrm{EffEnc}_N\, Z\right\rangle$ is defined as

$$\left|\mathrm{EffEnc}_N\, Z\right\rangle := \left|\mathrm{Enc}_N \{x_1, \dots, x_{k_1}\}\right\rangle \left|\mathrm{Enc}_N \{x_{k_1+1}, \dots, x_{k_2}\}\right\rangle \dots \left|\mathrm{Enc}_N \{x_{k_{s-1}+1}, \dots, x_k\}\right\rangle. \tag{21}$$

Example 5. Suppose $k = 13 = 8 + 4 + 1$ and $Z = (x_1, \dots, x_{13})$. Then,

$$\left|\mathrm{EffEnc}_N\, Z\right\rangle = \left|\mathrm{Enc}_N \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}\right\rangle \left|\mathrm{Enc}_N \{x_9, x_{10}, x_{11}, x_{12}\}\right\rangle \left|\mathrm{Enc}_N \{x_{13}\}\right\rangle. \tag{22}$$

■

Proposition 3. *For any positive integers $k \leq N$ and distinct integers $x_1, \dots, x_k \in \{1, \dots, N\}$, $\left|\mathrm{EffEnc}_N\, Z\right\rangle$ comprises at most $\lfloor 2k \log_2(N/k + 1) \rfloor + 8k = O(k \log(N/k) + k)$ trits, where $Z = (x_1, \dots, x_k)$.*

Proof. For any $S' \subset \{1, \dots, N\}$, $\left|\mathrm{Enc}_N\, S'\right\rangle$ comprises at most $|S'| \log_2(N/|S'| + 1) + 2|S'|$ trits. Thus, the number of trits in $\left|\mathrm{EffEnc}_N\, Z\right\rangle$ is at most

$$\sum_{l=0}^{\lfloor \log_2 k \rfloor} \left(2^l \log_2 \frac{N + 2^l}{2^l} + 2^{l+1} \right) = \sum_{l=0}^{\lfloor \log_2 k \rfloor} 2^l \log_2 \frac{N + 2^l}{k} + \sum_{l=0}^{\lfloor \log_2 k \rfloor} 2^{l+1} + \sum_{l=0}^{\lfloor \log_2 k \rfloor} 2^l \log_2 \frac{k}{2^l} \tag{23}$$

$$\leq \sum_{l=0}^{\lfloor \log_2 k \rfloor} 2^l \log_2 \frac{N + k}{k} + \sum_{l=0}^{\lfloor \log_2 k \rfloor} 2^{l+1} + k \sum_{l=0}^{\lfloor \log_2 k \rfloor} \frac{2^l}{k} \log_2 \frac{k}{2^l} \tag{24}$$

$$\leq 2k \log_2 \left(\frac{N}{k} + 1 \right) + 4k + 2k \sum_{l=0}^{\lfloor \log_2 k \rfloor} \frac{1}{2^{\lceil \log_2 k \rceil - l}} (\lceil \log_2 k \rceil - l) \tag{25}$$

$$\leq 2k \log_2 \left(\frac{N}{k} + 1 \right) + 4k + 2k \sum_{j=0}^{\infty} \frac{j}{2^j} \tag{26}$$

$$\leq 2k \log_2 \left(\frac{N}{k} + 1 \right) + 8k, \tag{27}$$

which proves the claim. □

Note that since we always assume that we work with sets of known sizes, basic set operations can be "lifted" from $|\mathrm{Enc}_N\, S\rangle$ to $|\mathrm{EffEnc}_N\, Z\rangle$.

Definition 5. For any positive integers N, k with $k \le N$, let $\mathsf{EffContains}_{N,k}$ be the operation that performs

$$\mathsf{EffContains}_{N,k} \,|\mathrm{EffEnc}\, Z\rangle\, |x\rangle\, |0\rangle = |\mathrm{EffEnc}\, Z\rangle\, |x\rangle\, |x \in Z?\rangle \tag{28}$$

for any integer $x \in \{1, \dots, N\}$ and distinct integers $x_1, \dots, x_k \in \{1, \dots N\}$, where $Z = (x_1, \dots, x_k)$ and the last bit on the right-hand side of (28) is 1 if $x = x_j$ for some $j \in \{1, \dots, k\}$ and 0 otherwise.

Proposition 4. *For all positive integers N, k with $k \le N$, $\mathsf{EffContains}_{N,k}$ can be implemented reversibly using $O(\log N)$ ancillas and $O(\mathrm{poly}(N))$ gates.*

In particular, the runtime bound in Proposition 4 does not depend on k.

Proof of Proposition 4. This follows immediately from Proposition 2(i): Introduce $s \le \log_2 k = O(\log N)$ ancilla bits (where s is defined as in Definition 4), run $\mathsf{Contains}_{N,2^{a_j}}$ on $\left|\mathrm{Enc}_N\{x_{k_{j-1}+1}, \dots, x_{k_j}\}\right\rangle |x\rangle$ for all $j = 1, \dots, s$ (where $a_1, \dots, a_s$ and $k_1, \dots, k_s$ are defined as in Definition 4) such that the outcome is stored in the j^{th} ancilla, apply a logical OR over the s ancilla bits, and finally uncompute the s ancilla bits by running $\mathsf{Contains}_{N,2^{a_j}}^{-1}$ for $j = 1, \dots, N$. □

Note that if $N' > N$ are integers and $S \subset \{1, \dots, N\}$, then $|\mathrm{Enc}_N\, S\rangle$ and $|\mathrm{Enc}_{N'}\, S\rangle$ only differ in the number of additional $|0\rangle$'s at the end of the encoding. For the remainder of the paper, whenever the value of N is clear from context, we will drop the subindex N for simplicity of notation and simply write $|\mathrm{Enc}\, S\rangle$, $|\mathrm{EffEnc}\, Z\rangle$, $\mathsf{Contains}_k$, $\mathsf{Convert}$, Union_{k_1,k_2} and $\mathsf{EffContains}_k$ instead.

B. Efficient and reversible set generation

Suppose we want to generate a set $X(\nu) = \{x_1 \dots, x_r\} \subset \{1, \dots, N\}$ of size r from some input register $|\nu\rangle$. We assume that we can generate the elements iteratively, i.e. we have access to circuits that generate x_i from ν and $\{x_1, \dots, x_{i-1}\}$. As discussed in at the beginning of this section, simply using the basic encoding $|\mathrm{Enc}\, X\rangle$ and adding one elment x_i to $|\mathrm{Enc}\, X\rangle$ at a time is problematic, as this incurs problems with reversibility or exponential computational overheads. In this subsection, we show how this can be circumvented using the $|\mathrm{EffEnc}\, Z\rangle$ encoding from the previous subsection.

The intuition for why $|\mathrm{EffEnc}\, Z\rangle$, unlike $|\mathrm{Enc}\, X\rangle$, allows for time-efficient uncomputation is that by splitting X into smaller subsets, we can generate and uncompute these subsets more efficiently, thus avoiding the necessity of uncomputing the entire set for each newly added element, which can be seen as the reason for the exponential overhead in the naive reversible implementation given at the beginning of this section.

Theorem 2. *Let N, r be known positive integers with $r \le N$, and let $\mathcal{I}$ be a finite set. Let $X : \mathcal{I} \to \mathcal{P}_r([N])$, where $\mathcal{P}_r([N]) := \{S \subset \{1, \dots, N\}, |S| = r\}$. Suppose that for $i = 1, \dots r$, $\mathsf{Calculate}_i$ are reversible circuits such that for all $\nu \in \mathcal{I}$, there is a permutation $(x_1, \dots, x_r)$ of the elements of $X(\nu)$ such that for all $i = 1, \dots, r$,*

$$\mathsf{Calculate}_i \,|\nu\rangle\, |\mathrm{EffEnc}\, Z_{i-1}\rangle\, |0\rangle\, |0\rangle = |\nu\rangle\, |\mathrm{EffEnc}\, Z_{i-1}\rangle\, |x_i\rangle\, |0\rangle\,, \tag{29}$$

where $Z_i = (x_1, \dots, x_i)$, and the last register in (29) comprises at most A ancillas. Then, the operation

$$|\nu\rangle\, |0\rangle \mapsto |\nu\rangle\, |\mathrm{EffEnc}\, Z_r\rangle \tag{30}$$

can be implemented reversibly using $O(r \log(N/r) + r + \log N + A)$ ancillas, $O(r^2)$ calls to $\mathsf{Calculate}_i$ for some $i \in \{1, \dots, r\}$, and $O(\mathrm{poly}(N))$ additional gates.

Proof. For any positive integer i, write $g(i)$ to be the largest integer g such that 2^g divides i. Then, for all $i \in \{1 \dots, r-1\}$ and $l \in \{0, \dots, g(i)\}$ such that $i + 2^l \le r$, let $R_{i,l}$ be the operation that performs

$$R_{i,l}\, |\nu\rangle\, |\mathrm{EffEnc}\, Z_i\rangle\, |0\rangle = |\nu\rangle\, |\mathrm{EffEnc}\, Z_i\rangle\, |\mathrm{Enc}\{x_{i+1}, \dots x_{i+2^l}\}\rangle \tag{31}$$

for all $\nu \in \mathcal{I}$, where $x_1, \dots, x_r$ and $Z_1, \dots, Z_r$ are as in the statement of the theorem. Note that if $l \le g(i) - 1$, then

$$|\mathrm{EffEnc}\, Z_i\rangle\, |\mathrm{Enc}\{x_{i+1}, \dots, x_{i+2^l}\}\rangle = |\mathrm{EffEnc}\, Z_{i+2^l}\rangle\,. \tag{32}$$

We also define $R_{0,l}$ for all $l \leq \lfloor \log_2 r \rfloor$ to be the operation that performs

$$R_{0,l} |\nu\rangle |0\rangle = |\nu\rangle |\mathrm{Enc}\{x_1, \dots, x_{2^l}\}\rangle = |\nu\rangle |\mathrm{EffEnc}\, Z_{2^l}\rangle \tag{33}$$

for all $\nu \in \mathcal{I}$. Note that since $\mathsf{Convert}\, |x_i\rangle |0\rangle = |x_i\rangle |\mathrm{Enc}\{x_i\}\rangle$, a reversible implementation of $R_{i,l}$ can be obtained recursively via

$$R_{i,0} = \mathsf{Calculate}_{i+1}^{-1} \mathsf{Convert}\ \mathsf{Calculate}_{i+1} \tag{34}$$

for all $i \in \{0, \dots, r-1\}$, and

$$R_{i,l+1} = R_{i,l}^{-1} R_{i+2^l,l}^{-1} \mathsf{Union}_{2^l,2^l} R_{i+2^l,l} R_{i,l} \tag{35}$$

for all $i \in \{0, \dots, r-1\}$ and $l \in \{0, \dots, g(i)-1\}$ such that $i + 2^{l+1} \leq r$. Indeed, for $i \in \{1, \dots, r\}$ and $l \leq g(i) - 1$, (32) implies that

$$R_{i,l} |\nu\rangle |\mathrm{EffEnc}\, Z_i\rangle |0\rangle = |\nu\rangle |\mathrm{EffEnc}\, Z_{i+2^l}\rangle \,. \tag{36}$$

Hence, $R_{i+2^l,l} R_{i,l}$ maps $|\nu\rangle |\mathrm{EffEnc}\, Z_i\rangle |0\rangle |0\rangle |0\rangle$ to

$$|\nu\rangle |\mathrm{EffEnc}\, Z_i\rangle |\mathrm{Enc}\{x_{i+1}, \dots, x_{i+2^l}\}\rangle |\mathrm{Enc}\{x_{i+2^l+1}, \dots, x_{i+2^{l+1}}\}\rangle |0\rangle \,, \tag{37}$$

and $\mathsf{Union}_{2^l,2^l}$ maps the latter to

$$|\nu\rangle |\mathrm{EffEnc}\, Z_i\rangle |\mathrm{Enc}\{x_{i+1}, \dots, x_{i+2^l}\}\rangle |\mathrm{Enc}\{x_{i+2^l+1}, \dots, x_{i+2^{l+1}}\}\rangle |\mathrm{Enc}\{x_{i+1}, \dots, x_{i+2^{l+1}}\}\rangle \,. \tag{38}$$

Finally, $R_{i,l}^{-1} R_{i+2^l,l}^{-1}$ uncomputes the registers containing $|\mathrm{Enc}\{x_{i+1}, \dots, x_{i+2^l}\}\rangle |\mathrm{Enc}\{x_{i+2^l+1}, \dots, x_{i+2^{l+1}}\}\rangle$, proving Eq. (35) for $i \in \{1, \dots, r\}$. A similar argument also shows that (35) holds for $i = 0$.

Suppose that r has binary expansion $r = 2^{a_1} + 2^{a_2} + \dots + 2^{a_s}$, with integers $\lfloor \log_2 r \rfloor = a_1 > a_2 > \dots > a_s = g(r)$. For $j = 1, \dots, s$, define $r_j := 2^{a_1} + \dots + 2^{a_j}$, and

$$R_j := R_{r_{j-1},a_j} \cdots R_{r_1,a_2} R_{0,a_1} \,. \tag{39}$$

Note that $a_j \leq g(r_{j-1}) - 1$ for all $j \in \{2, \dots, s\}$, so (32) implies that (39) is well-defined. Note moreover that $R_j |\nu\rangle |0\rangle = |\nu\rangle \left|\mathrm{EffEnc}\, Z_{r_j}\right\rangle$ for all $j \in \{1, \dots, s\}$. In particular, R_s performs the desired operation (30).

It follows from (35) that each individual $R_{i,l}$ constitutes a reversible implementation using at most $O(r \log(N/r) + r + \log N + A)$ ancillas. Hence, Proposition 1 implies that R_s can be implemented reversibly using at most $O(r \log(N/r) + r + \log N + A)$ ancillas.

To bound the number of calls to $\mathsf{Calculate}_{i'}$ and additional gate count of R_s, let $L_{i,l}$ be the number of calls of $R_{i,l}$ to $\mathsf{Calculate}_{i'}$ for some $i' \in \{1, \dots, r\}$, and let $M_{i,l}$ be the number of additional gates of $R_{i,l}$, respectively. Let $L_l = \max\{L_{i,l} : i \in \{0, \dots, r-1\}, l \leq g(i), i + 2^l \leq r\}$ and $M_l = \max\{M_{i,l} : i \in \{0, \dots, r-1\}, l \leq g(i), i + 2^l \leq r\}$. Eq. (34) and (35) clearly imply $L_l = 2 \cdot 4^l$. Moreover, since $\mathsf{Union}_{2^l,2^l}$ can be implemented using at most $p(N)$ gates, where p is a polynomial independent of l, it follows from (35) that $M_{l+1} \leq 4M_l + p(N)$, implying $M_l = O(4^l \operatorname{poly}(N))$. Thus, R_s uses at most

$$L_{a_1} + \dots + L_{a_s} \leq 2(1 + 4 + \dots + 4^{\lfloor \log_2 r \rfloor}) = O(r^2) \tag{40}$$

calls to $\mathsf{Calculate}_{i'}$ for some $i' \in \{1, \dots, r\}$, and

$$M_{a_1} + \dots + M_{a_s} = O((1 + 4 + \dots + 4^{\lfloor \log_2 r \rfloor}) \operatorname{poly}(N)) = O(\operatorname{poly}(N)) \tag{41}$$

additional gates. □

Theorem 2 generates the efficient encoding $|\mathrm{EffEnc}\, Z_r\rangle$ of the set $X(\nu)$ instead of the simple encoding $|\mathrm{Enc}\, X(\nu)\rangle$. For most applications, the former is sufficient, since the value of r is known and simple set queries for checking properties of $X(\nu)$ (e.g. checking if $X(\nu)$ contains certain elements) are generally just as simple to implement reversibly using $|\mathrm{EffEnc}\, Z_r\rangle$ as with $|\mathrm{Enc}\, X(\nu)\rangle$ (see Proposition 4). We remark however that $|\mathrm{EffEnc}\, Z_r\rangle$ can be converted to $|\mathrm{Enc}\, X(\nu)\rangle$ using a sequence of calls to Union (for appropriate set sizes) and uncomputations. Since at most $O(\log r)$ union operations are required, is easy to see that this can be implemented with $O(\operatorname{poly}(N))$ calls to $R_{i,l}$ (as defined in the proof of Theorem 2) for suitable values of i, l, and $O(\operatorname{poly}(N))$ additional gates.

Corollary 1. *With the same notation as in Theorem 2, the operation*

$$|\nu\rangle |0\rangle \mapsto |\nu\rangle |\mathrm{Enc}\, X(\nu)\rangle \tag{42}$$

can be implemented reversibly using $O(r \log(N/r) + r + \log N + A)$ ancillas, $O(\operatorname{poly}(N))$ calls to $\mathsf{Calculate}_i$, and $O(\operatorname{poly}(N))$ additional gates.

V. SPEEDUP OF EPPSTEIN'S ALGORITHM

In this section, we provide an example of how to use the toolkit − the divide-and-conquer hybrid approach from Section III and the set-generation procedure from Section IV − to polynomially speed up Eppstein's algorithm [5] for the cubic Hamiltonian cycle problem using a small quantum computer. The problem asks whether a given cubic graph $G = (V, E)$ has a Hamiltonian cycle, i.e. a cycle going through every vertex exactly once.

A. Eppstein's algorithm

In this section, we review Eppstein's classical algorithm for solving this problem in time $O(2^{n(G)/3}\,\text{poly}(n(G)))$, where $n(G)$ denotes the number of vertices of a graph G.

Note first of all that, without loss of generality, one can assume that the graph is triangle-free, since triangles can be removed by merging the three vertices of a triangle into a single vertex.

Eppstein's algorithm introduces the concept of "forced" edges that a Hamiltonian cycle has to contain. In other words, if an edge is forced, we are only looking for Hamiltonian cycles which contain that edge.

Definition 6. Let $G = (V, E)$ be a simple triangle-free graph with maximum degree at most 3, and $F \subset E$. Then, the *forced cubic Hamiltonian cycle* (FCHC) problem asks whether G has a Hamiltonian cycle containing all edges in F. We call edges in F *forced*, and edges in $E\backslash F$ *unforced*. We call (G, F) an *FCHC instance.*

Roughly speaking, Eppstein's algorithm solves FCHC by recursively selecting an unforced edge and creating two subinstances by either adding that edge to F or removing it from G. In both cases, the fact that G is cubic induces additional edges to be either added to F or to be removed.

The details of Eppstein's algorithm are given in Alg. 2. The formulation of the algorithm here has been modified from Eppstein's original formulation in several places. In particular, we adapted it to the Hamiltonian cycle problem (instead of the travelling salesman problem [18]) and solve it as a decision problem (rather than finding a Hamiltonian cycle). We also made several smaller changes to make the transition the the quantum algorithm later easier. For clarity, and to make this section self-contained, Eppstein will in the following always refer to the algorithm in Alg. 2 instead of the original formulation of this algorithm in [5].

We first introduce a few important concepts.

Definition 7. An FCHC instance (G, F) is called *trivial-reduction-free* if

(i) G does not contain any vertices of degree two with unforced incident edges,

(ii) G does not contain any vertices of degree three with exactly two forced edges, and

(iii) G does not contain any cycles of four unforced edges such that two of its opposite vertices are incident to a forced edge and at least one of the other vertices is incident to an unforced edge that is not part of the cycle.

In other words, an FCHC instance is trivial-reduction-free if and only if none of the conditions of step 1 of Eppstein apply.

Definition 8. Let $G = (V, E)$ be a simple, triangle-free graph with maximum degree at most 3, ω be a cycle of four edges, and $F \subset E$. We say that ω *unforced-isolated w.r.t.* F if all edges of ω are in $E\backslash F$, and each of the four vertices of ω is incident to an edge in F. Moreover, denote by $C(G, F)$ the set of $4-$cycles in G which are unforced-isolated w.r.t. F.

Note that if (G, F) is trivial-reduction-free, then all unforced 4-cycles are unforced-isolated w.r.t. F. The correctness of Alg. 2 is given by the following proposition.

Proposition 5. *Let (G, F) be a trivial-reduction-free FCHC instance. Suppose that G has only vertices of degree 2 or 3, and that no three edges in F meet in a single vertex. Suppose moreover that $G\backslash F$ is a collection of disjoint 4-cycles and isolated vertices. Then, G has a Hamiltonian cycle containing all edges in F if and only if G is connected.*

Proof. The "only if" direction is trivial. Assume that $G = (V, E)$ is connected. Note first of all that all vertices outside of $C(G, F)$ have degree 2 and both their incident edges are in F.

For each 4-cycle $\omega \in C(G, F)$, let $h_1(\omega), h_2(\omega)$ be two opposite edges in ω and $h_3(\omega), h_4(\omega)$ be the other two edges in ω. Let $F_1 := F \cup \{h_1(\omega), h_2(\omega) : \omega \in C(G, F)\}$. Note that every vertex in G is incident to an edge in F_1 and that every vertex in $G_1 = (V, F_1)$ has degree two. Thus, G_1 is a collection of cycles. Consider the graph G_2 whose vertices are the connected components of G_1, and two connected components H_1, H_2 of G_1 are joined by an edge in G_2 iff there

```
EPPSTEIN($G, F$):
  1. Repeat the following steps ("trivial reductions") until none of the conditions apply
     a. If $G$ contains a vertex with degree two with at least one unforced incident edge, add all its incident edges to $F$.
     b. If $G$ contains a vertex with degree three with exactly two forced edges, remove the unforced edge.
     c. If $G$ contains a cycle of four unforced edges such that two of its opposite vertices are each incident to a forced edge
        and at least one of the other vertices is incident to an unforced edge that is not part of the cycle, then add to $F$ all
        non-cycle edges that are incident to a vertex of the cycle.
  2. Check if any of the following conditions ("terminal conditions") apply
     a. If $G$ contains a vertex of degree 0 or 1, or if $F$ contains three edges meeting at a vertex, return *false*.
     b. If $G \backslash F$ is a collection of disjoint 4-cycles and isolated vertices
         i. If $G$ is disconnected, return *false*.
        ii. Otherwise, return *true*.
     c. If $F$ contains a non-Hamiltonian cycle, return *false*.
  3. Choose an edge $yz$ according to the following cases:
     a. If $G \backslash F$ contains a 4-cycle, exactly two vertices of which are incident to an edge in $F$, let $y$ be one of the other two
        vertices of the cycle and let $yz$ be an edge of $G \backslash F$ that does not belong to the cycle.
     b. If there is no such 4-cycle, but $F$ is nonempty, let $xy$ be any edge in $F$ and $yz$ be an adjacent edge in $G \backslash F$ such that
        $yz$ is not part of an isolated 4-cycle in $G \backslash F$.
     c. Otherwise, let $yz$ be any edge in $G$ that is not part of an isolated 4-cycle in $G \backslash F$.
  4. Call EPPSTEIN($G, F \cup \{yz\}$).
  5. Call EPPSTEIN($G \backslash \{yz\}, F$).
  6. Return the disjunction (logical OR) of steps 4 and 5.
```

ALG 2: Eppstein's algorithm (modified)

exists a 4-cycle $\omega \in C(G, F)$ which "separates" H_1, H_2 in G, i.e., $h_1(\omega) \in H_1$ and $h_2(\omega) \in H_2$ or vice-versa. Note that since G is connected, so is G_2. Note moreover that if H_1 and H_2 are adjacent in G_2 and separated by $\omega \in C(G, F)$ in G, replacing $h_1(\omega), h_2(\omega)$ with $h_3(\omega), h_4(\omega)$ in G_1 would result in replacing the two disconnected cycles H_1, H_2 by a single cycle going through the same vertices. Thus, consider a spanning tree of G_2 and let $G_1' = (V, F_1')$ be the graph obtained from G_1 by replacing $h_1(\omega), h_2(\omega)$ with $h_3(\omega), h_4(\omega)$ for all ω corresponding to an edge in the spanning tree. Then, by the previous observation, G_1' is connected. Moreover, since $F \subset F_1'$ and every vertex has degree 2 in G_1', it follows that G_1' is a Hamiltonian cycle containing all edges in F. □

Note in particular that Proposition 5 implies that step 2c of EPPSTEIN is in fact unneccessary and the algorithm still performs correctly if that step is omitted. We include that step nevertheless, since the early termination of these instances simplifies the runtime analysis in Appendix A.

The main idea of bounding the runime of EPPSTEIN is to introduce a "problem size metric" defined as follows.

Definition 9. For an FCHC instance (G, F), let $s(G, F) := \max(n(G) - |F| - |C(G, F)|, 0)$. We call $s(G, F)$ the *size* of (G, F).

Proposition 6. *Let (G, F) be an FCHC instance such that no three edges in F meet at a vertex. Suppose moreover that F is not a collection of cycles. Then, $n(G) - |F| - |C(G, F)| > 0$.*

Proof. First of all, note that the general case can be reduced to the special case of $C(G, F) = \emptyset$ by adding one edge of each $\omega \in C(G, F)$ to F. Suppose now that $C(G, F) = \emptyset$, and let $G' = (V, F)$, where V is the set of vertices of G. Then, every vertex has degree at most 2 in G'. Moreover, since F is not a collection of cycles, at least one vertex has degree < 2 in G'. Hence,

$$2|F| = \sum_{v \in V} \deg_{G'} v < 2|V| \tag{43}$$

and hence $n(G) - |F| > 0$. □

It can be shown [5] that every application of steps 3–6 creates two FCHC instances (G_1, F_1) and (G_2, F_2) such that $s(G_1, F_1), s(G_2, F_2) \leq s(G, F) - 3$ or $s(G_1, F_1) \leq s(G, F) - 2$ and $s(G_2, F_2) \leq s(G, F) - 5$. This leads to a recursive runtime bound of $T(s(G, F))$, where $T(s) = \max(2T(s-3), T(s-2)+T(s-5)) + O(\mathrm{poly}(n(G)))$, leading to $T(s) = O(2^{s/3}\,\mathrm{poly}(n(G)))$. In particular, Eppstein$(G, \emptyset)$ solves the cubic Hamiltonian cycle problem in a runtime of $O(2^{n(G)/3}\,\mathrm{poly}(n(G)))$. We provide the details of this runtime analysis in the Appendix A.

B. Quantum improvement of Eppstein using small quantum computer

Using Theorem 1, the main part of obtaining a speedup is to provide a quantum speedup of Eppstein using few qubits.

Theorem 3. *There exists a quantum algorithm that, for any FCHC instance (G, F), decides FCHC in a runtime of $O(2^{s/4}\,\mathrm{poly}(n))$ using $O(s \log(n/s) + s + \log n)$ qubits, where $s = s(G, F)$ and $n = n(G)$.*

We will prove this in Section V C. Theorems 1 and 3 immediately imply an improvement to Eppstein's algorithm.

Theorem 4. *Let $c > 0$ be an arbitrary constant. Then, given a quantum computer with $M = cn$ qubits, there exists a hybrid quantum-classical algorithm that solves the cubic Hamiltonian cycle problem for n-vertex graphs in runtime $O(2^{(1/3-f(c))n}\,\mathrm{poly}(n))$, where $f(c) > 0$.*

Note that with Theorem 3, we have a quantum algorithm that satisfies all the criteria to apply Theorem 1. However, to make things fully rigorous, we also need to deal with the fact that Alg. 2 includes initial rewritings and reductions, and as such is not immediately a special case of Alg. 1. This is a minor technicality, which we resolve as follows for completeness.

Proof of Theorem 4. Note first of all that it is sufficient to prove that one can solve the FCHC problem in a runtime of $O(\max(2^{\gamma s(G,F) - f(c) n(G)}, 2^{\gamma_Q s(G,F)})\,\mathrm{poly}(n(G)))$, where $\gamma = 1/3$ and $\gamma_Q = 1/4$. Next, note that it is sufficient to do this only for trivial-reduction-free FCHC instances, since the others can be reduced to the trivial-reduction-free case by one application of step 1 of Eppstein.

Thus, with the notation of Section III, let $\mathcal{P}$ be the set of all trivial-reduction-free FCHC instances. Then Alg is given as follows: Trivial$(P) = 1$ if any of the terminal conditions in step 2 of Eppstein apply, and $f(P)$ is the value returned in step 2. Moreover, $l = 2$ and R_1 and R_2 are given by first selecting an edge yz according to step 3 of Eppstein, forcing and deleting it, respectively, followed by performing all possible trivial reductions (i.e., step 1 of Eppstein). Note that indeed, R_1 and R_2 map trivial-reduction-free FCHC instances to trivial-reduction-free FCHC instances. Finally, the runtime analysis of Eppstein (see Proposition 10 in Appendix A) gives $k = 3$, $l_1 = l_2 = l_3 = 2$ and

$$C = \begin{pmatrix} 3 & 2 & 5 \\ 3 & 5 & 2 \end{pmatrix}. \tag{44}$$

The result now follows from Theorem 1. □

C. Proof of Theorem 3

To prove Theorem 3, note first of all that we can without loss of generality assume that (G, F) is trivial-reduction-free, because if it is not, G and F can be classically pre-processed by repeated applications of step 1 of Eppstein. Note that $s(G, F)$ is non-increasing under this step.

The recursive steps 4 and 5 in Eppstein yield a binary recursion tree, with the two branches corresponding to either step 4 or 5. More formally, consider the binary rooted tree defined as follows.

Definition 10. Let (G, F) be an FCHC instance. Define RecTree(G, F) to the binary rooted tree constructed as follows: the root of the tree is a vertex labelled (G, F). Then, given a vertex $(\tilde{G}, \tilde{F})$, if Eppstein$(\tilde{G}, \tilde{F})$ terminates in step 2, it becomes a leaf of the tree. Otherwise, if Eppstein$(\tilde{G}, \tilde{F})$ calls Eppstein(G_1, F_1) and Eppstein(G_2, F_2) in steps 4 and 5, respectively, create two children of $(\tilde{G}, \tilde{F})$ labelled (G_1, F_1) and (G_2, F_2), respectively.

Moreover, let $\tau(G, F)$ be the number of edges that are forced or deleted in step 1 of Eppstein(G, F), before the algorithm moves on.

Intuitively, RecTree(G, F) is the tree of FCHC instances explored by steps 4 and 5 of Alg. 2. Note that by Proposition 10 in Appendix A, RecTree(G, F) has depth at most $s(G, F)/2$. We first show that in a "good" branch of the recursion tree, a total of at most $O(s(G, F))$ trivial reductions are performed.

Proposition 7. *Let (G,F) be trivial-reduction-free and let $(G,F)=(G_1,F_1),(G_2,F_2),\dots,(G_l,F_l)$ be a path in* RecTree(G,F)*, such that* Eppstein(G_l,F_l) *returns true in step 2. Then,*

$$\sum_{j=1}^{l}\tau(G_j,F_j)\leq 4s(G,F). \tag{45}$$

Proof. First, note that no edge which is part of some $\omega\in C(G,F)$ will be forced at any point of the algorithm. On the other hand, every Hamiltonian cycle contains exactly two (opposite) edges of each $\omega\in C(G,F)$. Hence, along the path from (G_1,F_1) to (G_l,F_l), a total of at most $n(G)-|F|-2|C(G,F)|\leq s(G,F)$ edges will be forced. In particular, at most $s(G,F)$ edges will be forced in step 1a or 1c of Eppstein(G_j,F_j) over all $j=1,\dots,l$. As for the number of edges deleted in step 1b, note that since (G,F) is trivial-reduction-free, every deletion of an edge in step 1b of Eppstein(G_j,F_j) for some $j\in\{1,\dots,l\}$ is induced by an additional edge being forced. Each such edge can induce at most two edges being deleted in step 1b. Therefore, at most $2(s(G,F)+s(G,F)/2)=3s(G,F)$ edges are deleted in step 1b of Eppstein(G_j,F_j) over all $j=1,\dots,l$. □

The quantum algorithm we construct is essentially a quantum version of a non-recursive variant of Eppstein which proceeds by successively forcing and deleting edges according to a given input of a suitable search space. Note that G and F are classical input parameters and as such, the quantum circuit may depend on G and F. We will not actively modify G or F in the quantum algorithm. Instead, the removal and forcing of additional edges are done by quantumly storing encodings of a set $X\subset\{1,\dots,3|E|\}$, where for $e\in\{1,\dots,|E|\}$, $e\in X$ means that edge number e is forced, and $e+|E|\in X$ means that edge number e has been removed (we assume that all edges and vertices are enumerated in a pre-specified order, i.e., with some abuse of notation [19], $E=\{1,\dots,|E|\}$ and $V=\{1,\dots,|V|\}$). For convenience we also introduce dummy variables which correspond to the values $2|E|+1,\dots,3|E|$.

Before moving to the quantum algorithm, we first show how to perform Eppstein non-recursively. By Proposition 7 and Proposition 10 in Appendix A, a given path leading to an accepting leaf of the recursion tree RecTree(G,F) adds at most $r:=\lfloor s(G,F)/2\rfloor+4s(G,F)$ elements to X, corresponding to at most $4s(G,F)$ edges deleted or forced in step 1 (Proposition 7) of Alg. 2, and at most $s(G,F)/2$ in steps 4 or 5 (Proposition 10). Thus, a sequence of r binary variables $\nu_1,\dots,\nu_r\in\{0,1\}$ suffice to enumerate all relevant leafs of RecTree(G,F), where each variable ν_i corresponds to an edge deleted or forced in step 1, step 4 or step 5. More precisely, the value of ν_i is ignored if the i^{th} element added to X is through a trivial reduction (i.e., in step 1 of Eppstein), otherwise the value of ν_i specifies if the edge chosen in step 3 is forced (step 4) or removed (step 5). Thus, Eppstein induces a mapping from $\mathcal{I}:=\{0,1\}^r\to\mathcal{P}_r([3|E|])$, $\vec{\nu}:=(\nu_1,\dots,\nu_r)\mapsto X=X(\vec{\nu})$.

Note that although the same task could be achieved by only introducing $\lfloor s(G,F)/2\rfloor$ binary variables instead of r, the additional $4s(G,F)$ variables ensure that at any given point of the implementation, X has a pre-determined size, which would not be the case otherwise (indeed, Proposition 7 only provides an upper bound to the number of trivial reductions overall but these are generally distributed in a previously unknown way). This will be important when we later want to use the results of Section IV.

The non-recursive (classical) variant of Eppstein is given in Alg. 3. Write $X=F'\cup D$, where $F'=\{e\in\{1,\dots,|E|\}:e\in X\}$ and $D=\{e\in\{1,\dots,|E|\}:e+|E|\in X\}$ are the edges which would be forced and deleted in either of steps 1, 4 or 5 of Eppstein(G,F), respectively. The algorithm first computes $X(\vec{\nu})$ from $\vec{\nu}$ (we call this operation Reduce), and then checks the conditions of step 2 for the FCHC instance $(G\backslash D,F\cup F')$ and returns the corresponding value. We call the second step Check. For completeness, Check also returns false if none of the conditions of step 2 apply (note that if this is the case, Propostions 5 and 7 imply that $\vec{\nu}$ corresponds to a branch that does not find a Hamiltonian cycle). The full non-recursive version Eppstein simply goes through all 2^r values of $\vec{\nu}$, yielding a runtime of $O(2^r\operatorname{poly}(n(G)))$. Alternatively, by picking $\vec{\nu}\in\mathcal{I}$ uniformly at random, an expected rutnime of $O(2^r/2^t\operatorname{poly}(n(G)))=O(2^{s/2}\operatorname{poly}(n(G)))$ can be achieved.

Note that we have omitted checking the conditions of step 2c of Eppstein in the non-recursive formulation, as it does not affect the correctness of the algorithm (see also remark after Proposition 5). Indeed, for a "good" branch (i.e., a value of $\vec{\nu}$ that corresponds to finding a Hamiltonian cycle), the condition of step 2c of Eppstein is never fulfilled, whereas Proposition 5 ensures that if $F\cup F'$ contains a non-Hamiltonian cycle, Check will return false. Instead of checking the condition of step 2c of Eppstein, the condition in line 34 of Alg. 3 has been modified to account for the case when F is a collection of (non-Hamiltonian) cycles, in which case step 3b of Eppstein cannot be applied.

To turn Alg. 3 into a quantum algorithm, we first show that Reduce and Check can be performed *reversibly* in polynomial time, and with only $O(s\log(n/s)+s+\log n)$ bits, where $s=s(G,F)$ and $n=n(G)$, and we assume that Check simply writes the result on a single output bit. Note that throughout NonRecursiveEppstein(G,F), G and F are never modified. As such, it is sufficient to write Reduce and Check as reversible circuits Reduce$_{G,F}$

```
procedure NonRecursiveEppstein($G, F$)
    $s := n(G) - |F| - |C(G,F)|$, $r := \lfloor s/2 \rfloor + 4s$
    for all $\vec{\nu} \in \{0,1\}^r$ do
        $X :=$ Reduce($G, F, \vec{\nu}$)
        $h :=$ Check($G, F, X$)
        if $h = 1$ then
            return true
        end if
    end for
    return false
end procedure

procedure Reduce($G, F, \vec{\nu} = (\nu_1, \dots, \nu_r)$)
    $X := \emptyset$
    for $i = 1, \dots, r$ do
        $x :=$ Calculate($G, F, X, \nu_i$)
        $X := X \cup \{x\}$
    end for
    return $X$
end procedure

procedure Calculate($G = (V, E), F, X, \nu$)
    $F' := \{e \in \{1, \dots, |E| : e \in X\}$, $D := \{e \in \{1, \dots, |E| : e + |E| \in X\}$, $G' := G \backslash D$
    if $G'$ contains a vertex with degree 2 with at least one edge in $E \backslash (F \cup F' \cup D)$ then
        $e :=$ one of the edges in $E \backslash (F \cup F' \cup D)$ incident to that vertex, $a := 0$
    else if $G'$ contains a vertex with degree 3 with exactly two edges in $F \cup F'$ then
        $e :=$ the third edge incident to that vertex, $a := 1$
    else if $G' \backslash (F \cup F')$ contains a cycle of 4 edges with two of its opposite vertices being incident to an edge in $F \cup F'$ and one of the other two vertices being incident to a non-cycle edge in $E \backslash (F \cup F' \cup D)$ then
        $e :=$ that non-cycle edge in $E \backslash (F \cup F' \cup D)$, $a := 0$
    else if $G'$ contains a vertex of degree 0 or 1, or if $F \cup F'$ contains three edges meeting at a vertex, or if $G' \backslash (F \cup F')$ is a collection of disjoint 4-cycles and isolated vertices then
        $e := i + 2|E|$, $a := 0$
    else if $G' \backslash (F \cup F')$ contains a 4-cycle, two vertices of which are incident to an edge in $F \cup F'$ then
        $e :=$ an edge in $G' \backslash (F \cup F')$ incident to one of the other two vertices and which does not belong to the cycle, $a := \nu$
    else if $F \cup F'$ is nonempty and not a collection of cycles then
        $e :=$ any edge in $E \backslash (F \cup F' \cup D)$ that is adjacent to an edge in $F \cup F'$, $a := \nu$
    else
        $e :=$ any edge in $G'$ that is not part of a 4-cycle in $G' \backslash (F \cup F')$, $a := \nu$
    end if
    return $e + a|E|$
end procedure

procedure Check($G = (V, E), F, X$)
    $F' := \{e \in \{1, \dots, |E| : e \in X\}$, $D := \{e \in \{1, \dots, |E| : e + |E| \in X\}$, $G' := (V, E \backslash D)$
    if $G'$ contains a vertex of degree 1 then
        return false
    else if $F \cup F'$ contains three edges meeting at a vertex then
        return false
    else if $G \backslash (F \cup F')$ is not a collection of disjoint 4-cycles and isolated vertices then
        return false
    else if $G$ is disconnected then
        return false
    end if
    return true
end procedure
```

ALG 3: Non-recursive variant of Eppstein. Note that G and F are never modified, and that in the Reduce subroutine, $|X| = i$ after each cycle of the loop.

and $\textsc{Check}_{G,F}$, which depend on G and F, and which perform

$$|\vec{\nu}\rangle\,|0\rangle\,|0\rangle\,|0\rangle\,|0\rangle \xrightarrow{\textsc{Reduce}_{G,F}} |\vec{\nu}\rangle\,|\mathrm{EffEnc}\,Z(\vec{\nu})\rangle\,|0\rangle\,|0\rangle \xrightarrow{\textsc{Check}_{G,F}} |\vec{\nu}\rangle\,|\mathrm{EffEnc}\,Z(\vec{\nu})\rangle\,|h(Z(\vec{\nu}))\rangle\,|0\rangle\,, \tag{46}$$

where $Z(\vec{\nu}) = (x_1, \ldots, x_r)$, $x_1, \ldots, x_r$ are the elements of $X(\vec{\nu})$ in the order in which they are added to X in $\textsc{Reduce}$, $h(Z(\vec{\nu}))$ is the output bit returned by $\textsc{Check}(G, F, X(\vec{\nu}))$, and the last register comprises at most $O(s\log(n/s) + s + \log n)$ ancilla bits.

Next, we turn this into a quantum process by replacing every elementary reversible operation by its corresponding quantum operation. The final step is to use amplitude amplification [20] (or alternatively fixed point search [21]) to quantumly search for the value $|h(Z(\vec{\nu}))\rangle = |1\rangle$ on the output register. Note that since t of the r input bits are irrelevant, the dimension of the target space, if a Hamiltonian cycle exists, is at least 2^t. Thus, fixed point search requires $O(\sqrt{2^r/2^t}) = O(2^{s/4})$ repetitions of $\textsc{Reduce}_{G,F}$ and $\textsc{Check}_{G,F}$.

It thus only remains to show that for any given trivial-reduction-free FCHC instance (G, F), both $\textsc{Reduce}_{G,F}$ and $\textsc{Check}_{G,F}$ can be performed reversibly in polynomial time and with only $O(s\log(n/s) + s + \log n)$ bits, where $s = s(G, F)$ and $n = n(G)$.

1. *Reversible space-efficient implementation of* $\textsc{Reduce}_{G,F}$

The basic idea of the $\textsc{Reduce}_{G,F}$ algorithm is to reversibly generate an efficient encoding of $X(\vec{\nu}) \subset \{1, \ldots, 3|E|\}$ from $|\vec{\nu}\rangle$ using Theorem 2 by specifying suitable reversible $\mathsf{Calculate}_i$ operations. For each value of $i = 1\ldots, r$, let $\mathsf{Calculate}_{G,F,i}$ be the operation that finds the next edge e_i and an *action bit* $a_i \in \{0, 1\}$ following the implementation of $\textsc{Calculate}$ in Alg. 3. That implementation in turn follows the implementation of $\textsc{Eppstein}$, i.e., an edge e_i is selected according to the rules of step 1 and 3, and $a_i = 0$ if e_i will be forced and $a_i = 1$ if e_i will be removed. If no more edges are forced or deleted (i.e., one of the terminal conditions of step 2a or 2b of $\textsc{Eppstein}$ apply), we set e_i to be a dummy variable $e_i = 2|E| + i$ and $a_i = 0$.

The task then becomes to reversibly implement the operations

$$\mathsf{Calculate}_{G,F,1}\,|\nu_1\rangle\,|0\rangle = |\nu_1\rangle\,|x_1\rangle \tag{47}$$

and

$$\mathsf{Calculate}_{G,F,i}\,|\nu_i\rangle\,|\mathrm{EffEnc}\,Z_{i-1}\rangle\,|0\rangle = |\nu_i\rangle\,|\mathrm{EffEnc}\,Z_{i-1}\rangle\,|x_i\rangle \tag{48}$$

for $i = 2, \ldots, r$, where $x_i = e_i + a_i|E|$ and $Z_i = (x_1, \ldots, x_i)$. Note that by introducing the dummy variables, we ensure that for any $i \in \{1, \ldots, r\}$, $X_i = \{x_1, \ldots, x_i\}$ has exactly i elements, even if no edges have been forced or deleted in some of the steps.

The primary challenge of the implementation is to maintain reversibility and at the same time use few ancillas. For this, it is important that any ancillas used are as soon as possible reset to $|0\rangle$ in order to avoid accumulating unnecessary junk bits.

Proposition 8. *Let (G, F) be a trivial-reduction-free FCHC instance and $i \in \{1, \ldots, r\}$, where $r = \lfloor s(G, F)/2\rfloor + 4s(G, F)$. Then, the operation $\mathsf{Calculate}_{G,F,i}$ defined by Eq. (47) and (48) can be implemented reversibly using $O(\log n(G))$ ancillas and $O(\mathrm{poly}(n(G)))$ gates.*

Proof. The basic idea to maintain reversibility is to introduce a counter, initially set to 0, which is increased once suitable values for e_i and a_i have been found. Then, by controlling all operations on that counter being 0, we ensure that no further edges are selected once an edge and action bit has been found, and hence that only one edge and action bit is selected.

More precisely, we introduce a counter from 0 to 7, which we call the *flag counter*, and denote it by $\mathcal{FC}$. We also introduce an additional ancilla bit, which we call the *flag bit*, and denote it by $\mathcal{FB}$. Both are initially set to zero. We denote the register of $O(\log n)$ bits containing the value of e_i as $\mathfrak{e}$ and the register containing the value of a_i as $\mathfrak{a}$. We assume that $\mathfrak{e}$ and $\mathfrak{a}$ are both initially set to zero.

For clarity of notation, we will only cover the case $i \geq 2$ here. The case $i = 1$ is fully analogous, but without the $|\mathrm{EffEnc}\,Z_{i-1}\rangle$ register and ignoring any operations involving it. As before, we write $Z_{i-1} = (x_1, \ldots, x_{i-1})$, $X_{i-1} = \{x_1, \ldots, x_{i-1}\}$, $F' = \{e \in \{1, \ldots, |E|\} : e \in X_{i-1}\}$ and $D = \{e \in \{1, \ldots, |E|\} : e + |E| \in X_{i-1}\}$.

The basic building block of the implementation consists of a *controlled check-and-select* (CCS) operation illustrated in Fig. 2. We will implement seven different check-and-select operations, each corresponding to lines 24–25, 26–27, 28–29, 30–31, 32–33, 34–35, and 36–37, of Alg. 3, respectively. For example, the check-and-select operation

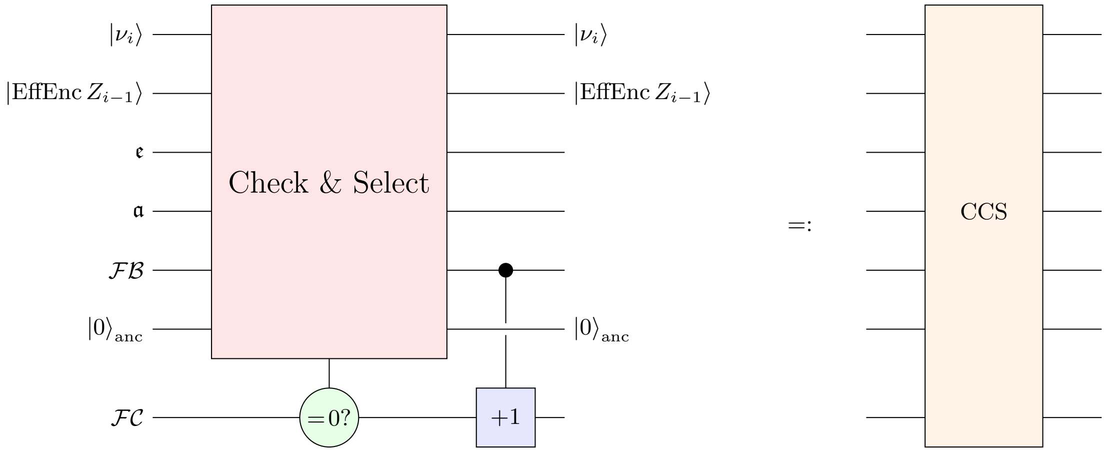

$|\nu_i\rangle$
$|\mathrm{EffEnc}\, Z_{i-1}\rangle$
$\mathfrak{e}$
$\mathfrak{a}$
$\mathcal{FB}$
$|0\rangle_{\mathrm{anc}}$
$\mathcal{FC}$
Check & Select
$=0?$
$+1$
$|\nu_i\rangle$
$|\mathrm{EffEnc}\, Z_{i-1}\rangle$
$|0\rangle_{\mathrm{anc}}$
=:
CCS

FIG. 2: Controlled check-and-select (CCS) operation. The ancilla register consists of $O(\log n)$ bits.

corresponding to lines 24–25 of Alg. 3 does the following: if $G\backslash D$ contains a vertex with degree 2 with at least one edge in $E\backslash(F \cup F' \cup D)$, it adds to $\mathfrak{e}$ the value of e of one of the edges in $E\backslash(F \cup F' \cup D)$ incident to such a vertex (if multiple such vertex/edge combinations exist, e is taken to be the first such edge of the first such vertex), leaves $\mathfrak{a}$ invariant (such that it stays in 0), and flips $\mathcal{FB}$.

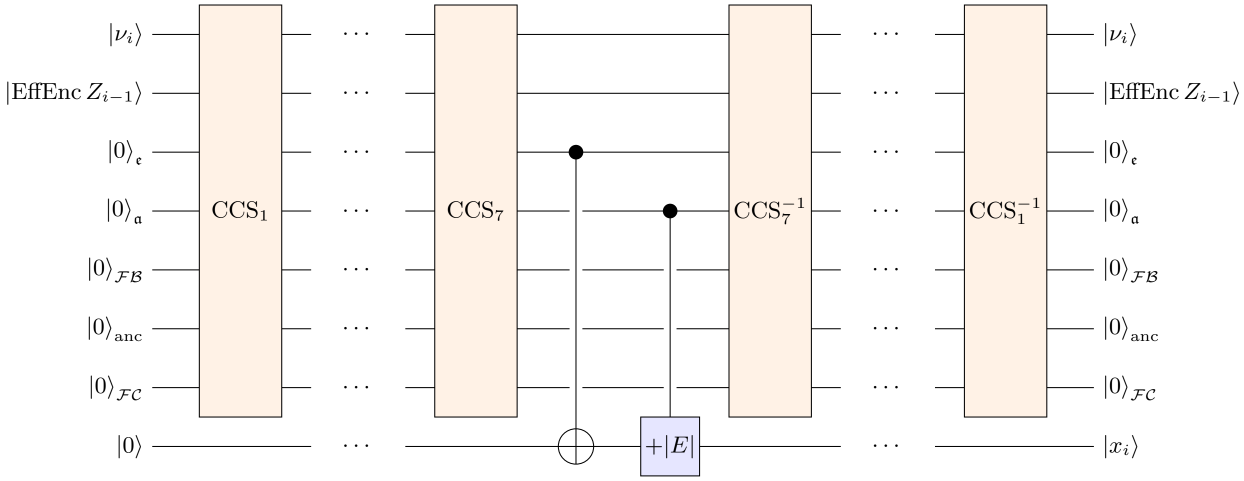

$|\nu_i\rangle$
$|\mathrm{EffEnc}\, Z_{i-1}\rangle$
$|0\rangle_{\mathfrak{e}}$
$|0\rangle_{\mathfrak{a}}$
$|0\rangle_{\mathcal{FB}}$
$|0\rangle_{\mathrm{anc}}$
$|0\rangle_{\mathcal{FC}}$
$|0\rangle$
CCS_1
CCS_7
$+|E|$
CCS_7^{-1}
CCS_1^{-1}
$|\nu_i\rangle$
$|\mathrm{EffEnc}\, Z_{i-1}\rangle$
$|0\rangle_{\mathfrak{e}}$
$|0\rangle_{\mathfrak{a}}$
$|0\rangle_{\mathcal{FB}}$
$|0\rangle_{\mathrm{anc}}$
$|0\rangle_{\mathcal{FC}}$
$|x_i\rangle$

FIG. 3: Reversible implementation of $\mathsf{Calculate}_{G,F,i}$ using $O(\log n)$ ancillas. The seven CCS operations use the check-and-select operations corresponding to lines 24–25, 26–27, 28–29, 30–31, 32–33, 34–35, and 36–37, of Alg. 3, respectively.

The full reversible implementation of $\mathsf{Calculate}_{G,F,i}$ is given in Fig. 3. It consists of these seven CCS operations concatenated in sequence, then adds $\mathfrak{e} + \mathfrak{a}|E|$ to the output register, and finally applies the inverses of the CCS operations. The latter ensure that all registers except for the input registers ($|\nu_i\rangle\, |\mathrm{EffEnc}\, Z_{i-1}\rangle$) and the ouput register are reset to $|0\rangle$.

It thus remains to show how to reversibly implement each of the seven check-and-select operations. It is easy to see, however, that each of these can be implemented using $O(\log n)$ ancilla bits. Indeed, all check-and-select operations can be formulated as a search over a set of objects that can be classically enumerated beforehand, since G and F are classical inputs, and the checking of each individual object can always be implemented using a constant number of ancillas and calls to $\mathsf{EffContains}_{i-1}$.

For example, checking whether any given vertex v has any of the properties in question (e.g., whether v has degree 2 with at least one incident edge in $E\backslash(F \cup F' \cup D)$) reduces to checking which, if any, of its incident edges e have been forced (i.e, $e \in X_{i-1}$) or deleted (i.e, $e+|E| \in X_{i-1}$). The latter can be done using a constant number of calls to $\mathsf{EffContains}_{i-1}$. Controlled on the outcome, a suitable edge and action bit is added to $\mathfrak{e}_{\mathrm{temp}}$ and $\mathfrak{a}_{\mathrm{temp}}$, respectively, and $\mathcal{FB}_{\text{temp}}$ flipped, where $\mathfrak{e}_{\text{temp}}$, $\mathfrak{a}_{\text{temp}}$ and $\mathcal{FB}_{\text{temp}}$ are ancilla registers of the same sizes as $\mathfrak{e}$, $\mathfrak{a}$, and $\mathcal{FB}$, respectively. The full CCS operation is then implemented by introducing a counter from 0 to n initially set to 0, and then, controlled on the counter being 0, performing the previous operation for all n vertices in sequence, followed by incrementing that counter controlled on $\mathcal{FB}_{\text{temp}}$ (this is conceptually the same as the sequence of CCS operations in Fig. 2–3, except with $\mathcal{FC}$ replaced by a counter from 0 to n, and $\mathfrak{e}$, $\mathfrak{a}$, and $\mathcal{FB}$ replaced by $\mathfrak{e}_{\text{temp}}$, $\mathfrak{a}_{\text{temp}}$, and $\mathcal{FB}_{\text{temp}}$, respectively). Note that the final value of the counter is then $n+1-$ the first vertex with the property (if the counter is 0, no vertex has that property). Then, the values of $\mathfrak{e}_{\text{temp}}$, $\mathfrak{a}_{\text{temp}}$ and $\mathcal{FB}_{\text{temp}}$ are added to $\mathfrak{e}$, $\mathfrak{a}$, and $\mathcal{FB}$, respectively. After that, applying the inverse of the property checks for all vertices resets all ancillas. This covers the CCS operations corresponding to lines 24–25, 26–27, 34–35, and all but one of the subcases of lines 30–31 of Alg. 3.

A similar iteration through all edges in E covers the CCS operation corresponding to lines 36–37 of Alg. 3.

One can similarly check if $G\backslash(F \cup F' \cup D)$ contains 4-cycles with certain properties, and select incident edges accordingly if it does. Indeed, note that $G\backslash F$ has at most $O(n)$ 4-cycles, which can be enumerated in a pre-specified order, so a counter using $O(\log n)$ bits suffices. This covers the CCS operations corresponding to lines 28–29 and 32–33 of Alg. 3.

Finally, to check if $G\backslash(F \cup F' \cup D)$ is a collection of disjoint 4-cycles and isolated vertices, note that this is the case if and only if every edge in $E\backslash(F \cup F' \cup D)$ is part of exactly one 4-cycle in $G\backslash(F \cup F' \cup D)$. This can thus be checked in a similar manner by sequentially going through all edges e, and then, for each (not necessarily unforced-isolated) 4-cycle $\omega \ni e$ in $G\backslash F$, checking if any of its edges are in $F' \cup D$. This covers the final subcase of the CCS operation corresponding lines 30–31 of Alg. 3. □

Using Theorem 2, we obtain the efficient reversible implementation of Reduce from Proposition 8.

Corollary 2. *Let (G, F) be a trivial-reduction-free FCHC instance. Then, the operation* $\textsc{Reduce}_{G,F}$ *defined by Eq. (46) can be implemented reversibly with $O(s\log(n/s)+s+\log n)$ ancillas and $O(\text{poly}(n))$ gates, where $s = s(G,F)$ and $n = n(G)$.*

2. *Reversible space-efficient implementation of* $\textsc{Check}_{G,F}$

The reversible and space-efficient implementation of $\textsc{Check}_{G,F}$ follows the (non-reversible) implementation of Check in Alg. 3 by checking in turn whether the conditions in lines 44, 46, 48, and 50, respectively, apply.

It is clear that checking the first three conditions of Check, namely whether G' contains a vertex of degree 1, whether $F \cup F'$ contains three edges meeting at a vertex, or whether $G\backslash(F \cup F' \cup D)$ is a collection of disjoint 4-cycles and isolated vertices, can each be done in the same way as the check-and-select operations in the proof of Proposition 8, and thus can be implemented reversibly using $O(\log n(G))$ ancillas and $O(\text{poly}(n(G)))$ gates. Hence, it only remains to check whether $G\backslash D$ is connected. We use the fact that there is a classical reversible space-efficient algorithm to check that. Indeed, [22] shows that there is a (not necessarily reversible) classical algorithm that decides in time $O(\text{poly}(n(G)))$ and space $O(\log n(G))$ if a graph G is connected. Ref. [23] then shows that any given (not necessarily reversible) computation requiring memory S and time T can be implemented reversibly using S ancillas and $2^{O(S)}$ gates (see also [15, 24]). This implies the following.

Proposition 9 ([22, 23])**.** *There exists a classical deterministic algorithm that checks if a given graph G is connected in time $O(\text{poly}\, n(G))$, which can be implemented reversibly using $O(\log n(G))$ ancillas.*

Note that Proposition 9 only requires oracular access to the adjacency matrix of the graph in question [22]. Moreover, we can easily access the required adjacency matrix, i.e., the map

$$|\text{EffEnc}\, Z(\vec{\nu})\rangle\, |v\rangle\, |w\rangle\, |0\rangle \mapsto |\text{EffEnc}\, Z(\vec{\nu})\rangle\, |v\rangle\, |w\rangle\, |v \sim_{G'} w\rangle \tag{49}$$

can be implemented reversibly with $O(1)$ ancillas and at most three call to $\mathsf{EffContains}_r$, where the last bit in (49) is 1 if the vertices v and w are connected in $G' = G\backslash D$, and 0 otherwise. This completes our space-efficient reversible implementation of $\textsc{Check}_{G,F}$.

Corollary 3. *Let (G, F) be a trivial-reduction-free FCHC instance. Then, the operation* $\textsc{Check}_{G,F}$ *defined by Eq. (46) can be implemented reversibly using $O(\log n(G))$ ancillas and $O(\text{poly}(n(G)))$ gates.*

This concludes the proof of Theorem 3. □

VI. CONCLUSION AND OUTLOOK

The recent progress in experimental quantum computing [25–27] increases confidence that fully scalable quantum computers will be realised at some point in the upcoming decades. However, the rate at which the number of qubits we can manipulate with relevant precision and coherence times currently grows provides significant motivation for studying potential uses of size-limited quantum computers. Complementary to research dedicated to solving small-yet-hard simulation and ground-state problems [28–31], which are promising applications for really small quantum computers, in this work, we investigated ways to achieve speedups of classical algorithms by exploiting quantum computers significantly smaller than the problem size. Concretely, we provide a framework for designing hybrid quantum-classical algorithms, which can allow for polynomial asymptotic speedups given a quantum computer which is any constant fraction of the problem size.

Our result also implies that we can achieve a trade-off between the speedup we obtain, and the size of problem we wish to tackle. Thus, a small quantum computer can dramatically speed up the solving of small problems, but can be used to achieve more modest speedups of larger instances as well. Such trade-offs have, to our knowledge, not been explored before our works.

We provided the general formalism in the form of the so-called divide-and-conquer hybrid approach, which enables us to realise such trade-offs for a broad class of recursive classical algorithms, and we provided a characterisation of the space-efficiency of the quantum subroutines required to achieve polynomial speedups. Moreover, we provided a toolkit for the space-efficient reversible generation and manipulation of sets, which is often the bottleneck of the space requirements of many such algorithms. As an illustration, we show how this framework can be applied to speed up the algorithm of Eppstein for detecting Hamilton cycles in cubic graphs.

We also identify a number of questions that remain unresolved, both from a purely theoretical and from an applied perspective.

First, the algorithms which we so far have applied our hybrid approach to are not the absolutely best known algorithms for their respective problems. Indeed, Eppstein's algorithm has subsequently been improved from $O(1.2599^n \operatorname{poly}(n))$ to $O(1.2509^n \operatorname{poly}(n))$ [32] and $O(1.2312^n \operatorname{poly}(n))$ [33], and a Monte-Carlo algorithm with runtime $O(1.2009^n \operatorname{poly}(n))$ is known [34]. It would be interesting to apply our framework to speed up the actually best classical algorithms, which would yield an asymptotic speedup over the best classical algorithms for a quantum computer the size of any constant fraction of the problem size.

Second, our approach currently focuses on Grover-based speedups; however, the quantum backtracking techniques [6, 35] and subsequent improvements [36] lead to better performing quantum algorithms. It remains an open question whether these methods can also be made to fit in our hybrid approach, and whether they would thus yield better speedups.

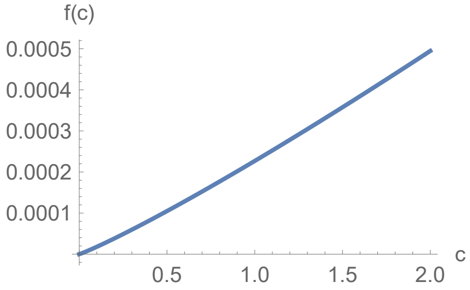


FIG. 4: Plot of $f(c)$ in Theorem 4.

From a more applied perspective, there are three key issues which currently prevent our algorithms from being practical. First, the precise degree of the speedup we obtain is still quite small (see Fig. 4 for the plot of $f(c)$ obtained for speeding up Eppstein's algorithm using a straightforward trit encoding [37]). Our results are primarily conceptual, however, in that a polynomial speedup can in principle be obtained even for arbitrarily small values of c, and we remark that there is likely to be room for significantly improving the actual degree of that speedup.

Second is the fact that we deal with asymptotic speedups, and focus on algorithm performance in the worst-case exponential run-times. For the issue of asymptotic statements, there has lately been increasing interest in analysing performance for finite-size settings [38], which were moderately promising, but required arbitrarily-sized quantum computers. It would be interesting to see if a similar claim could be made for size-limited quantum computers.

Finally, we assume ideal noiseless settings, which is still remote [39]. Note that exponential run-times essentially require full fault tolerance to yield reliable results. It would be interesting to consider our hybrid approach for heuristic algorithms, which run for low-polynomial times, and while they may fail to find solutions for most truly hard instances, still perform very well in practice. These much more efficient algorithms would be more important for real-world solutions to NP-hard problems. Furthermore, in this case, it is more likely that intermediary efficient error mitigation schemes, as opposed to full fault tolerance, suffices to achieve quantum-enhanced and usable NP heuristics.

ACKNOWLEDGMENTS

We thank J.I. Cirac for helpful discussions. VD is partially funded through the Quantum Software Consortium.

[1] V. Dunjko, Y. Ge, and J. I. Cirac, Phys. Rev. Lett. **121**, 250501 (2018).

[2] Note that since reductions of one NP-complete problem to another generally incur significant polynomial slowdowns, and we only expect polynomial speedups, a speedup for 3SAT does not imply a speedup for other NP-complete problems. Consequently, each problem and algorithm must be treated individually.

[3] S. Bravyi, G. Smith, and J. A. Smolin, Phys. Rev. X **6**, 021043 (2016).

[4] T. Peng, A. Harrow, M. Ozols, and X. Wu, "Simulating large quantum circuits on a small quantum computer," (2019), arXiv:1904.00102.

[5] D. Eppstein, Journal of Graph Algorithms and Applications **11**, 61 (2007).

[6] D. J. Moylett, N. Linden, and A. Montanaro, Phys. Rev. A **95**, 032323 (2017).

[7] Many algorithms that are not a priori given in this form can be formulated as such.

[8] For simplicity, we formulate the divide-and-conquer hybrid approach for decision problems here. The approach can be generalised to algorithms with more general outputs, subject to size constraints of the output.

[9] In general, the recursive algorithm can also take additional parameters, but these can be incorporated into P.

[10] J. L. Bentley, D. Haken, and J. B. Saxe, SIGACT News **12**, 36 (1980).

[11] A. Ambainis, SIGACT News **35**, 22 (2004).

[12] R. A. Moser and D. Scheder, in *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11 (ACM, New York, NY, USA, 2011) pp. 245–252.

[13] E. Dantsin, A. Goerdt, E. A. Hirsch, R. Kannan, J. Kleinberg, C. Papadimitriou, P. Raghavan, and U. Schöning, Theoretical Computer Science **289**, 69 (2002).

[14] T. Schöning, in *40th Annual Symposium on Foundations of Computer Science (Cat. No.99CB37039)* (1999) pp. 410–414.

[15] H. Buhrman, J. Tromp, and P. Vitányi, in *Automata, Languages and Programming*, edited by F. Orejas, P. G. Spirakis, and J. van Leeuwen (Springer Berlin Heidelberg, Berlin, Heidelberg, 2001) pp. 1017–1027.

[16] Note that the binary representation without leading zeros of a positive integer y has $\lceil \log_2(y+1) \rceil$ bits.

[17] Note that in the Supplemental Material of [1], $\mathsf{Convert}_N$ was called Append_0.

[18] The result can be generalised to the travelling salesman problem, subject to constraints on the distances depending on the number of available qubits.

[19] For simplicity of notation, we will in this section not distinguish between an edge e and its number in the enumeration, and simply write $e = 10$ to mean that e is the 10^{th} edge according to the enumeration. We do the same for vertices.

[20] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, in *Quantum Computation and Information*, AMS Contemporary Mathematics Series, Vol. 305 (AMS, 2002).

[21] T. J. Yoder, G. H. Low, and I. L. Chuang, Phys. Rev. Lett. **113**, 210501 (2014).

[22] O. Reingold, in *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05 (ACM, New York, NY, USA, 2005) pp. 376–385.

[23] K.-J. Lange, P. McKenzie, and A. Tapp, Journal of Computer and System Sciences **60**, 354 (2000).

[24] R. Williams, "Space-efficient reversible simulations," (2000).

[25] "Ieee spectrum. ibm edges closer to quantum supremacy with 50-qubit processor," `https://spectrum.ieee.org/tech-talk/computing/hardware/ibm-edges-closer-to-quantum-supremacy-with-50qubit-processor` (2017).

[26] "American Physical Society meeting. engineering superconducting qubit arrays for quantum supremacy," `http://meetings.aps.org/Meeting/MAR18/Session/A33.1` (2018).

[27] "Intel newsroom. 2018 ces: Intel advances quantum and neuromorphic computing research," `https://newsroom.intel.com/news/intel-advances-quantum-neuromorphic-computing-research/` (2018).

[28] S. Lloyd, Science **273**, 1073 (1996).

[29] D. Wecker, M. B. Hastings, and M. Troyer, Phys. Rev. A **92**, 042303 (2015).

[30] M.-H. Yung, J. D. Whitfield, S. Boixo, D. G. Tempel, and A. Aspuru-Guzik, "Introduction to quantum algorithms for physics and chemistry," in *Quantum Information and Computation for Chemistry* (John Wiley & Sons, Inc., 2014) pp. 67–106.

[31] Y. Ge, J. Tura, and J. I. Cirac, Journal of Mathematical Physics **60**, 022202 (2019).

[32] K. Iwama and T. Nakashima, in *Proceedings of the 13th Annual International Conference on Computing and Combinatorics*, COCOON'07 (Springer-Verlag, Berlin, Heidelberg, 2007) pp. 108–117.

[33] M. Xiao and H. Nagamochi, Algorithmica **74**, 713 (2016).

[34] M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. v. Rooij, and J. O. Wojtaszczyk, in *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science* (2011) pp. 150–159.

[35] A. Montanaro, Theory of Computing **14**, 1 (2018).

[36] A. Ambainis and M. Kokainis, in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017 (ACM, New York, NY, USA, 2017) pp. 989–1002.

[37] The value of $f(c)$ depends on the constants A and B defined in Example 1. Straightforwardly encoding each trit using two qubits yields $A \approx 39$ and $B \approx 137$ for the FCHC quantum algorithm in Theorem 3. These values can likely be improved significantly.

[38] E. Campbell, A. Khurana, and A. Montanaro, "Applying quantum algorithms to constraint satisfaction problems," (2018), arXiv:1810.05582.

[39] J. Preskill, Quantum **2**, 79 (2018).

Appendix A: Runtime analysis of Eppstein

In this section, we prove the runtime of Eppstein (as defined in Alg. 2) of $O(2^{s(G,F)/3}\,\mathrm{poly}(n(G)))$. Note that since $s(G,F) \leq n(G)$, this in particular implies a runtime of $O(2^{n(G)/3}\,\mathrm{poly}(n(G)))$, which is the bound commonly cited in literature. The analysis is essentially the same as that in [5].

We first introduce a few notions for convenience.

Definition 11. For any FCHC instance (G,F), let TrivRed(G,F) be the FCHC instance obtained from (G,F) by applying step 1 of Eppstein to (G,F).

In other words, TrivRed is the first step of Eppstein which performs all possible trivial reductions (step 1a, 1b, and 1c) until no more such reductions are possible. In particular, TrivRed(G,F) is trivial-reduction-free for any FCHC instance (G,F), and TrivRed$(G,F) = (G,F)$ whenever (G,F) is trivial-reduction-free.

Definition 12. Let (G,F) be a trivial-reduction-free FCHC instance. We say that (G,F) is *non-terminal* if

(i) G does not contain any vertices of degree 0 or 1,

(ii) F does not contain three edges meeting at a vertex,

(iii) $G\backslash F$ is not a collection of disjoint 4-cycles and isolated vertices, and

(iv) F does not contain a non-Hamiltonian cycle.

Otherwise, we call (G,F) *terminal*.

In other words, a trivial-reduction-free FCHC instance is non-terminal if and only if none of the terminal conditions in step 2 of Eppstein apply.

Definition 13. For any trivial-reduction-free and non-terminal FCHC instance (G,F), let EdgeSelect(G,F) be the edge selected in step 3 of Eppstein(G,F).

In other words, if (G,F) is trivial-reduction-free and non-terminal, then steps 4 and 5 of Eppstein(G,F) call Eppstein$(G,F\cup\{$EdgeSelect$(G,F)\})$ and Eppstein$(G\backslash\{$EdgeSelect$(G,F)\},F)$, respectively.

Eppstein's key idea to bounding the runtime was to show that with each recursive call, $s(G,F)$ reduces by a constant larger than 1, leading to a runtime that is polynomially better than a trivial path-search algorithm (see also Lemma 7 in [5]).

Proposition 10. *Let (G,F) be a trivial-reduction-free and non-terminal FCHC instance and let $e =$ EdgeSelect(G,F). Suppose that F is nonempty, and let $(G_1,F_1) =$ TrivRed$(G,F\cup\{e\})$ and $(G_2,F_2) =$ TrivRed$(G\backslash\{e\},F)$. Suppose that (G_1,F_1) and (G_2,F_2) are non-terminal. Then,*

- $s(G_1,F_1), s(G_2,F_2) \leq s(G,F) - 3$, *or*
- $s(G_1,F_1) \leq s(G,F) - 2$ *and* $s(G_2,F_2) \leq s(G,F) - 5$, *or vice-versa.*

Proof. We prove the claim by going through all possible cases that can occur in EdgeSelect(G,F).

Suppose first that e is selected according to step 3a of Eppstein (see Fig. 5), i.e., $G\backslash F$ contains a 4-cycle ω with exactly two vertices incident to edges in F, and e is one of the other two edges adjacent to vertices in the cycle. Note that indeed, since (G,F) is trivial-reduction-free, every vertex in ω is adjacent to a non-cycle edge. Note moreover

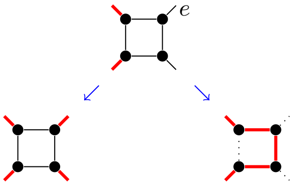


FIG. 5: If EdgeSelect(G,F) selects e according to step 3a of Eppstein, $C(G_1,F_1) \geq C(G,F)+1$, $|F_1| \geq |F|+2$, and $|F_2| \geq |F|+3$.

that since (G,F) is trivial-reduction-free, the two vertices of ω which are incident to edges in F must be adjacent. Hence, ω has two opposite vertices which are incident to an edge in $F \cup \{e\}$. Thus, TrivRed$(G, F \cup \{e\})$ forces the last non-cycle edge adjacent to ω. It follows that $|F_1| \geq |F|+2$ and $|C(G_1,F_1)| \geq |C(G,F)|+1$, and hence $s(G_1,F_1) \leq s(G,F)-3$. On the other hand, removing e leads to two unforced edges in ω, which will be forced, leading to one of the vertices in ω to have two incident forced edges. This leads to the removal of the third (cycle) edge, hence forcing the final edge in ω. It follows that $|F_2| \geq |F|+3$, and hence $s(G_2,F_2) \leq s(G,F)-3$.

Next, suppose that $e = yz$ is selected according to step 3b of Eppstein. Let w be the third vertex adjacent to y in G. Note that since (G,F) is trivial-reduction-free, $yw \notin F$ and $\deg z = \deg w = 3$. We distinguish two cases.

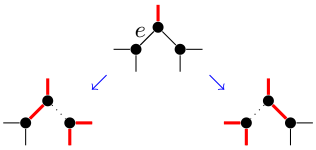


FIG. 6: If EdgeSelect(G,F) selects e according to step 3b of Eppstein, and neither z nor w have an incident edge in F, then $|F_1|, |F_2| \geq |F|+3$.

In the first case, suppose that neither z nor w have an incident edge in F (Fig. 6). Then, TrivRed$(G, F \cup \{e\})$ first removes yw and then adds both remaining edges incident to w to F. Hence, $s(G_1,F_1) \leq s(G,F)-3$. Similarly, $s(G_2,F_2) \leq s(G,F)-3$.

In the second case, suppose that z or w, have an incident edge in F. Note that since $xy \in F$, yz and yw cannot be part of a 4-cycle of unforced edges, because otherwise EdgeSelect(G,F) would choose e according to step 3a of Eppstein. Thus, z, y, w are part of an unforced, and possibly closed, chain of $k \geq 4$ vertices, with the inner vertices each having an incident edge in F. There are two subcases here.

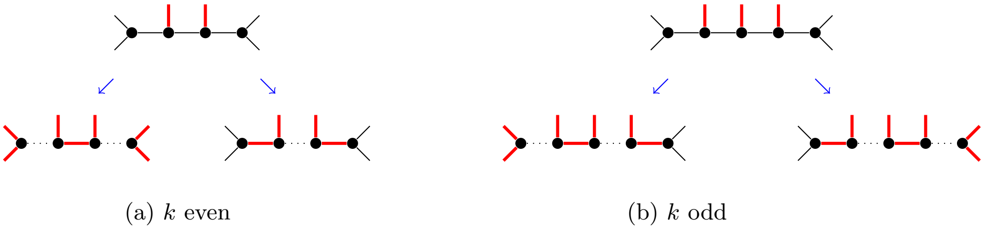


FIG. 7: EdgeSelect(G,F) selects e according to step 3b of Eppstein, and z,y,w are part of an unforced chain of $k \geq 4$ vertices such that the inner $k-2$ vertices each have an incident edge in F and the outer two vertices each have three unforced incident edges. (a) If k is even, then $|F_1| \geq |F|+5$ and $|F_2| \geq |F|+2$ or vice-versa. (b) If k is odd, then $|F_1|, |F_2| \geq |F|+4$.

In the first subcase, suppose that this chain terminates (see Fig. 7). Thus, z, y, w are part of an unforced chain of $k \geq 4$ vertices such that the inner $k-2$ vertices each have an incident edge in F and the outer two vertices have three unforced incident edges. Then, TrivRed$(G, F \cup \{e\})$ and TrivRed$(G \backslash \{e\}, F)$ force and delete alternating edges of that chain. If $k = 2l$ is even, then one of (G_1,F_1) and (G_2,F_2) has $l-1$ edges of that chain which are forced, and the other has l (Fig. 7a). Assume without loss of generality that (G_1, F_1) has $l-1$ edges of that chain forced. Then, both outer vertices of the chain will eventually have degree two, and thus TrivRed forces four additional edges. It follows that $|F_1| \geq |F| + 4 + (l-1) \geq |F| + 5$ and $|F_2| \geq |F| + l \geq |F| + 2$, and hence $s(G_1, F_1) \leq s(G, F) - 5$ and $s(G_2, F_2) \leq s(G, F) - 2$. On the other hand, if $k = 2l + 1$ is odd, then TrivRed$(G, F \cup \{e\})$ and TrivRed$(G \backslash \{e\}, F)$ both force l edges of the chain, and leave one of the two outer vertices with degree two, thus forcing its other two incident edges (Fig. 7b). It follows that $|F_1|, |F_2| \geq |F| + l + 2 \geq |F| + 4$ and hence $s(G_1, F_1), s(G_2, F_2) \leq s(G, F) - 4$.

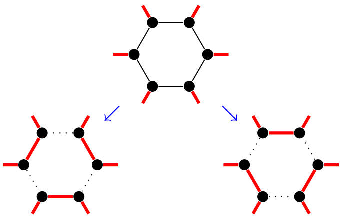


FIG. 8: If EdgeSelect(G, F) selects e according to step 3b of Eppstein, and z, y, w are part of an unforced cycle of $k \geq 6$ vertices with k even, each of which is incident to an edge in F, then $|F_1|, |F_2| \geq |F| + 3$.

In the second subcase, this unforced chain is a cycle of k vertices (Fig. 8), each of which is incident to an edge in F. Then, TrivRed$(G, F \cup \{e\})$ and TrivRed$(G \backslash \{e\}, F)$ force and delete alternating edges of that chain. Clearly, by the definition of Step 3b of Eppstein, $k > 4$. Moreover, if k is odd, TrivRed$(G, F \cup \{e\})$ and TrivRed$(G \backslash \{e\}, F)$ would both have three forced edges meeting in a vertex, and hence would be terminal. Hence, $k \geq 6$, and thus $|F_1|, |F_2| \geq |F| + 3$. Hence, $s(G_1, F_1), s(G_2, F_2) \leq s(G, F) - 3$.

Finally, note that since F is assumed to be non-empty and (G, F) to be non-terminal, EdgeSelect(G, F) does not choose e according to step 3c of Eppstein. □

Corollary 4. *For any FCHC instance (G, F), Eppstein(G, F) decides the FCHC problem in a runtime of $O(2^{s(G,F)/3}\,\mathrm{poly}(n(G)))$.*

Proof. Clearly, if (G, F) is trivial-reduction-free and terminal, Eppstein(G, F) only takes $O(\mathrm{poly}(n(G)))$ time. Moreover, if (G, F) is trivial-reduction-free, then by Propostion 6, $s(G, F) = 0$ implies that (G, F) is terminal. Hence, Proposition 10 implies that the runtime of Eppstein(G, F) can be bounded by some function $T(s(G, F))$ depending only on $s(G, F)$, where $T(s)$ satisfies $T(s) = O(\mathrm{poly}(n(G)))$ for $s \leq 0$, and

$$T(s) \leq \max(2T(s-3), T(s-2) + T(s-5)) \tag{A1}$$

for $s > 0$. Using standard techniques for solving linear recurrence relations, one obtains $T(s) = O(2^{s/3}\,\mathrm{poly}(n(G)))$ for $s \geq 0$. □

C Further articles as co-author

C.1 A generalization of the injectivity condition for projected entangled pair states

A generalization of the injectivity condition for projected entangled pair states

Andras Molnar, Yimin Ge, Norbert Schuch, and J. Ignacio Cirac

In this work, we introduce and study the notion of "semi-injective" PEPS, a generalisation of the well-studied injective PEPS. We construct parent Hamiltonians for which these states are the unique ground states. We then prove a "fundamental theorem of semi-injective PEPS", which provides necessary and sufficient conditions for when two tensors generate the same family of such states. Finally, we use these results to show that the third cohomology classification of symmetry protected topological (SPT) phases extends to semi-injective PEPS.

Tensor network states are local descriptions of quantum many-body states which are generally expected to be able to capture ground states of local Hamiltonians. In one spatial dimension, MPS are highly successful, giving rise not only to very good numerical methods for simulating such systems, but also a framework to classify SPT phases in one dimension. One reason for the success of MPS is that a general structure theory of MPS is known. The central component of the latter is the "fundamental theorem of MPS", which fully characterises when two MPS tensors give rise to the same family of states, namely when the two tensors are related by a "gauge" transformation acting on their virtual indices.

PEPS are the natural two-dimensional generalisations of MPS. However, in the case of PEPS, an analogous fundamental theorem was only known for the class of so-called "injective PEPS". Although a randomly chosen PEPS tensor gives rise to an injective PEPS with probability 1, there are many important examples which are not described by an injective PEPS. These include the AKLT model on the square lattice, the RVB state, certain Gibbs states, as well as all topologically ordered states.

The notion of semi-injective PEPS, which we introduce in Section II, comprises states that are constructed from distributing plaquette states of four virtual particles on a two-dimensional torus and acting with local invertible 4-body operators on the four corners where such plaquette states meet. We show that this class not only includes injective PEPS, but also many of the important examples which are not injective, including the CZX model, purifications of thermal states of commuting nearest neighbour Hamiltonians on square lattices, as well as the AKLT model on square lattices.

In Section IV, we construct different types of parent Hamiltonians for semi-injective PEPS. These are local, frustration-free Hamiltonians for which they are the unique ground states. The central result, stated as Theorem 1 and proven in Section V, is a fundamental theorem for semi-injective PEPS, which shows that any two tensors generating the same class of semi-injective PEPS are related by an invertible Matrix Product Operator (MPO) acting on their auxiliary indices. We also show that the product of the invertible 4-body operator corresponding to one of the semi-injective PEPS and the inverse of the one corresponding to the other has a special two-layer structure.

As in the case of MPS and injective PEPS, the fundamental theorem allows for a characterisation of symmetries of semi-injective PEPS: Theorem 2, which is proven in Section VI, shows that for semi-injective PEPS with an on-site symmetry, the corresponding MPOs form a projective representation of the symmetry group, and that these MPOs can be labelled by the third cohomology group $H^3(G, \mathbb{C}^*)$. We moreover show that the two-layer structure of the symmetry operators can also be assigned a cohomology label, and we show that the latter coincides with the $H^3(G, \mathbb{C}^*)$ label of the boundary.

Statement of individual contribution

This work is the result of frequent discussions between Andras Molnar, Norbert Schuch, J. Ignacio Cirac, and myself. Andras Molnar, who is the principal author of this article, carried out the majority of the scientific work and was moreover in charge of writing almost all parts of this article. I was significantly involved in the initial formulation of the class of states under consideration as well the parent Hamiltonian constructions. I was partially involved in the work on the fundamental theorem. I was only marginally involved in the remainder of the work, particularly the characterisation of SPT phases under this framework.

Permission to include:

Andras Molnar, Yimin Ge, Norbert Schuch, and J. Ignacio Cirac.
A generalization of the injectivity condition for projected entangled pair states.
Journal of Mathematical Physics, 59, 021902 (2018).

Permission to Reuse Content

REUSING AIP PUBLISHING CONTENT

Permission from AIP Publishing is required to:

- republish content (e.g., excerpts, figures, tables) if you are not the author
- modify, adapt, or redraw materials for another publication
- systematically reproduce content
- store or distribute content electronically
- copy content for promotional purposes

To request permission to reuse AIP Publishing content, use RightsLink® for the fastest response or contact AIP Publishing directly at rights@aip.org and we will respond within one week:

For RightsLink, use Scitation to access the article you wish to license, and click on the Reprints and Permissions link under the TOOLS tab. (For assistance click the "Help" button in the top right corner of the RightsLink page.)

To send a permission request to rights@aip.org, please include the following:

- Citation information for the article containing the material you wish to reuse
- A description of the material you wish to reuse, including figure and/or table numbers
- The title, authors, name of the publisher, and expected publication date of the new work
- The format(s) the new work will appear in (e.g., print, electronic, CD-ROM)
- How the new work will be distributed and whether it will be offered for sale

Authors do **not** need permission from AIP Publishing to:

- quote from a publication (please include the material in quotation marks and provide the customary acknowledgment of the source)
- reuse any materials that are licensed under a Creative Commons CC BY license (please format your credit line: "Author names, Journal Titles, Vol.#, Article ID#, Year of Publication; licensed under a Creative Commons Attribution (CC BY) license.")
- reuse your own AIP Publishing article in your thesis or dissertation (please format your credit line: "Reproduced from [FULL CITATION], with the permission of AIP Publishing")
- reuse content that appears in an AIP Publishing journal for republication in another AIP Publishing journal (please format your credit line: "Reproduced from [FULL CITATION], with the permission of AIP Publishing")
- make multiple copies of articles–although you must contact the Copyright Clearance Center (CCC) at www.copyright.com to do this

(…)

https://publishing.aip.org/resources/researchers/rights-and-permissions/permissions/ 03/12/2019

JOURNAL OF MATHEMATICAL PHYSICS **59**, 021902 (2018)

A generalization of the injectivity condition for projected entangled pair states

Andras Molnar, Yimin Ge, Norbert Schuch, and J. Ignacio Cirac
Max-Planck-Institut für Quantenoptik, Hans-Kopfermann-Str. 1, D-85748 Garching, Germany

(Received 29 September 2017; accepted 2 February 2018; published online 26 February 2018)

We introduce a family of tensor network states that we term semi-injective Projected Entangled-Pair States (PEPSs). They extend the class of injective PEPSs and include other states, like the ground states of the AKLT and the CZX models in square lattices. We construct parent Hamiltonians for which semi-injective PEPSs are unique ground states. We also determine the necessary and sufficient conditions for two tensors to generate the same family of such states in two spatial dimensions. Using this result, we show that the third cohomology labeling of symmetry protected topological phases extends to semi-injective PEPSs. *Published by AIP Publishing.* https://doi.org/10.1063/1.5007017

I. INTRODUCTION

Tensor Network States (TNS) are expected to approximate ground states of local Hamiltonians well.[1–3] Their local description in terms of simple tensors makes them suitable for both numerical and analytical investigations. First, this local structure enables calculations in large or infinite systems. In fact, the Density Matrix Renormalization Group (DMRG) algorithm,[4] which proved successful in simulating one-dimensional systems, can be re-expressed in terms of Matrix Product States (MPSs),[5,6] the simplest TNS. This connection also motivated a provably efficient algorithm to find the ground state of a gapped one-dimensional local Hamiltonian.[7] Algorithms based on higher-dimensional generalizations of MPSs, projected entangled-pair states (PEPSs[8,9]), are now also giving the best known numerical results for certain Hamiltonians in two dimensions (see, for example, Refs. 10–12). Second, TNS are useful for creating and analyzing exactly solvable models with translation symmetry. Indeed, paradigmatic wavefunctions appearing in different areas of research have a very simple PEPS description, where one can generate a whole family of many-body states by contracting, according to a given geometry, the so-called auxiliary indices of N copies of a single tensor. In two-dimensional systems, this includes, for instance, the cluster state[13,14] underlying measurement based computation, the rotationally invariant spin-liquid AKLT[15,16] and resonating valence bond (RVB)[17] states, and other chiral[18,19] and non-chiral states[20–23] embodying topological order. In particular, PEPSs encompass all known non-chiral topological orders.[24] All these states allow for the construction of local parent Hamiltonians and, by making use of their local description, the ground space structure and the behavior of the low-energy excitations can be fully analyzed.

In the last years, the theory of TNS has been considerably developed. In particular, in one dimension a general structural theory of MPS is known. First, the "fundamental theorem of MPS"[25,26] states that any two tensors generating the same family of wavefunctions can always be related by a "gauge" transformation acting on the auxiliary indices of each tensor independently. This result allowed for the classification of symmetry-protected topological (SPT) phases in one dimension.[27,28] Indeed, let us consider an MPS invariant under a symmetry group G. Then, the local gauge transformations which relate the tensor generating the MPS and that obtained by a symmetry action form themselves a *projective* representation of G.[29,30] The equivalence classes formed by those projective representations, which are labeled by the second group cohomology $H^2(G, U(1))$, thus characterize the different SPT phases under the action of G. Second, for any MPS, there exists a systematic way of constructing parent Hamiltonians with a controlled ground space.[25,31,32] All this renders MPSs a general framework for the study of solvable models and the classification of phases in one dimension.

In two dimensions, considerable progress has been made in understanding the description of topologically ordered phases in the PEPS framework.[20–23,33,34] However, a general structural theory of PEPSs and, in particular, a "fundamental theorem of PEPS" are only known for a specific class, termed "injective."[31,35] Those are PEPSs whose generating tensor, when considered as a map from the auxiliary to the physical indices, can be inverted. While formally, any random PEPS (after blocking few spins) is injective, many relevant examples such as the square lattice AKLT model,[16] the RVB[36,37] state, wavefunctions obtained from classical statistical mechanics models,[37] and all topologically ordered states, do not have this property. Their lack of injectivity prevents us from understanding their behavior under symmetries and, at the same time, makes the canonical construction of parent Hamiltonians with unique ground states for injective PEPS inapplicable.

In order to analyze SPT phases in two dimensions, one might thus tentatively restrict to injective PEPSs and apply the fundamental theorem, which yet again yields a projective local symmetry action on the auxiliary indices. However, in two dimensions, the classification of projective representations in terms of group cohomology is not stable under blocking and thus cannot be used to label phases under symmetries unless translational invariance is imposed.[38] This is remedied by the CZX model proposed by Chen *et al.*[38] It is made up from a *non-injective* PEPS with a corresponding symmetry action, in a way that it exhibits non-trivial symmetry invariants which are insensitive to blocking. Specifically, the symmetry action on the auxiliary indices at the boundary of any region is given by Matrix Product Unitary Operators (MPUOs). Those can be labeled by elements of the third cohomology group $H^3(G, U(1))$ of the symmetry group G, and therefore those elements are expected to label the different SPT phases.[38,39]

In this paper, we introduce *semi-injective* PEPSs and develop the structure theory for them. They significantly generalize injective PEPSs and include the square-lattice AKLT model, all wavefunctions based on classical models, and CZX-like states, among others. Our central result is a "fundamental theorem of semi-injective PEPS" which states that any two tensors generating the same semi-injective PEPS are related by an invertible Matrix Product Operator (MPO) acting on their auxiliary indices. For semi-injective PEPSs with an on-site symmetry, the corresponding MPOs form a representation of the group. We give a general and fully rigorous proof, based on the arguments of Chen *et al.*,[38] that these MPO representations are labeled by the third cohomology group $H^3(G, \mathbb{C}^*) = H^3(G, U(1))$, suggesting that this labels SPT phases for all semi-injective PEPSs, including those away from fixed point wavefunctions[40] or with non-unitary symmetries. We further show that symmetries of semi-injective PEPSs must have a special two-layer structure. This, by itself, enables the definition of an MPO acting on the physical indices alone and thus allows one to assign another label $H^3(G, \mathbb{C}^*)$ to the symmetry action, which we show to coincide with that of the MPO acting on the boundary. Therefore, the SPT labeling is just as much a property of the physical symmetry itself as it is of the boundary and can in fact be directly inferred by analyzing the structure of the physical symmetry action, without needing to invoke the underlying PEPS. As a corollary, this implies that the H^3 label of the MPO action is well defined and, in particular, the same on the horizontal and vertical boundaries, also in the absence of rotational symmetry. Furthermore, we also provide two different constructions for parent Hamiltonians with unique ground states.

The paper is structured as follows. In Sec. II, we introduce the formalism and define semi-injective PEPSs. In Sec. III, we give an overview of the main results of the paper. Readers who are interested only in the results rather than the proofs might thus restrict to Secs. II and III. In Sec. IV, we discuss the parent Hamiltonian for semi-injective PEPSs. In Sec. V, we summarize central results from the structure theory of MPS needed for the remainder of the paper. In Sec. VI, we develop the structure theory of semi-injective PEPSs, i.e., we give a local characterization of when two semi-injective PEPSs generate the same state. In Sec. VII, we apply this to characterize symmetric semi-injective PEPSs, yielding the characterization in terms of the third cohomology group.

II. FORMALISM

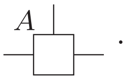
In this section, we introduce MPSs, PEPSs, and the graphical calculus commonly used in the field of tensor networks (TNs). We modify the standard notations in order to be able to represent TNs that are concatenations of several layers of two-dimensional TNS. Using this notation, we define semi-injective PEPSs. We show that this class of PEPSs contains all injective PEPSs as well as some examples that are not known to have injective PEPS description.

A. Notation

Translationally invariant MPSs are defined in terms of a single rank-three tensor $A \in \mathbb{C}^D \otimes \mathbb{C}^D \otimes \mathbb{C}^d$, $A = \sum_{\alpha,\beta,i} A^i_{\alpha\beta} \,|\alpha\rangle\,\langle\beta| \otimes |i\rangle = \sum_i A^i \otimes |i\rangle$. The corresponding state on n particles is

$$|V_n(A)\rangle = \sum_i \mathrm{Tr}\{A^{i_1} \ldots A^{i_n}\}\, |i_1 \ldots i_n\rangle\,. \tag{1}$$

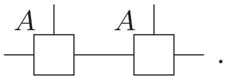
The tensor and the corresponding state can be represented graphically as follows. Each tensor is represented with a square with lines attached to it. The number of lines connected to the rectangle is the rank of the tensor, and each line represents one index. For example, the single MPS tensor is represented as



(2)

Tensor contractions are depicted by joining the lines corresponding to the indices contracted. For example, the contraction of two MPS tensors is

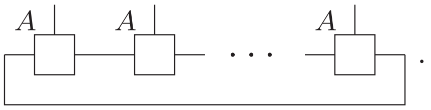
$$\sum_{\alpha\beta\gamma ij} A^i_{\alpha\beta} A^j_{\beta\gamma} |\alpha\rangle\langle\gamma| \otimes |ij\rangle \;=$$



(3)

Same way, the MPS can be depicted as

$$|V_n(A)\rangle = \sum_i \mathrm{Tr}\{A^{i_1} \ldots A^{i_n}\} |i_1 \ldots i_n\rangle \;=$$

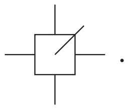


(4)

We refer to the contracted legs as *bonds* or *virtual indices*, D as the *bond dimension* of the MPS tensor A, the uncontracted leg as *physical index*, and d as the *physical dimension* of A.

We will define PEPSs now as generalizations of MPSs. We will consider a square lattice, although other geometries can also be used. Take a rank-five tensor B,

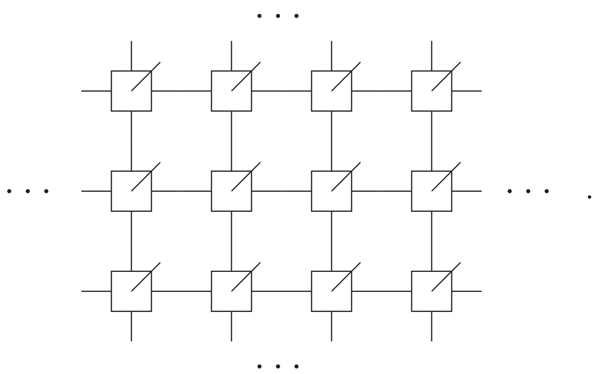
$$B =$$



(5)

Consider an $n \times m$ rectangular grid with periodic boundary conditions (that is, on a torus). The PEPS is defined then by placing the tensor B at every lattice point and contracting the neighboring tensors,

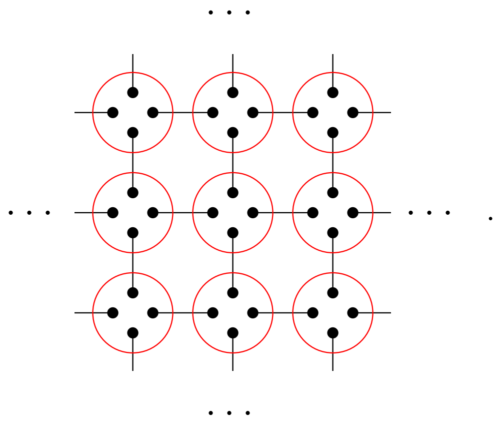
$$V_{n,m}(B) =$$



(6)

An equivalent description is the following. Place maximally entangled pair states on the edges of the grid. These particles are referred to as virtual particles. At every lattice point, act with an operator on the four virtual particles closest to the lattice point,

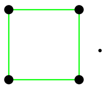
$$V_{n,m}(B) = $$



(7)

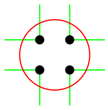
Here each dot represents a virtual particle, and the lines connecting them represent that they are in an entangled (here the maximally entangled) state. The red circles depict the operators acting on the four virtual particles. We call a PEPS *injective* if these operators are injective maps.

In the following, we use this notation when drawing tensor networks in two dimensions. For example, a four-partite state will be depicted as



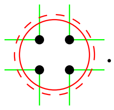
(8)

This four-partite state can equally be thought of as a non-translationally invariant MPS on four sites. Then each corner depicts an MPS tensor. Operators are depicted as circles or rectangles. For example,



(9)

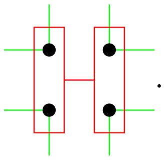
depicts a four-partite operator acting on the physical indices (black points) of four MPS tensors. In certain cases, we need more than two layers of tensors. In this case, the upper layer is drawn bigger. For example, a product of two operators acting on four MPS tensors is depicted as



(10)

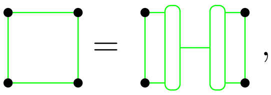
In this case, the operator depicted as a solid circle acts first on the four MPS tensors and the dashed one acts second.

We will often use a minimal rank decomposition of tensors. For convenience, we will refer to the operators in the decomposition as Schmidt vectors, even though the minimal rank decomposition is not necessarily a Schmidt decomposition, as we do not require orthogonality. For example, a minimal rank decomposition of a four-partite operator acting on four MPS tensors will be depicted as



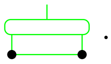
(11)

The Schmidt vectors of a four-partite state $|\phi\rangle$ can be related to its MPS description. We will therefore depict the minimal rank decomposition as



(12)

where the Schmidt vectors are denoted as



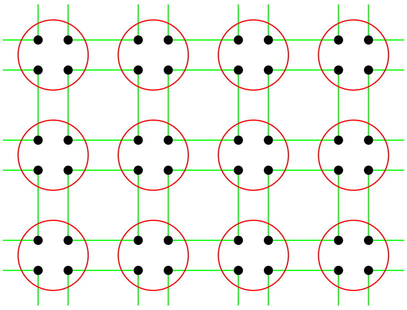
(13)

B. Semi-injective PEPS

In this section, we define semi-injective PEPSs. We show that this class contains all injective PEPSs. Moreover, we provide examples that are not known to admit an injective PEPS description, yet they can be written as semi-injective PEPSs.

Definition 1 (Semi-injective PEPS). Let $|\phi\rangle$ be a four-partite state with full rank reduced densities at every site, and let O be an invertible operator. The *semi-injective* PEPS $|\Psi_{n\times m}(\phi, O)\rangle$ is defined as

$$|\Psi_{n\times m}(\phi, O)\rangle =$$

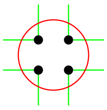


$$, \tag{14}$$

where the green rectangle is $|\phi\rangle$ and the red circle is O, and the state is defined on a torus with $n \times m$ copies of $|\phi\rangle$. We will often drop $|\phi\rangle$ and O and the indices n, m from the notation.

Note that the full rank assumption does not affect which states can be described as semi-injective PEPSs, it is only needed to avoid unnecessary degrees of freedom in the operator O.

These states can be written as a PEPS, but that PEPS is in general not injective. For example, a PEPS tensor generating the same state is



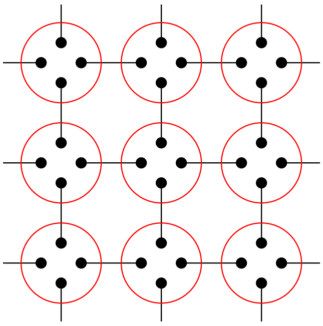
(15)

with an arbitrary (not necessarily translational invariant) MPS decomposition of $|\phi\rangle$ (TN states of this form have been used in other contexts, see, e.g., Ref. 41).

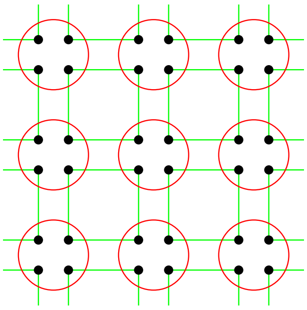
In the following, we show that these states include injective PEPSs as well as all the examples mentioned above.

1. *Injective PEPS*

In this case, the invertible operator is the PEPS tensor, and the four-partite state consists of two maximally entangled pairs (and a one-dimensional particle),

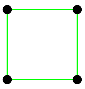


$$=$$

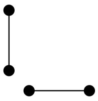


(16)

where the four-partite states are (the fourth particle is a scalar)



$$=$$



(17)

2. The CZX model

The CZX model readily admits the semi-injective PEPS form: the four-partite states are GHZ states $|0000\rangle + |1111\rangle$, whereas the invertible operators are unitary operators $U_{CZX} = X^{\otimes 4} \cdot \prod_{\langle ij\rangle} CZ_{ij}$.

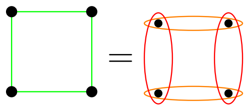
3. Purification of thermal states

Consider a commuting nearest neighbor Hamiltonian on a square lattice and the following purification of its Gibbs state:

$$|\Psi\rangle = \frac{1}{\sqrt{Z}} \left(e^{-\beta H/2} \otimes \mathrm{Id}\right) \bigotimes_i |\phi^+\rangle_i\,, \tag{18}$$

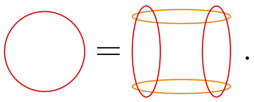
where $|\phi^+\rangle = \sum_j |jj\rangle$ is a maximally entangled pair state, and $e^{-\beta H} \otimes \mathrm{Id}$ acts non-trivially on one of the entangled pairs at every lattice site. This state admits a PEPS description: as the Hamiltonian terms are commuting, $e^{-\beta H/2}$ is a product of local operators. The PEPS tensors are then simply the product of the Schmidt vectors of these local operators acting on $|\phi^+\rangle$. This tensor does not become injective after blocking exactly because of the corners: after blocking, the operators lying entirely inside the blocked region can be inverted. Note that applying invertible operators on the tensor does not change injectivity. This factorizes the tensor into a product of tensors on the boundary and tensors on the corners. The boundary is injective, while the corners are not: the Schmidt vectors of the Hamiltonian terms commute; therefore, any antisymmetric state on the corner is mapped to zero.

Nevertheless, these states admit a semi-injective PEPS representation. Indeed, block 2×2 pairs of particles. The four-partite state in the semi-injective PEPS description consists of the four pairs of particles in the state



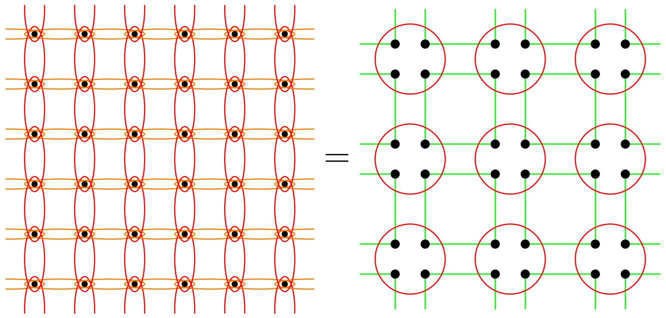
$$= \left(\prod_{\langle ij\rangle} e^{-\beta h_{ij}}\right) \otimes \mathrm{Id}^{\otimes 4} \cdot \bigotimes_{i=1}^{4} |\phi^+\rangle_i, \tag{19}$$

where both i and j run over the particles in one block and $|\phi^+\rangle$ is the maximally entangled state. The dots represent $|\phi^+\rangle$, while the ellipses represent $e^{-\beta h_{ij}} \otimes \mathrm{Id}^{\otimes 2}$. The invertible operator is a product of $e^{-\beta h_{ij}} \otimes \mathrm{Id}^{\otimes 2}$ on the 2×2 block shifted by half a lattice constant in both directions,



(20)

With this convention, the state is written as an injective PEPS as follows:

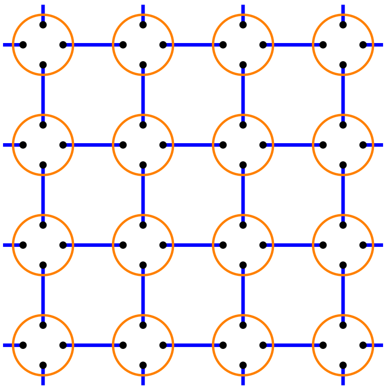


(21)

4. AKLT in two dimensions

The two-dimensional AKLT model is a spin-2 system which is constructed as follows: place a singlet $|01\rangle - |10\rangle$ on each edge of a square lattice. Then, at each vertex, project the four virtual qubits into the 5-dimensional symmetric subspace,

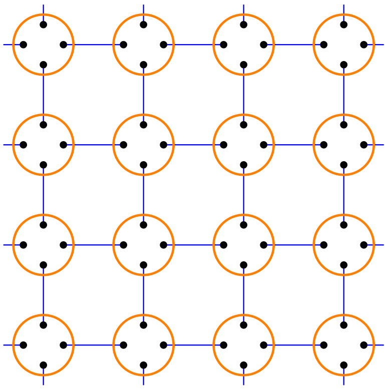
$$|\Psi_{AKLT}\rangle =$$



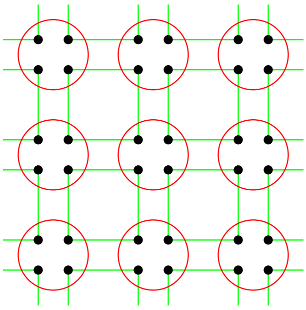
$$. \qquad (22)$$

Here the blue lines represent singlets and the orange circles represent the projectors into the symmetric subspace. As any virtual boundary state which is anti-symmetric on the two qubits of any corner is in the kernel of the map after appropriate applications of single-qubit Ys, the PEPS tensors describing this state cannot be injective, even after blocking.

The two-dimensional AKLT admits a semi-injective PEPS description as follows:

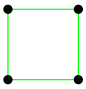


$$=$$

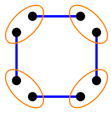


$$, \qquad (23)$$

with the four-partite state

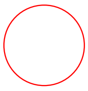


$$=$$

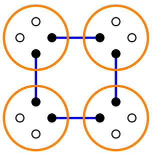


$$, \qquad (24)$$

where the blue lines are singlets, and the orange ellipses are the projectors into the 3-dimensional symmetric subspace (the four-partite state can thus effectively be viewed as a state on four qutrits: the one-dimensional AKLT state on four particles), and



$$= \qquad (25)$$



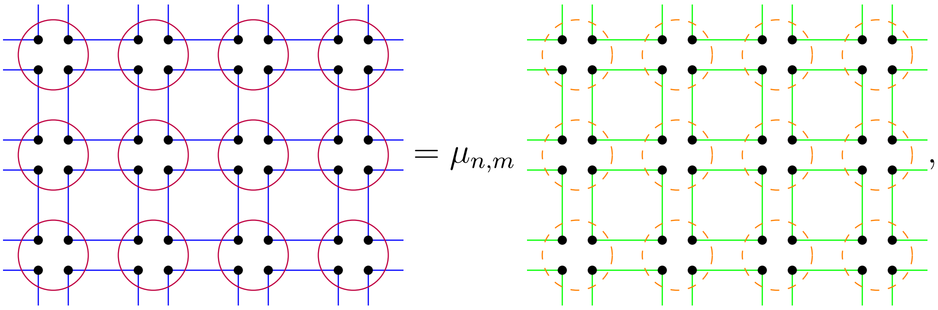
which acts on the qubits represented by the hollow dots, restricted to the symmetric subspace at each corner. It can be verified that the rank of (25) is 81. Clearly, the image of the adjoint of (25) is contained in the subspace which is symmetric in the pairs of qubits at each corner. The dimension of the latter is also 81. Thus, (25) is invertible.

III. SUMMARY AND RESULTS

In this section, we give a summary of the results obtained in this work. The detailed derivations of all these results are given in Secs. IV–VII. The results extend the properties known for injective PEPSs to semi-injective ones. First, we show how to construct local Hamiltonians for which they are the unique ground states. Next, we answer the question under which local conditions two semi-injective

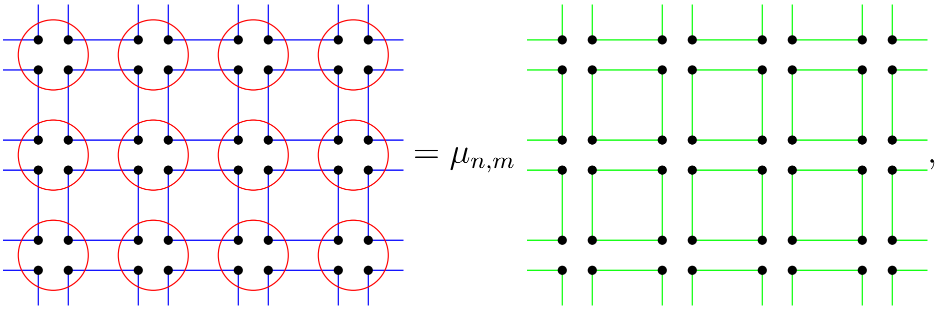
PEPSs generate the same state. We then use this result to characterize symmetries in semi-injective PEPSs. We also find that the third cohomology classification of SPT phases naturally extends to these states, and thus these states might be suitable to capture the physics of SPT phases. In the following, we give a detailed description of the results.

Consider two semi-injective PEPSs generated by (ϕ_A, O_A) and (ϕ_B, O_B). Suppose that on an $n \times m$ torus, they generate states that are proportional to each other,



(26)

where the purple circle and the blue rectangle depict O_A and $|\phi_A\rangle$, respectively, while the orange dashed circle and the green rectangle depict O_B and $|\phi_B\rangle$, respectively, and $\mu_{n,m} \in \mathbb{C}$. Inverting O_B, we obtain



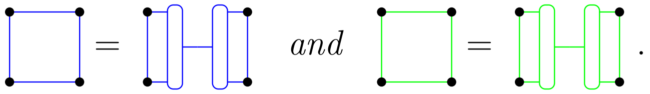
(27)

where the red circle denotes the invertible operator $O = O_B^{-1} O_A$.

In this setup, we prove the following:

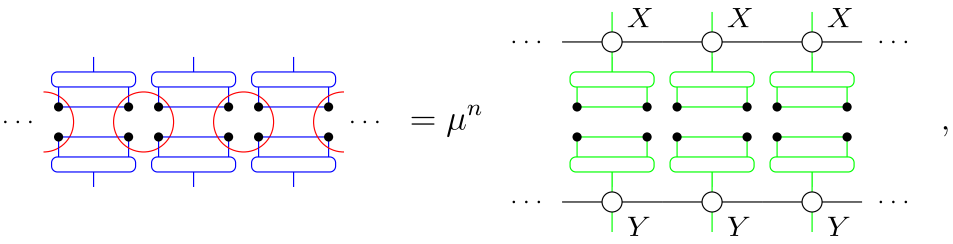
Theorem 1. *If Eq. (27) holds for some specific $n_0, m_0 \geq 3$, then for all $n, m \in \mathbb{N}$:*

1. *Equation (27) holds with $\mu_{n,m} = \mu^{nm}$.*
2. *The action of O corresponds to an MPO acting on the boundary as follows: Take a minimal rank decomposition of the four-partite states with respect to the vertical cut. That is, write*



(28)

Then for the Schmidt vectors defined as above, the following holds: There are two MPO tensors X and Y such that



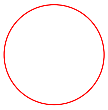
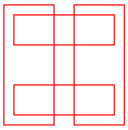
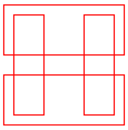
(29)

where $\mu \in \mathbb{C}$ is the proportionality constant from Point 1 above, $V_n(Y) = (V_n(X))^{-1}$ for every size n, and both X and Y become injective after blocking two tensors.

3. *The operator O is a four-particle non-translationally invariant MPO with the property that cutting the MPO into two halves yields a minimal rank decomposition of O.*
4. *The operator O is a product of two-body invertible operators,*

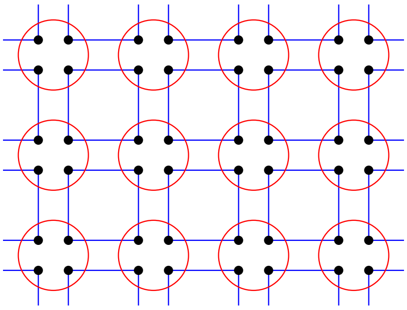
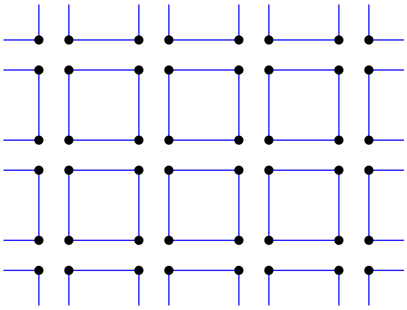
$$O = (O_{14} \otimes O_{23}) \cdot (O_{12} \otimes O_{34}) = \left(\tilde{O}_{12} \otimes \tilde{O}_{34}\right) \cdot \left(\tilde{O}_{14} \otimes \tilde{O}_{23}\right), \tag{30}$$

where the particles are numbered clockwise from the upper left corner and O_{ij} acts on particles i and j. Pictorially,

 $=$  $=$  . (31)

In Sec. VII, we use these results to rederive the third cohomology classification of SPT phases within the framework of semi-injective PEPSs. The setup in this case is as follows:

Let G be a group and O_g be a faithful (not necessarily unitary) representation of G. Let $|\phi\rangle$ be a four-partite state with full rank one-particle reduced densities. Suppose $\forall g \in G$, O_g is a symmetry of the semi-injective PEPS defined by $|\phi\rangle$ and Id,

 $= \mu^{nm}(g)$  , (32)

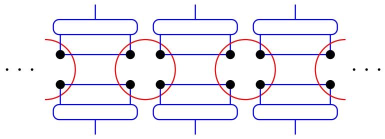
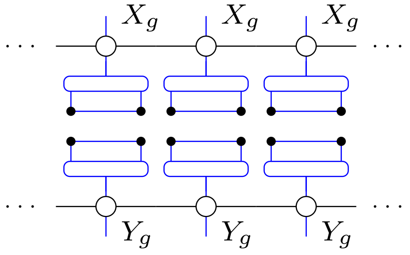
where the blue squares represent $|\phi\rangle$ and the red circles represent operators O_g.

Note that this setup can readily be applied for unitary symmetries of semi-injective PEPSs: let the semi-injective PEPS be defined by the four-partite state $|\phi\rangle$ and an invertible operator A. Let the unitary representation of the symmetry group G be U_g. Then, by inverting A in the symmetry condition, we arrive to Eq. (32) with $O_g = A^{-1}U_gA$.

Within this setup, we prove that

Theorem 2. *If Eq. (32) holds for some n, $m \geq 3$, then*

1. *$g \mapsto \mu(g)$ is a one-dimensional representation of G.*
2. *For every $g \in G$, there are two MPO tensors X_g and Y_g such that*

 $= \mu^n(g)$ 

, (33)

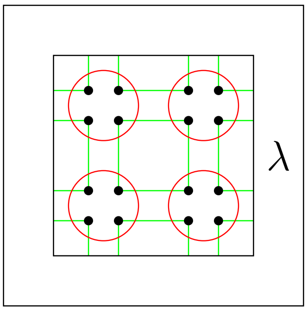
and $V_n(Y_g) = \left(V_n(X_g)\right)^{-1}$ for all n. Moreover, $V_n(X_g)$ and $V_n(Y_g)$ form projective representations of G with $V_n(X_g)V_n(X_h) = \lambda^n(g, h)V_n(X_gX_h)$ for a two-cocycle λ. In particular, $V_n(X_g)V_n(X_h)$ has only one block in its canonical form.

3. *There is a canonical way to assign an element from $H^3(G, \mathbb{C}^*)$ to the one-block MPO representation $g \mapsto X_g$.*

4. *O_g has a four-particle non-translationally invariant MPO representation with tensors $O_g^{(1)}, O_g^{(2)}, O_g^{(3)}, O_g^{(4)}$ in the sense of Theorem 1, Point 3, such that the MPO $V_n(O_g^{(1)}O_g^{(2)}O_g^{(3)}O_g^{(4)})$ forms a one-block projective MPO representation and its cohomology label coincides with that of the boundary. In particular, the MPO labels obtained from the vertical and horizontal boundaries are the same.*

IV. PARENT HAMILTONIAN

In this section, we prove that semi-injective PEPSs are unique ground states of their parent Hamiltonian. Let us consider a semi-injective PEPS $|\psi\rangle$. Corresponding to this state, we consider two parent Hamiltonian constructions. First, one can obtain the usual parent Hamiltonian by writing the state as a PEPS with the tensors in Eq. (15). That is, consider a 2×2 patch of the tensors. Let S be the subspace generated by the tensors with arbitrary boundary conditions:

$$S = \left\{ \vphantom{\Bigg|} \right.$$

$$\left. \lambda \;\middle|\; \lambda \in \mathbb{C}^{D_v^4 D_h^4} \right\}. \tag{34}$$

The Hamiltonian term $\tilde{h}_i$ centered around the plaquette state at position i is just the projector onto $S^\perp$ and the Hamiltonian $\tilde{H}$ is the sum over all positions of these projectors,

$$\tilde{h}_i = \mathrm{Proj}\left(S^\perp\right)_i \otimes \mathrm{Id}, \tag{35}$$

$$\tilde{H} = \sum_i \tilde{h}_i. \tag{36}$$

The second construction is to invert the operators O around a plaquette state at site i and project ϕ_i to zero,

$$h_i = \left(\prod_{\langle ji \rangle} O_j^{-1}\right)^\dagger P_i \left(\prod_{\langle ji \rangle} O_j^{-1}\right), \tag{37}$$

where j runs over all positions of operators that (partially) act on the plaquette state at position i and the projector P_i is the projector to the orthocomplement of $\mathbb{C}\,|\phi\rangle$: $P_i = (\mathrm{Id}_i - |\phi\rangle_i\langle\phi|) \otimes \mathrm{Id}$. Then the Hamiltonian H is the sum of the different terms,

$$H = \sum_i h_i. \tag{38}$$

Proposition 3. The semi-injective PEPS $|\psi\rangle$ is the unique ground state of both H and $\tilde{H}$ at all system sizes.

Proof. We first prove that H has a unique ground state. Then we prove that the kernel of $\tilde{H}$ is contained in that of H.

To see that $\ker H$ is one-dimensional, consider the following similarity transform:

$$\left(\prod_j O_j\right)^\dagger H \left(\prod_j O_j\right) = \sum_i P_i \otimes \mathrm{Id} \otimes \bigotimes_j \left(O_j^{-1}\right)^\dagger O_j^{-1}, \tag{39}$$

where the product runs over all sites j that are not neighbors of the projector P_i, and the identity acts on all virtual particles that are neighbors of the four-partite state $|\phi\rangle$. The kernel of each term in the sum is $|\phi\rangle_i \otimes \bigotimes_{j \notin i} \mathcal{H}_j$, where j runs over all virtual particles that are not in the four-partite state $|\phi\rangle$. Clearly the intersection of these subspaces is $\bigotimes_i |\phi\rangle_i$, that is, the kernel of H is one-dimensional.

To see that $\ker \tilde{H} \leq \ker H$, notice that every state in S_i [defined in Eq. (34)] is in the kernel of h_i. Therefore,

$$\ker \tilde{h}_i \leq \ker h_i. \tag{40}$$

Finally, as $\ker H = \cap_i \ker h_i$ and $\ker \tilde{H} = \cap_i \ker \tilde{h}_i$, the inclusion also holds for the kernel of the total Hamiltonians. □

V. BACKGROUND: MATRIX PRODUCT STATES

In this section, we recall some basic properties of MPSs. These definitions and theorems are mainly covered in Ref. 26. First, recall some basic properties of completely positive maps.

Definition 2. A completely positive map $T : \rho \mapsto T(\rho) = \sum_i A_i \rho A_i^\dagger$ is

- *irreducible* if there is no non-trivial projector P such that $T(\rho) = PT(\rho)P^\dagger$ for all $\rho = P\rho P^\dagger$, otherwise T is reducible;
- *primitive* if $\exists n$ such that $T^n(\rho) > 0$ for all $\rho \geq 0$.

Note that then the following statements hold:

Proposition 4. Let $T : \rho \mapsto T(\rho) = \sum_i A_i \rho A_i^\dagger$ be a completely positive map with spectral radius r. Then r is an eigenvalue with at least one positive semidefinite eigenvector. Moreover,

- *T is primitive if and only if r has multiplicity one, the corresponding eigenvector is positive definite, and there are no other eigenvalues of magnitude r.*
- *if T is irreducible but not primitive, then r has multiplicity one, and all eigenvalues of magnitude r are $r \cdot \exp\,[2\pi i n/K]$ for some K and $n = 1, 2, \ldots, K$. We call K the periodicity of T.*
- *T is reducible if and only if $A^i P = PA_i P$ for some non-trivial projector P.*

For proofs, see, e.g., Refs. 42 and 43. Now we define matrix product states.

Definition 3. An *MPS tensor* is a tensor $A \in \mathbb{C}^D \otimes (\mathbb{C}^D)^* \otimes \mathbb{C}^d$,

$$A = \sum_{i\alpha\beta} A^i_{\alpha\beta} \, |\alpha\rangle \langle\beta| \otimes |i\rangle = \sum_i A^i \otimes |i\rangle . \tag{41}$$

For any $n \in \mathbb{N}$, the state $V_n(A) \in (\mathbb{C}^d)^{\otimes n}$ is then defined as

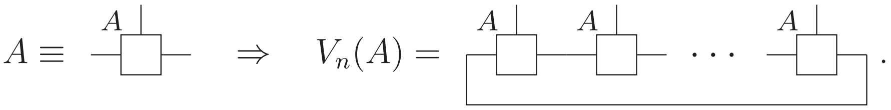
$$V_n(A) = \sum_{i_1 \ldots i_n} \mathrm{Tr}\left\{A^{i_1} \ldots A^{i_n}\right\} |i_1 \ldots i_n\rangle . \tag{42}$$

The *transfer matrix* of A, T_A, is the completely positive map $T_A : \rho \mapsto \sum_i A_i \rho A_i^\dagger$. We say that A is

- *injective*, if $\sum_i \mathrm{Tr}\{A^i \rho\}|i\rangle = 0$ implies $\rho = 0$;
- *normal*, if T_A is primitive;
- *periodic*, if T_A is irreducible but not primitive.

An MPS is called *normal, injective, or periodic,* if it can be generated by a normal, injective, or periodic MPS tensor.

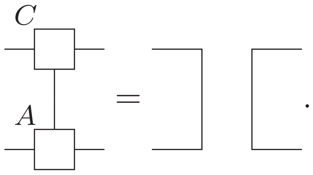
We often depict an MPS tensor and the corresponding MPS as follows:



(43)

The horizontal legs of the MPS tensor A are often referred to as the *virtual indices*, while the vertical one referred to as the *physical index* of A. The dimension of the virtual indices, D, is called the *bond dimension* of A.

Note that, unlike in Ref. 26, for convenience, we do not suppose that the spectral radius of a normal tensor is 1. Note also that an MPS tensor is injective if and only if it has a left inverse, C, such that $\sum_i A^i \otimes C^i = \mathrm{Id}$ in the sense as depicted,



(44)

Here, and in the following, we use the following graphical calculus[44] of tensors. A tensor is depicted as a box or circle, with some lines attached to it. These lines represent the indices of the tensor. Tensor contraction is depicted by joining the lines. In the picture above, for example, we have contracted the physical indices of A and C. The result is the identity tensor from the bottom indices to the top indices. We have omitted drawing a box for the identity.

A frequently used concept in MPS theory is the blocking of tensors.

Definition 4 (Blocking). The MPS tensor B is a *blocking* of A if $B = \sum_{i_1\dots i_k} A^{i_1}\dots A^{i_k} \otimes |i_1\dots i_k\rangle$. Note that $V_n(B) = V_{kn}(A)$.

We will often write the above contraction of tensors as a product. That is, for any two MPS tensors C and D, $CD := \sum_{ij} C^i D^j \otimes |ij\rangle$. With this notation, $B = AA\dots A$. We will use this notation even if one of the tensors does not have a physical index.

Note that a normal tensor stays normal after blocking. Moreover, injective and normal MPSs are the same up to blocking:

Proposition 5. Any injective tensor is proportional to a normal tensor. Conversely, for any normal tensor, $\exists L_0 \in \mathbb{N}$ such that it becomes injective after blocking any $L \geq L_0$ tensors. The minimal such L_0 is called the injectivity length.

This statement was proven, e.g., in Ref. 45. Note that L_0 might be bigger than the primitivity length of T_A, that is, the minimal n for which $T_A^n(\rho) > 0$ for all $\rho \geq 0$. There is, however, a universal bound depending only on the bond dimension D.

Note that being normal or injective are properties which are stable under taking the tensor product of MPS tensors:

Proposition 6. The tensor product of two normal MPS tensors is normal. The tensor product of two injective MPS tensors is injective.

Proof. First, we prove that the tensor product of two normal tensors A and B is normal. The transfer matrix of $A \otimes B$ is $T_A \otimes T_B$, where T_A is the transfer matrix of A and T_B is the transfer matrix of B. Denote the spectrum of any operator T by $\sigma(T)$. Then $\sigma(T_A \otimes T_B) = \sigma(T_A)\cdot\sigma(T_B)$. Therefore, $T_A \otimes T_B$ has a unique eigenvalue with magnitude (and value) equal to the spectral radius. The corresponding eigenvector of $T_A \otimes T_B$ is $\rho_A \otimes \rho_B$ if ρ_A and ρ_B are the eigenvectors of T_A and T_B with maximum eigenvalue, respectively. $\rho_A \otimes \rho_B$ is positive and is full rank, so $T_A \otimes T_B$ is primitive.

Second, the tensor product of two injective tensors is injective: if A and B are injective and A^{-1} and B^{-1} are their left inverses, then $A^{-1} \otimes B^{-1}$ is a left inverse of $A \otimes B$. □

Proposition 7. Given two normal tensors A and B with injectivity length at most L, the two MPSs generated by them either become perpendicular in the thermodynamic limit, i.e.,

$$\frac{|\langle V_n(A)|V_n(B)\rangle|}{\|V_n(A)\|\cdot\|V_n(B)\|} \to 0 \tag{45}$$

as $n \to \infty$, or the following three equivalent statements hold:

- $V_n(A) = \lambda^n V_n(B)$ *for some* $\lambda \in \mathbb{C}$ *for all n;*
- $\exists n \geq 2L + 1$ *such that* $V_n(A) = \lambda^n V_n(B)$ *for some* $\lambda \in \mathbb{C}$;
- $A^i = \lambda X B^i X^{-1}$, *for some* $\lambda \in \mathbb{C}$; *and this X is unique up to a constant.*

We call the normal tensors A and B *essentially different* if the MPSs generated by them are not proportional in the above sense. The proof of these statements can be found in Ref. 26.

Corollary 7.1. Given a set of pairwise essentially different normal tensors, A_i, $\exists N \in \mathbb{N}$ *such that the MPSs* $V_n(A_i)$ *are linearly independent for all* $n > N$.

Proposition 8. Any MPS $V_n(A)$ *can be decomposed into a linear combination of normal and periodic MPSs,*

$$V_n(A) = \sum_i \mu_i^n V_n(A_i), \tag{46}$$

where each A_i *is either normal or periodic.*

The proof can be found in Ref. 26. We provide a simplified proof here.

Proof. We prove this by induction on the bond dimension D. If $D = 1$, A_i is proportional to a normal MPS. Suppose now that the statement is true for all $D < D_0$. Consider an MPS tensor A with bond dimension D_0. If its transfer matrix T_A is irreducible, then A is either periodic or proportional to a normal MPS tensor. Otherwise, there exists a non-trivial projector P such that $A_i P = P A_i P$; see Proposition 4. Then $V_n(A) = V_n(PAP) + V_n(QAQ)$ with $Q = 1 - P$. Finally, the bond dimension of PAP (and QAQ) can be compressed to the rank of P (corr. Q): write $P = YX$ for some $X : \mathbb{C}^{D_0} \to \mathbb{C}^D$, $Y : \mathbb{C}^D \to \mathbb{C}^{D_0}$, $XY = \mathrm{Id}_D$. Then XAY generates the same MPS as PAP. The bond dimension of the resulting MPS is smaller than D_0; thus, by the induction hypothesis, they can be written as a linear combination of normal or periodic MPSs. □

Proposition 9. Let A be a periodic MPS tensor with periodicity K. After blocking K tensors, $V_n(A)$ *decomposes into K essentially different normal MPSs,*

$$V_{Kn}(A) = \sum_{i=1}^{K} V_n(B_i), \tag{47}$$

where the B_i*'s are pairwise essentially different normal MPS tensors on K spins. Moreover,* $V_n(A) = 0$ *if* $n \notin K\mathbb{N}$.

This statement has been proven as Lemma 5 in Ref. 46. Proposition 9 from Ref. 26 is a corollary of this:

Corollary 9.1. For any MPS tensor A $\exists K$ *such that after blocking K tensors,* $V_{Kn}(A)$ *decomposes into the following linear combination of normal tensors:*

$$V_{Kn}(A) = \sum_i \left(\sum_j \mu_{ij}^n \right) V_n(B_i), \tag{48}$$

where the B_i*'s are pairwise essentially different normal tensors on K sites.*

Finally, the following statement, together with Corollary 7.1, provides the "uniqueness" of this decomposition:

Proposition 10. If for $\mu_1, \ldots \mu_r \in \mathbb{C}\backslash\{0\}$ *and* $\lambda_1, \ldots \lambda_s \in \mathbb{C}\backslash\{0\}$

$$\sum_{i=1}^{r} \mu_i^n = \sum_{j=1}^{s} \lambda_j^n \tag{49}$$

for all $n \in \mathbb{N}$, *then* $r = s$ *and* $\mu_i = \lambda_{p(i)}$ *for some permutation p and for all i.*

This statement has been proven as Lemma 9 in Ref. 47.

We will also consider non-translationally invariant MPSs.

Definition 5. Let d_i and D_i $(i = 1, \ldots, k)$ be positive integers. Let $X_i = \sum_{j=1}^{d_i} X_i^j \otimes |j\rangle \in \mathbb{C}^{D_i} \otimes \left(\mathbb{C}^{D_{i+1}}\right)^* \otimes \mathbb{C}^{d_i}$ be tensors for $i = 1, \ldots, k$, where we identify $k+1$ with 1. Then the *non-translationally invariant* MPS defined by these tensors is

$$V(X_1, \ldots, X_k) = \sum_{i_1=1}^{d_1} \cdots \sum_{i_k=1}^{d_k} \mathrm{Tr}\left\{X_1^{i_1} \ldots X_k^{i_k}\right\} |i_1 \ldots i_k\rangle . \tag{50}$$

A non-translationally invariant MPS is called *injective after blocking l sites;* if $\forall i = 1, \ldots, k$, the tensor $X_i X_{i+1} \ldots X_{i+l}$ satisfies that if $\mathrm{Tr}\left\{\rho X_i^{j_1} X_{i+1}^{j_2} \ldots X_{i+l-1}^{j_l}\right\} |j_1 \ldots j_l\rangle = 0$, then $\rho = 0$.

Proposition 11. Let $X_1, \ldots, X_k$ define a non-translationally invariant MPS that is injective after blocking l sites. Then the MPS is also injective after blocking any $m \geq l$ sites.

Proof. We prove this by induction on m. For $m = l$, the statement is true by assumption. Suppose that the MPS is injective after blocking m tensors. Let $\rho \in \mathbb{C}^{D_{i+m}} \otimes \mathbb{C}^{D_i}$ such that for $m + 1$ consecutive sites

$$\sum_{j_1 \ldots j_{m+1}} \mathrm{Tr}\left\{\rho X_i^{j_1} X_{i+1}^{j_2} \ldots X_{i+m}^{j_{m+1}}\right\} \cdot |j_1 \ldots j_{m+1}\rangle = 0 \tag{51}$$

for some i. Then, as the tensor $X_i \ldots X_{i+m-1}$ is injective,

$$X_{i+m}^{j_{m+1}} \rho = 0 \quad \forall j_{m+1} \in \{1, 2, \ldots, d_{i+m}\}. \tag{52}$$

Take any matrix $M \in \mathbb{C}^{D_i} \otimes \mathbb{C}^{D_{i+1}}$. Then

$$0 = X_{i+1}^{j_2} \ldots X_{i+m}^{j_{m+1}} \rho M \in \mathbb{C}^{D_{i+1}} \otimes \mathbb{C}^{D_{i+1}}. \tag{53}$$

Then

$$\sum_{j_2 \ldots j_{m+1}} \mathrm{Tr}\left\{X_{i+1}^{j_2} \ldots X_{i+m}^{j_{m+1}} \rho M\right\} \cdot |j_1 \ldots j_{m+1}\rangle = 0. \tag{54}$$

The block of the m consecutive tensors $X_{i+1} \ldots X_{i+m}$ is injective; therefore, $\rho M = 0$. As M was arbitrary, $\rho = 0$; thus, the MPS is injective after blocking $m + 1$ sites. □

Finally, we introduce Matrix Product Operators (MPOs).

Definition 6. A matrix product operator is an operator written in MPS form

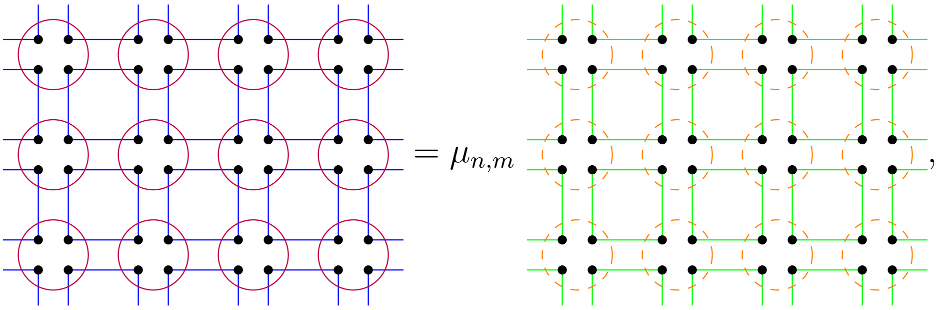
$$V_n(X) = \sum_{i_1 \ldots i_n j_1 \ldots j_n} \mathrm{Tr}\{X^{i_1 j_1} \ldots X^{i_n j_n}\} |i_1 \ldots i_n\rangle \langle j_1 \ldots j_n| . \tag{55}$$

As MPOs are just special MPSs, all the definitions and structure theorems above apply. In particular, we will use the terminology *normal, injective, and periodic* for MPOs too.

VI. CANONICAL FORM

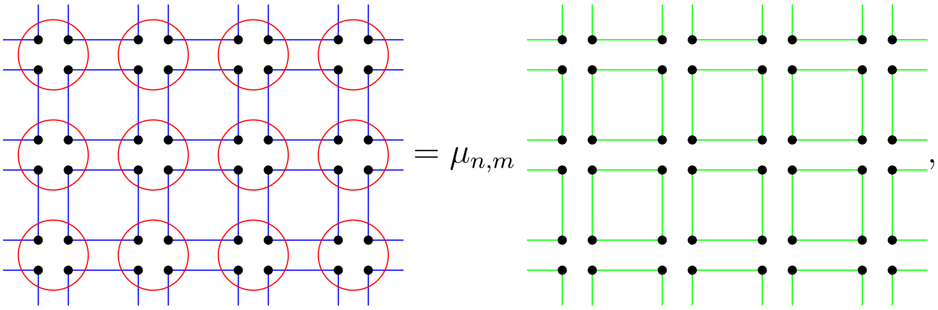
In this section, we investigate when two semi-injective PEPSs defined by (ϕ_A, O_A) and (ϕ_B, O_B) describe the same state for some (sufficiently large) system size. We find that this question can be decided locally: the two states are proportional for a large system size if and only if they are proportional on a 3 × 3 torus. Moreover, the boundary degrees of freedom are related by an invertible MPO whose inverse is also an MPO. Finally, we show that $O_B^{-1} O_A$ has to be a product of two-particle invertible operators. In Appendix A, we also provide some examples that explain why the situation is more complicated than in the case of injective PEPSs.

Consider two semi-injective PEPSs generated by (ϕ_A, O_A) and (ϕ_B, O_B). Suppose that on an $n \times m$ torus, they generate states that are proportional to each other,



(56)

where the purple circle and the blue rectangle depict O_A and $|\phi_A\rangle$, respectively, while the orange dashed circle and the green rectangle depicts O_B and $|\phi_B\rangle$, respectively, and $\mu_{n,m} \in \mathbb{C}$. Inverting O_B, we obtain

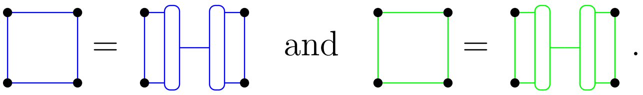


(57)

where the red circle denotes the invertible operator $O = O_B^{-1} O_A$. This equation is the starting point of our investigation below. First we prove that it holds for all system sizes:

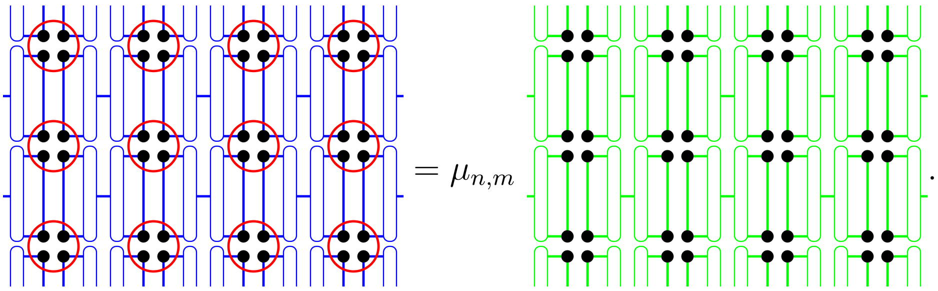
Proposition 12. If Eq. (57) holds for some $n_0 \geq 3$, $m_0 \geq 3$, then it also holds for any $n, m \in \mathbb{N}$ and the proportionality constant is $\mu_{n,m} = \mu^{nm}$.

Proof. Take a minimal rank decomposition of the four-partite states with respect to the vertical cut. That is, write



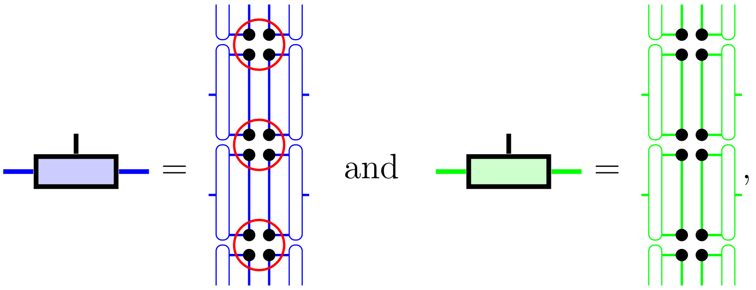
(58)

Using this decomposition, Eq. (57) reads as



(59)

This gives rise to an MPS description of the states with the following tensors:



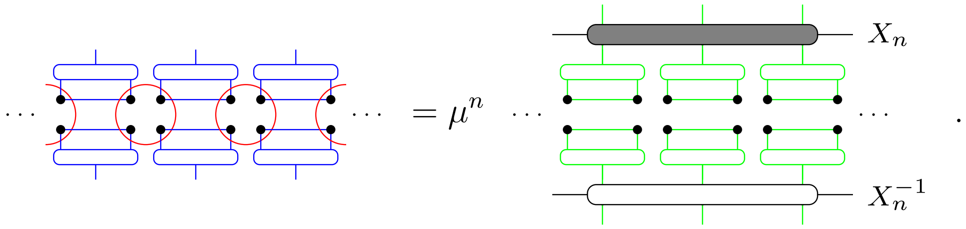
(60)

where the physical index of the MPS tensor is all physical indices of the virtual particles, while the virtual indices of the MPS correspond to the virtual indices of the minimal rank decomposition of the four-partite states. These tensors are injective: the green tensor is just a tensor product of the Schmidt vectors, and as the Schmidt vectors (and their tensor product) are linearly independent, that tensor is injective. The blue tensor is obtained by acting with an invertible operator on the tensor product of Schmidt vectors; therefore, it is also injective.

Thus, using Proposition 7, if Eq. (57) holds for $n_0 \geq 3$, $m_0 \geq 3$, then it also holds when the system size in the horizontal direction is changed to any n by keeping the system size in the vertical direction m_0. Therefore Eq. (57) holds for m_0 and any n, and the proportionality constant is $\mu_{n,m_0} = \mu_{m_0}^n$ for some $\mu_{m_0} \in \mathbb{C}$. The argumentation above holds with respect to the horizontal cut. Therefore the system size can be changed along the vertical direction too: as Eq. (57) holds for n, m_0, it also holds for n, m and the proportionality constant is then $\mu_{n,m_0}^{m/m_0} = \mu^{nm}$ for some $\mu \in \mathbb{C}$. □

Note that this implies that it is decidable whether two semi-injective PEPSs are equal for all system size. Moreover, it is also practically checkable: it is enough to calculate the overlap between two states (and their norms) on a 3 × 3 torus. The overlaps can be calculated by standard tensor network techniques. The cost of this computation scales as the 12th power of the Schmidt rank.

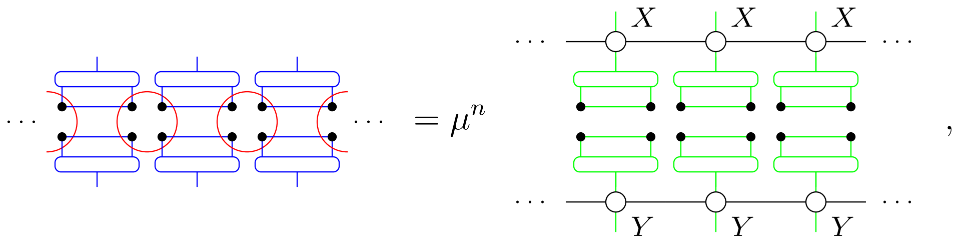
Using Proposition 7, we conclude that up to a constant there is a uniquely defined operator X_n on the boundary for which



(61)

This construction, however, does not reveal anything about the properties of the gauges X_n and X_n^{-1}: they are globally defined and the definition depends on the system size. In the following, we explore their structure and show that they can both be written as normal MPOs.

Theorem 13. *Suppose Eq. (57) holds for some $n, m \geq 3$. Then there are two MPO tensors X and Y such that*



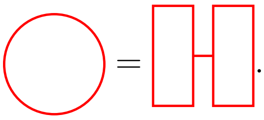
(62)

where $\mu \in \mathbb{C}$ is the proportionality constant from Proposition 12, and $V_n(Y) = (V_n(X))^{-1}$ for every size n and both X and Y become injective after blocking two tensors.

Before proceeding to the proof, notice that

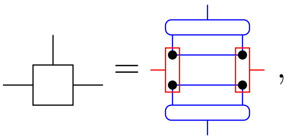
Lemma 14. The l.h.s. of Eq. (62) can be described by an MPS that becomes injective after blocking two tensors.

Proof. Take a minimal rank decomposition of the operators O,



(63)

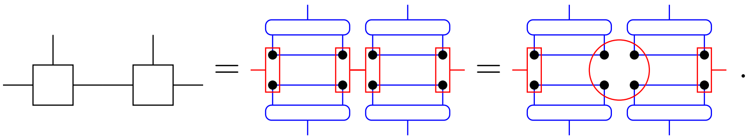
Then the l.h.s. of Eq. (62) is an MPS with the MPS tensor



(64)

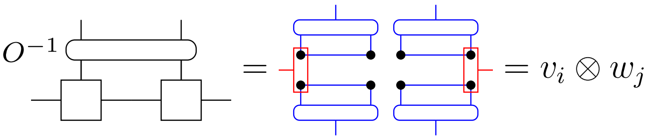
where the physical indices of the MPS are both the physical indices and the two virtual indices belonging to the decomposition of $|\phi_A\rangle$ on the r.h.s. of Eq. (64), while the virtual indices of the MPS are the virtual indices belonging to the decomposition of O on the r.h.s. of Eq. (64).

We prove now that this MPS tensor is injective after blocking two tensors. To see this, block two tensors and note that contracting the middle indices gives back O,



(65)

Inverting O does not change the injectivity of the MPS tensor, as it is an invertible operation on its physical indices. Therefore it is enough to prove that

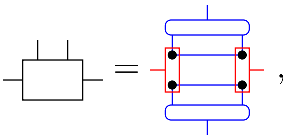


(66)

is injective. Both v_i and w_j are linearly independent, as the Schmidt vectors of O are linearly independent and the one body reduced densities of the four-partite states are full rank. Therefore the vectors $v_i \otimes w_j$ are also linearly independent, that is, the corresponding tensor is injective. □

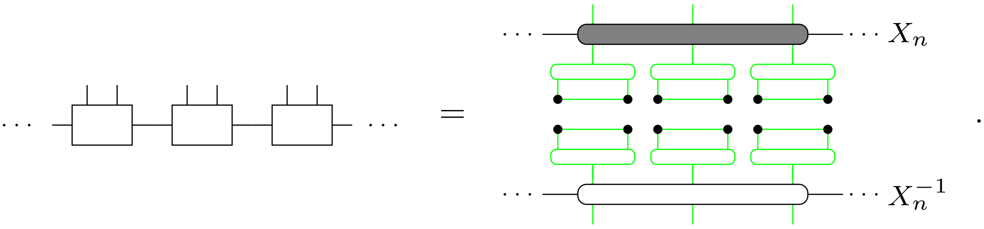
We now proceed to the proof of Theorem 13.

Proof of Theorem 13. We first prove that X_n and X_n^{-1} are proportional to an MPS. Write the l.h.s. of Eq. (61) as an MPS with two physical indices,



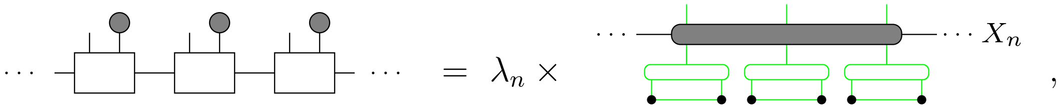
(67)

where the left physical index of the MPS tensor corresponds to the indices on the top of the r.h.s. (physical and virtual indices of the Schmidt vector), while the right one to the indices on the bottom of the r.h.s., and the virtual indices of the MPS correspond to the Schmidt index of the decomposition of O. With this notation, Eq. (61) reads as



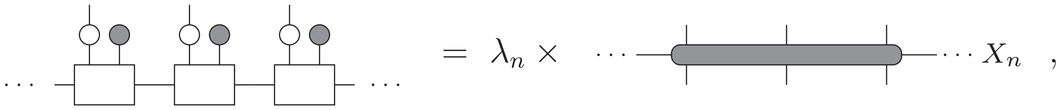
(68)

Applying a product linear functional on the lower half of the r.h.s. (and the right indices of the MPS on the l.h.s.), the equation changes to



(69)

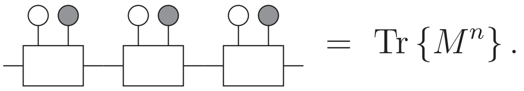
for some $\lambda_n \in \mathbb{C}$. Notice that the Schmidt vectors on the r.h.s. can be inverted: they are an injective mapping from the Schmidt index to the physical degrees of freedom, as they are linearly independent. Therefore,



(70)

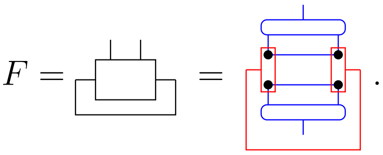
where the white circle depicts the inverse of the Schmidt vectors of $|\phi_B\rangle$. This shows that X_n (and similarly X_n^{-1}) is an MPS with some MPS tensor $\tilde{X}$ (and $\tilde{Y}$) as long as the l.h.s. is not 0. It is thus sufficient to prove that there is a translationally invariant product linear functional (the gray circles), which is independent of n, which does not map the l.h.s. to 0.

Consider two linear functionals acting on the MPS tensor,



(71)

We show now that there are linear functionals a, b such that for the corresponding $M_{a,b}\mathrm{Tr}\{M_{a,b}\} \neq 0$. Let us consider the map F: $(a, b) \mapsto \mathrm{Tr}\{M_{a,b}\}$. Graphically, this map is



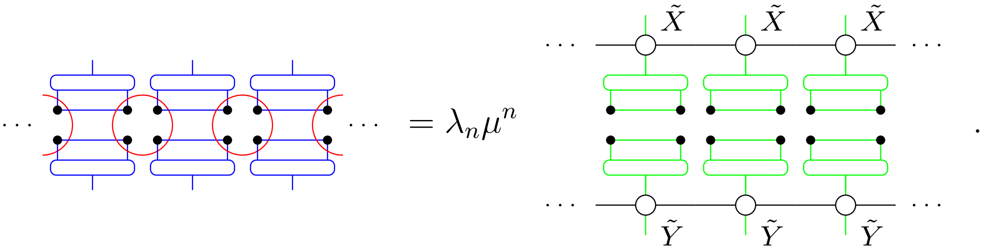
(72)

Notice that F equals to the operator O with the left and right side interchanged applied to the tensor product of the Schmidt vectors of $|\phi\rangle$. As O is invertible, F is not zero. Therefore there are linear functionals a, b such that $F(a, b) = \mathrm{Tr}\{M_{a,b}\} \neq 0$. As $\mathrm{Tr}\{M_{a,b}\} \neq 0$, $M_{a,b}$ is not nilpotent and thus

$$\mathrm{Tr}\{M_{a,b}^n\} = \sum_{i=1}^{R} \xi_i^n \tag{73}$$

for some $\xi_1, \ldots \xi_R \in \mathbb{C}\backslash\{0\}$, $R > 0$. Let $S := \{n \in \mathbb{N} \mid \mathrm{Tr}\{M_{a,b}^n\} \neq 0\}$. Notice that $|S| = \infty$. Then, choosing the linear functional appearing in Eq. (70) to be b, the l.h.s. is non-zero for all system sizes $n \in S$. Therefore, X_n can be written as an MPO for all $n \in S$. Similarly, using the linear functional a instead of b on the lower part of Eq. (68), we arrive to the conclusion that X_n^{-1} is also a non-zero MPO for all $n \in S$, for the same S.

Therefore there is a $\lambda_n \in \mathbb{C}$ such that $\forall n \in S$,



(74)

Here, μ_n is the proportionality constant appearing in Eq. (62), and $V_n(\tilde{X})$ and $V_n(\tilde{Y})$ are translationally invariant MPOs on n sites, such that $V_n(\tilde{Y}) = \left(V_n(\tilde{X})\right)^{-1}/\lambda_n$. Their defining tensors, $\tilde{X}$ and $\tilde{Y}$, are independent of n. Note that the MPOs $V_n(\tilde{X})$ and $V_n(\tilde{Y})$ are defined for $\forall n \in \mathbb{N}$, but we have not yet proven that Eq. (74) holds for $n \notin S$.

In the following, we prove that Eq. (74) also holds $\forall n \in \mathbb{N}$ for some injective MPO $V_n(X)$, $V_n(Y)$ with $\lambda_n = 1$.

Using Corollary 9.1, there exists $K \in \mathbb{N}$ such that after blocking K tensors, both $V_n(\tilde{X})$ and $V_n(\tilde{Y})$ ($n \in K\mathbb{N}$) can be decomposed into a linear combination of normal MPOs. As the tensor product of normal MPSs is again a normal MPS (Proposition 6), $V_n(\tilde{X}) \otimes V_n(\tilde{Y})$ has a decomposition into normal MPOs that are tensor products. Denote these essentially different normal MPOs by $V_n(X_i) \otimes V_n(Y_i)$. That is, $\forall n \in K\mathbb{N}$,

$$V_n(\tilde{X}) \otimes V_n(\tilde{Y}) = \sum_{i=1}^{L} \sum_{j=1}^{M_i} \zeta_{ij}^n V_n(X_i) \otimes V_n(Y_i), \tag{75}$$

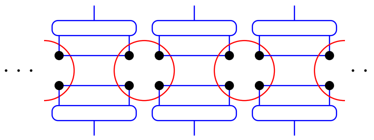
where $V_n(X_i) \otimes V_n(Y_i)$ are essentially different normal MPOs. Using this decomposition in Eq. (74), the l.h.s. is described by a normal MPO (Lemma 14), while the r.h.s. is described by the sum above for an infinite number of system sizes (indeed, for all $n \in K\mathbb{N} \cap S$). As essentially different MPSs become linearly independent for large system sizes (Corollary 7.1), Eq. (75) can describe a normal MPO only if either $L = 1$ or otherwise all but one i satisfy

$$\sum_{j=1}^{M_i} \zeta_{ij}^n = 0, \quad \forall n \in S \cap K\mathbb{N}. \tag{76}$$

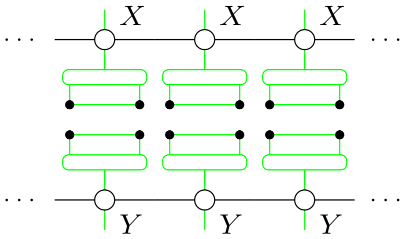
Recalling that Eq. (73) vanishes $\forall n \in \mathbb{N}\backslash S$, we conclude that

$$\sum_{k=1}^{R} \xi_k^n \sum_{j=1}^{M_i} \zeta_{ij}^n = \sum_{kj} (\xi_k \zeta_{ij})^n = 0 \quad \forall n \in K\mathbb{N}, \tag{77}$$

where i is chosen such that the sum of ζ_{ij}^n vanishes $\forall n \in S \cap k\mathbb{N}$. Applying Proposition 10 to Eq. (77), all $(\xi_k \zeta_{ij})^K = 0$, that is, $\zeta_{ij} = 0$ for all j and all but one i. Therefore, $L = 1$ in Eq. (75). Using Proposition 9, we conclude that $V_n(\tilde{X}) \otimes V_n(\tilde{Y})$ does not contain periodic MPOs; therefore $K = 1$. Thus, both the l.h.s. and the r.h.s. of Eq. (74) are proportional to normal MPOs. Using Proposition 7, we conclude that the equality in Eq. (74) holds $\forall n \in \mathbb{N}$. We have therefore proven that there are normal MPO tensors X and Y [the ones appearing in the unique normal MPO in Eq. (75)] such that $\forall n \in \mathbb{N}$ and some $\lambda_n \in \mathbb{C}$,

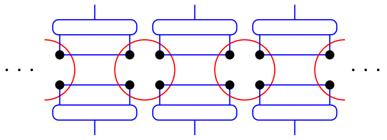


$$= \lambda_n \mu^n$$

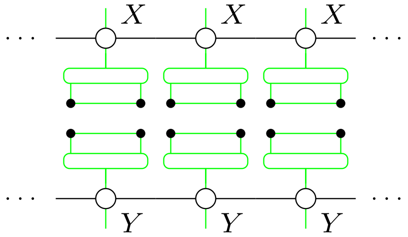


$$. \tag{78}$$

These MPO tensors also satisfy $V_n(Y) = (V_n(X))^{-1}/\lambda_n$ for all $n \in S$. As both $V_n(Y)$ and $V_n(X)$ are normal MPOs, the equality holds $\forall n \in \mathbb{N}$ and thus $\lambda_n = \lambda^n$ for some $\lambda \in \mathbb{C}$. Absorbing this constant into Y, $V_n(Y) = (V_n(X))^{-1}$ and



$$= \mu^n$$



$$. \tag{79}$$

□

Corollary 14.1. Suppose that $\forall n \in \mathbb{N}$ *Eq. (62) holds also for some other MPOs* $V_n(\tilde{X})$ *and* $V_n(\tilde{Y})$ *and* $V_n(\tilde{Y}) = \left(V_n(\tilde{X})\right)^{-1}$*. Then* $V_n(\tilde{X}) = \lambda^n V_n(X)$ *and* $V_n(\tilde{Y}) = \lambda^{-n} Y^{(n)}$ *for some* $\lambda \in \mathbb{C}$.

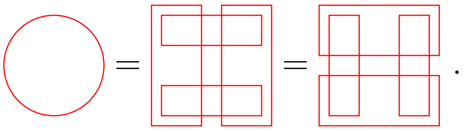
Proof. Due to the uniqueness of the gauge in Eq. (61), $V_n(\tilde{X}) = \lambda_n V_n(X)$ and $V_n(\tilde{Y}) = \lambda_n^{-1} V_n(Y)$. Decomposing $V_n(\tilde{X})$ and $V_n(\tilde{Y})$ to their canonical forms, we see that the only normal MPS appearing in the decomposition is $V_n(X)$ and $V_n(Y)$, and that $\lambda_n = \sum_i \lambda_i^n$ and $\lambda_n^{-1} = \sum_i \eta_i^n$. But then $1 = \sum_{ij}(\lambda_i \eta_j)^n$ and thus by Proposition 10, $\lambda_n = \lambda^n$. □

It turns out that the fact that the boundaries of the two semi-injective PEPSs are related by an MPO severely restricts the form of O. We will indeed find that

Proposition 15. The operator O *from Eq. (57) can be written as a product of invertible two-body operators,*

$$O = (O_{14} \otimes O_{23}) \cdot (O_{12} \otimes O_{34}) = \left(\tilde{O}_{12} \otimes \tilde{O}_{34}\right) \cdot \left(\tilde{O}_{14} \otimes \tilde{O}_{23}\right), \tag{80}$$

where the particles are numbered clockwise from the upper left corner and O_{ij} *acts on particles i and j. Pictorially,*

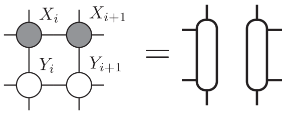


(81)

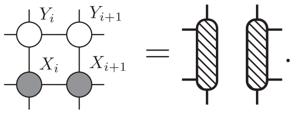
We will prove that O has a four site long non-translationally invariant MPO decomposition, with the property that cutting the MPO into two halves yields a minimal rank decomposition of O. Moreover, we will show that the product of the Schmidt vectors of O and O^{-1} is tensor products. Before proceeding to the proof, we show that if O and O^{-1} are both MPOs of this form, O has to have the two-layer structure (81).

Lemma 16. Consider two non-translationally invariant MPOs on $n = 2k$ *sites with tensors* $X_1, \ldots, X_n$ *and* $Y_1, \ldots, Y_n$*. Suppose that*

1. $V(X_1, \ldots, X_n) \cdot V(Y_1, \ldots, Y_n) = \mathrm{Id}$;
2. *both* $X_i X_{i+1}$ *and* $Y_i Y_{i+1}$ *are injective for all* $i = 1, \ldots, n$ *with* $n + 1 \equiv 1$;
3. *the product of* $X_i X_{i+1}$ *and* $Y_i Y_{i+1}$ *factorizes as depicted,*

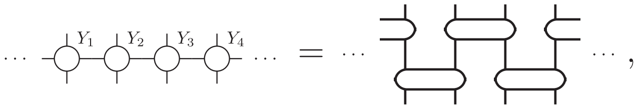


(82)



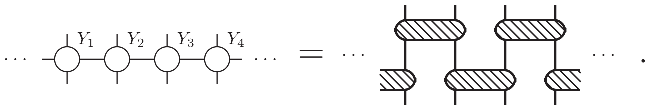
(83)

Then $V(X_1, \ldots, X_n)$ *(and* $V(Y_1, \ldots, Y_n)$*) admits a two layer description,*



(84)

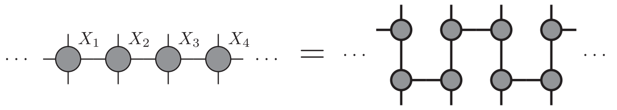
where all two-body operators on the r.h.s. are invertible. Equation (84) also holds when shifted by one site (with other invertible operators),



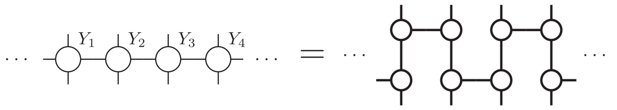
(85)

Note that for the translationally invariant setting, conditions 2 and 3 are satisfied naturally after blocking some tensors.

Proof. Take a Schmidt decomposition of the tensors $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ in an alternating way,

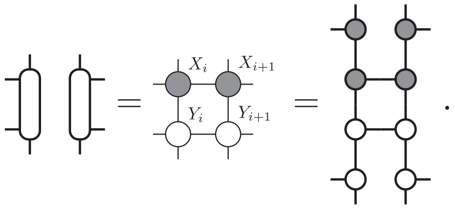


(86)



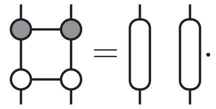
(87)

We will prove that the two-body operators defined this way are invertible. They naturally have to be injective from the outside to the middle indices, otherwise $V(X_1, \ldots, X_n)$ and $V(Y_1, \ldots, Y_n)$ would not be invertible. Suppose that there is an operator which is not injective from the middle to the outside. Suppose it happens in the lower layer of $V(X_1, \ldots, X_n)$. Consider a 2-site part of the MPO,



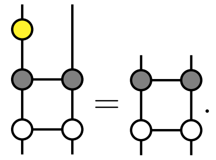
(88)

As we took a minimal rank decomposition, the outer tensors on the l.h.s. are invertible. Therefore, the product of the operators in the middle is a product,



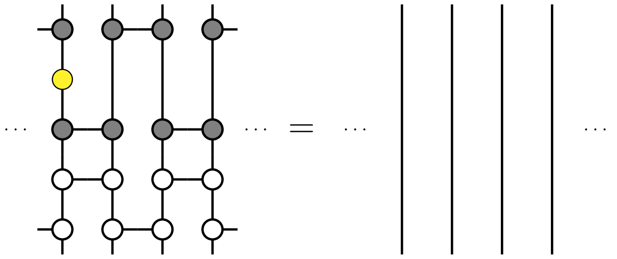
(89)

Therefore if the gray operator is not injective from top to bottom, then its kernel factorizes. Suppose the left operator on the r.h.s. has a non-trivial kernel. Then we can insert a non-trivial projector y on top that does not change the value of the product,



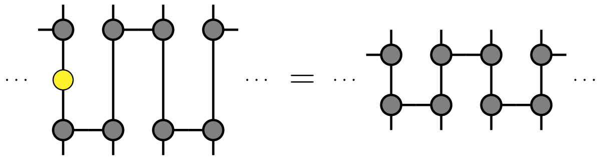
(90)

Inserting this back into the product $V(X_1, \ldots, X_n) \cdot V(Y_1, \ldots, Y_n)$, we get that



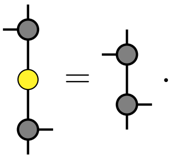
(91)

As $V(Y_1, \ldots, Y_n)$ is invertible, its left inverse is unique and equal to $V(X_1, \ldots, X_n)$. Therefore



(92)

By assumption, the tensors defining the MPO are injective after blocking at least two sites. Therefore, by inverting all but one tensor, we conclude that

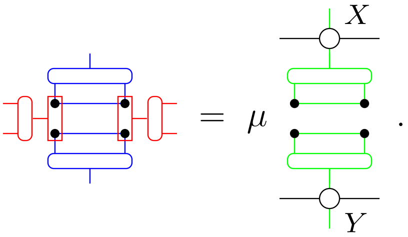


(93)

But this is not possible unless the yellow tensor is the identity. Thus, the two-body operators are invertible. □

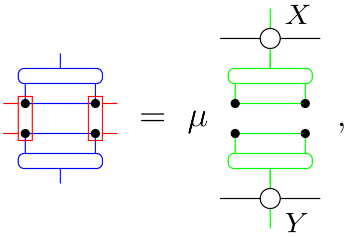
We now proceed to the proof of Proposition 15. Note that it is enough to show that both O and O^{-1} admit an MPO description that satisfies the conditions of Lemma 16.

Proof of Proposition 15. Write the l.h.s. of Eq. (62) as an injective MPS with tensors defined in Eq. (64). The r.h.s. of Eq. (62) is also an injective MPS. Therefore, the generating tensors are related by a gauge transformation,



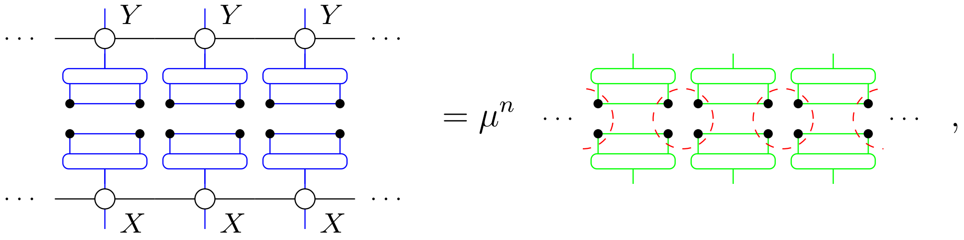
(94)

Absorbing the gauge in the decomposition of the operator, we have



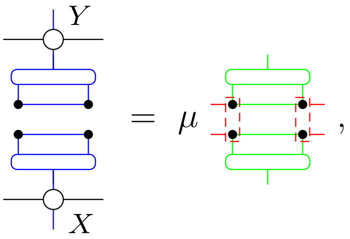
(95)

where the red rectangles depict a minimal rank decomposition of the operator O. As $V_n(X)$ and $V_n(Y)$ are inverses of each other, the inverse relation of Eq. (62) reads



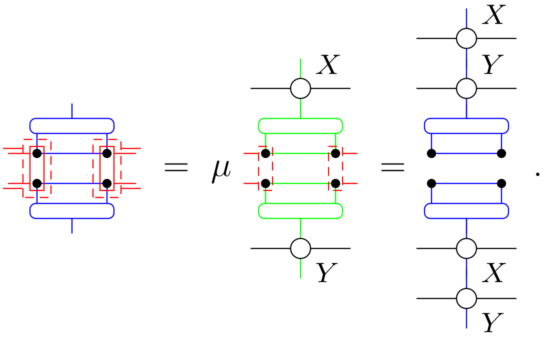
(96)

where the dashed red circles denote O^{-1}. Therefore, with an appropriate minimal rank decomposition of O^{-1}, the generating tensors are related as follows:



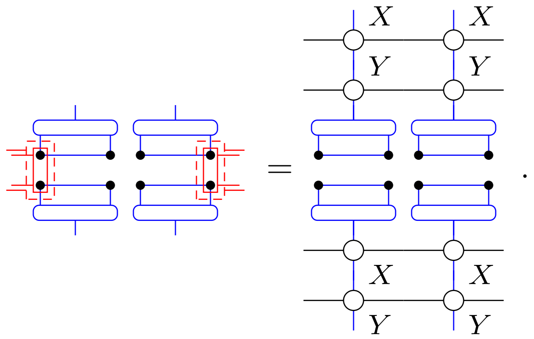
(97)

where the dashed rectangles denote the Schmidt decomposition of O^{-1}. Therefore, applying the Schmidt vectors of O and then O^{-1} to the Schmidt vectors of $|\phi_A\rangle$, we obtain



(98)

Contracting two copies of Eq. (98), the middle operator is OO^{-1} = Id, so

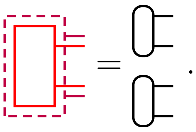


(99)

Notice that the l.h.s. is a product with respect to the vertical cut, whereas the r.h.s. is product with respect to the horizontal cut. Therefore both sides have to be product with respect to both vertical and horizontal cuts. Note that then $V_n(X)$ and $V_n(Y)$ satisfy the conditions of Lemma 16 and thus are products of invertible two-body operators in the sense of Eqs. (84) and (85). Similarly, both terms on the l.h.s. factorize with respect to the horizontal cut. As the one-body reduced densities of $|\phi_A\rangle$ are full rank, the product of the Schmidt vectors of O and O^{-1} factorizes

$$\left(O^{-1}\right)^{(13)}_{kl} O^{(13)}_{ij} = A^{(1)}_{ik} \otimes A^{(3)}_{jl}. \tag{100}$$

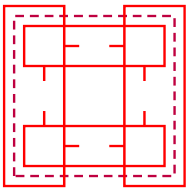
The same holds for the Schmidt vectors of all neighboring bipartition in any order. Similarly, the equation holds for the bipartition (13)–(24) and also for the reordering of O and O^{-1}. Equation (100) can be pictorially represented as



(101)

Consider the operator

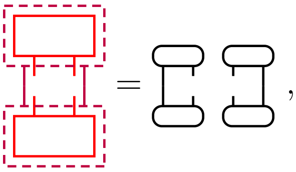
$$Z = \left(O^{(13)}_{j_1 j_2} \otimes O^{(24)}_{j_3 j_4}\right) O^{-1} \left(O^{(12)}_{i_1 i_2} \otimes O^{(34)}_{i_3 i_4}\right) =$$



. (102)

Note that Z factorizes with respect to the bipartition (13)–(24): to see this, decompose O^{-1} with respect to the bipartition (12)–(34). Then

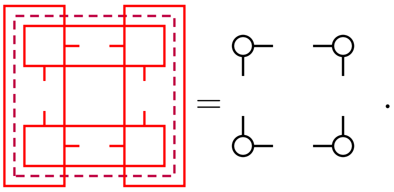
$$O^{-1}\left(O^{(12)}_{i_1i_2}\otimes O^{(34)}_{i_3i_4}\right) =$$



(103)

and therefore it factorizes with respect to the bipartition (13)–(24), and so does Z. Similarly, Z also factorizes with respect to the bipartition (12)–(34). Therefore, Z is a four-partite product,

$$Z =$$



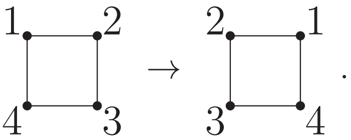
(104)

As contracting the open indices of Z gives back the operator $OO^{-1}O = O$, and as Z has a tensor product structure, this construction gives rise to an MPO description of O. Similarly, contracting only the vertical (horizontal) indices the lower (upper) two layers gives $O^{-1}O = \mathrm{Id}$ ($OO^{-1} = \mathrm{Id}$) on the lower (upper) two layers, and thus we obtain a minimal rank decomposition of O in the horizontal (vertical) cut. As the Schmidt vectors are linearly independent, the MPO tensors become injective after blocking two tensors.

The above construction can be repeated for O^{-1}. This leads to an MPO decomposition of O^{-1}.

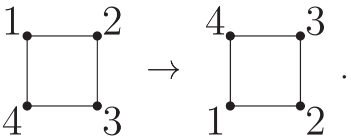
These two decompositions satisfy the conditions of Lemma 16: the MPOs become injective after blocking two tensors; moreover, the product of two neighboring tensors of O and O^{-1} factorizes. Therefore, O (and O^{-1}) is a product of invertible two-body operators. □

The above form provides an equivalent characterization of when two semi-injective PEPSs are equal for all system sizes. Before stating the theorem, we introduce two swap operators on four particles. The horizontal swap, H_A, exchanges the virtual particles of $|\phi_A\rangle$ in the horizontal direction,



(105)

The vertical swap, V_A, reflects the particles of $|\phi_A\rangle$ in the vertical direction,

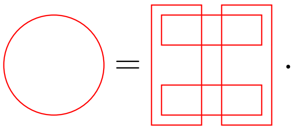


(106)

We denote the product of H_A and V_A as S_A: $S_A = H_AV_A = V_AH_A$. Define H_B, V_B, and S_B similarly for $|\phi_B\rangle$. Note that H_A and H_B are different in general as the Hilbert spaces of the virtual particles might differ.

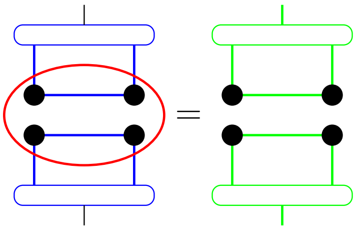
Theorem 17. *Two semi-injective PEPSs are equal [Eq. (57) holds] if and only if the following conditions are satisfied:*

- *The operator O factorizes into two-body operators as*

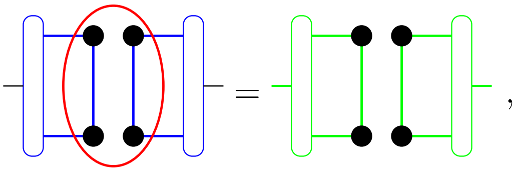


(107)

- *The Schmidt vectors of the four-partite states satisfy*



(108)

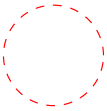


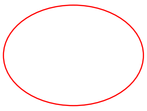
(109)

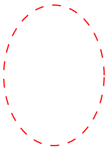
where the horizontal ellipse denotes $H_B O H_A$, and the vertical ellipse denotes $V_B O V_A$.

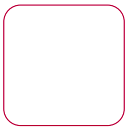
Note that the last two conditions are equivalent to the property that the two states are equal on an $n \times 1$ and a $1 \times n$ torus for all n, and therefore they are easily checkable.

Proof. The necessity of these conditions is clear from above. We now prove the sufficiency. Let

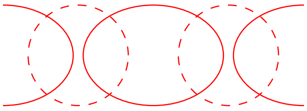
$$O^{-1} =$$  , (110)

$$H_B O H_A =$$  , (111)

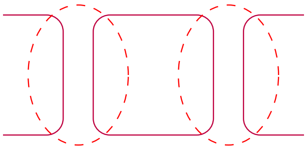
$$V_A O^{-1} V_B =$$  , (112)

$$S_B O S_A =$$  . (113)

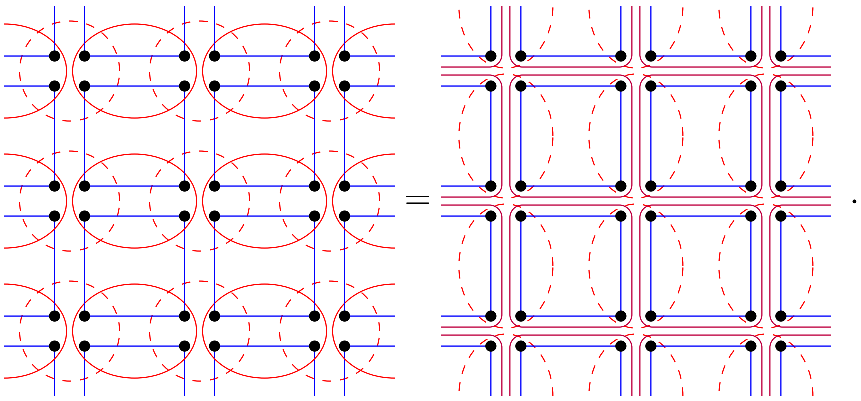
Due to the two layer structure of O and O^{-1} [Eq. (81)], the following operator is a product in the horizontal cut,

 $$= A \otimes B \ ,$$ (114)

where $H_B O H_A$ is the lower layer. The vertical swap of the previous operator is

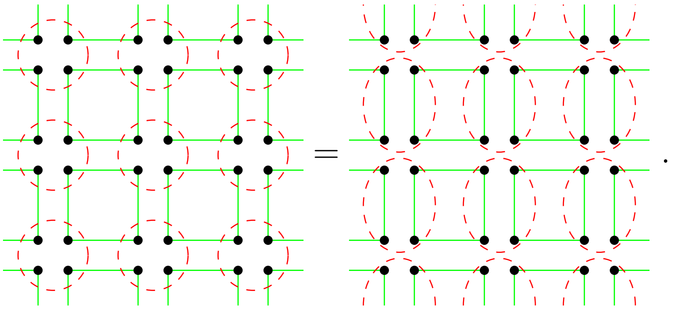
 $$= B \otimes A \ ,$$ (115)

where $S_B O S_A$ is the lower layer. Consider now these operators acting on the semi-injective PEPS defined by $|\phi_A\rangle$ and Id,



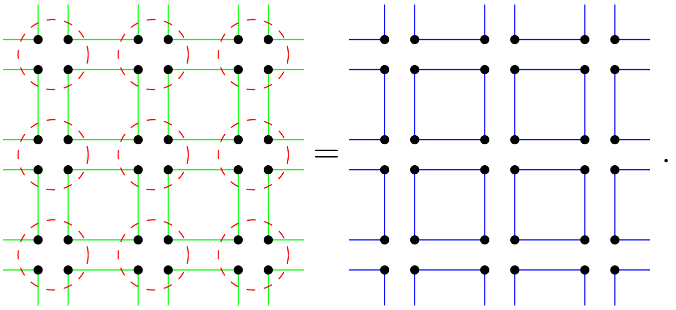
(116)

From Eqs. (108) and (109), the action of the lower layers on each side is to change $|\phi_A\rangle$ to $|\phi_B\rangle$. Therefore,



(117)

Using once more Eq. (109), the r.h.s. is a tensor product of ϕ_A at every position,



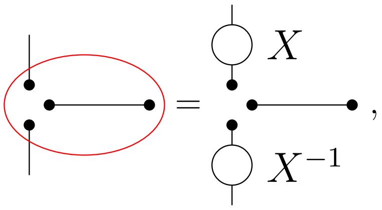
(118)

Applying O on both sides on each site, we see that Eq. (57) holds. □

As a simple application, one can derive the canonical form of injective PEPSs.[35]

Corollary 17.1. Two injective PEPSs generate the same state if and only if they are related by a product gauge transformation.

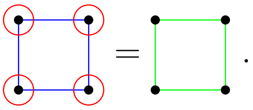
Proof. The conditions of Theorem 17 that Schmidt vectors map to Schmidt vectors read as



(119)

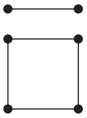
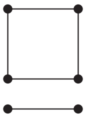
and therefore the operator O is a product on the two leftmost particles (and on one particle it is the inverse of the other). Similarly the other condition implies that the operator is a product on the two rightmost particles. Therefore O has a product structure in the desired form. □

We now show that if the span of the Schmidt vectors of both states with respect to both the vertical and horizontal cut contain product states, then $|\phi_A\rangle$ and $|\phi_B\rangle$ are stochastic local operations and classical communiation (SLOCC)[48]-equivalent, that is, there are invertible operators O_1, O_2, O_3, O_4 acting on the virtual particles such that $O_1 \otimes O_2 \otimes O_3 \otimes O_4|\phi_A\rangle = |\phi_B\rangle$. Pictorially,



(120)

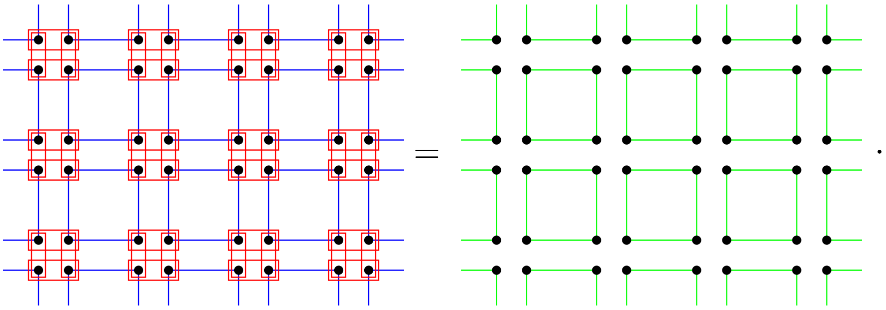
Note that there are examples for states that do not have product states in the span of their Schmidt vectors, but they generate the same state and are not SLOCC equivalent. For example, consider

$$|\phi_A\rangle =$$  , $$|\phi_B\rangle =$$  , (121)

then the semi-injective PEPS defined by $|\phi_A\rangle$ and Id and $|\phi_B\rangle$ and Id (more precisely the isomorphism that rearranges the tensor product to the right order) are the same on every torus, yet these states are not SLOCC equivalent.

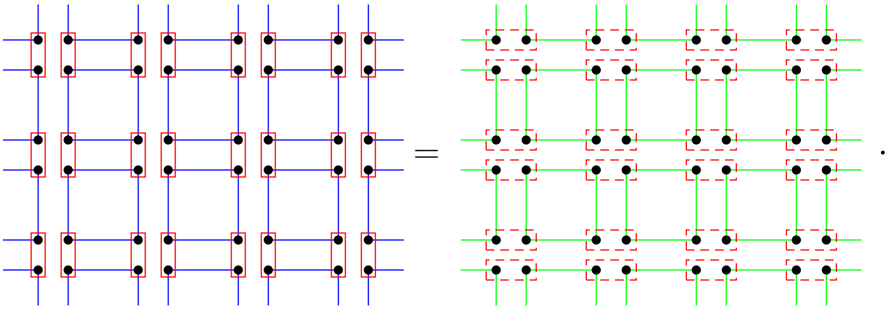
Theorem 18. *If the span of the Schmidt vectors of both four-partite states in Eq. (57) contains a product state for both the vertical and horizontal cut on both sides, then the two four-partite states are SLOCC equivalent.*

Proof. By Theorem 17, Eq. (57) implies



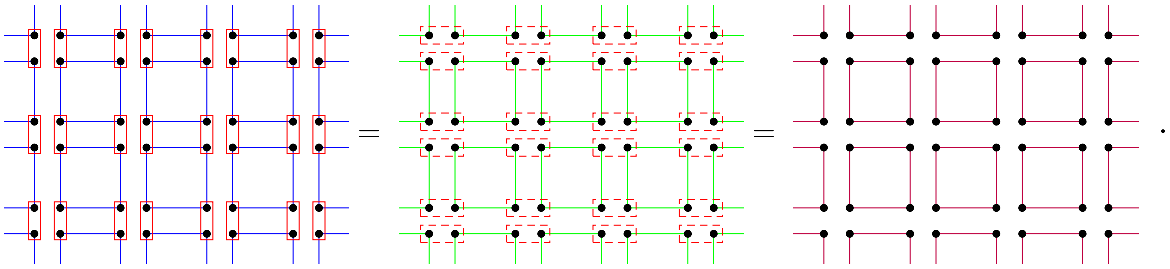
(122)

Inverting the upper layer, we get



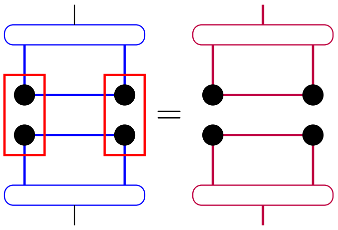
(123)

The l.h.s. is the product in the vertical direction, and the r.h.s. is the product in the horizontal direction. Therefore the two sides describe a state that factorizes in both directions. Let $|\xi\rangle$ be this state and denote this state with a purple square. Then,

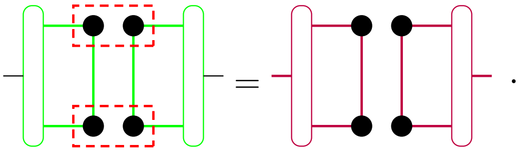


(124)

Equivalently, for the Schmidt vectors, we get

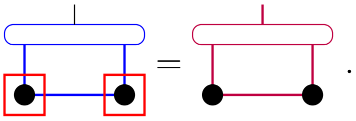


(125)



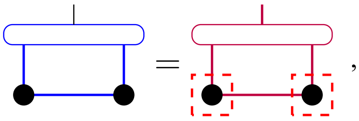
(126)

If the span of the Schmidt vectors on the l.h.s. contains a product vector, the same is true for the Schmidt vectors on the r.h.s. Therefore, choosing a product Schmidt vector on the bottom in Eq. (125) and applying a product linear functional, we get that for some not necessarily invertible operators,



(127)

A similar equation also holds for the lower part, as well as for both sides of $|\phi_B\rangle$. Inverting the operators appearing in Eq. (125), by the same argument, we obtain the inverse relation



(128)

and similarly along all other cuts. Equations (127) and (128) ensure that the one particle operators can be chosen invertible; thus, $|\phi_A\rangle$ and $|\xi\rangle$ are SLOCC equivalent. Similarly, $|\phi_B\rangle$ and $|\xi\rangle$ are SLOCC equivalent. Therefore $|\phi_A\rangle$ and $|\phi_B\rangle$ are SLOCC equivalent. □

Corollary 18.1. If two semi-injective PEPSs, defined by qubit four-partite states with genuine four-partite entanglement, are equal, then the four-partite states are SLOCC equivalent.

Proof. Notice that if the four-partite states are entangled for both the vertical and horizontal cut, then the span of the Schmidt vectors is at least two dimensional. As any two-dimensional subspace contains a product vector, the previous theorem applies. □

Based on Corollary 18.1, we provide a full classification of semi-injective PEPSs defined with four-partite qubit states in Appendix C.

VII. SPT PHASES

In this section, we show how the third cohomology labeling of the SPT phases[38,49] extends to semi-injective PEPSs. First, we show how to assign an element from the third cohomology group $H^3(G, \mathbb{C}^*)$ to a (projective) MPO representation of G. Here, and in the following, the action of G on $\mathbb{C}^*$ is trivial. Second, given a group of on-site symmetries of a semi-injective PEPS, there are three MPO representations associated with it: the boundary along the vertical cut, the boundary along the horizontal cut, and finally the symmetry operators themselves. We show that the associated third cohomology labels coincide. The importance of this statement is twofold. First, the labeling is encoded in the local operators already; thus, one does not have to look at the boundary of the system to find the labeling. Second, the labeling corresponding to the vertical and horizontal boundary coincides despite the model not necessarily having rotational symmetry.

A. Third cohomology labeling of MPO representations

Consider a group G and a projective MPO representation thereof, that is, a tensor X_g that generates an MPO $V_n(X_g)$ for all $g \in G$ such that $V_n(X_g)V_n(X_h) = \lambda_n(g, h)V_n(X_{gh})$ for all $g, h \in G$, where $\lambda_n(g,h) \in \mathbb{C}$. We will restrict ourselves to MPO representations for which $\lambda_n(g, h) = \lambda^n(g, h)$. We call such MPO projective representations *one-block projective MPO representations*. In this section, we show how to assign an element from the third cohomology group $H^3(G, \mathbb{C}^*)$ to such a representation.

We first show that we can suppose without loss of generality that X_g is normal. The proof is analogous to Theorem 13.

Lemma 19. Let $g \mapsto \tilde{X}_g$ be a one-block projective MPO representation of a group G, that is, $V_n(\tilde{X}_g)V_n(\tilde{X}_h) = \lambda^n(g,h)V_n(\tilde{X}_{gh})$ for some $\lambda(g,h) \in \mathbb{C}$. Then $\forall g \in G$ there is a normal tensor X_g such that $V_n(\tilde{X}_g) = V_n(X_g)$.

Proof. First we prove that $V_n(\tilde{X}_e) = \mu^n \mathrm{Id}$ for some $\mu \in \mathbb{C}$; therefore, there is a normal tensor X_e such that $V_n(\tilde{X}_e) = V_n(X_e)$. Then, as $V_n(\tilde{X}_g)V_n(\tilde{X}_{g^{-1}}) = \lambda^n(g, g^{-1})\mu^n \mathrm{Id}$, we will see that $V_n(\tilde{X}_g)$ can also be described with a normal MPO.

To see that $V_n(\tilde{X}_e) = \mu^n \mathrm{Id}$, notice that, as $g \mapsto V_n(\tilde{X}_e)$ is a representation, $V_n(\tilde{X}_e) = \mu_n \mathrm{Id}$ and that $V_n(\tilde{X}_e)V_n(\tilde{X}_e) = \mu_n^2 \mathrm{Id} = \mu_n \lambda^n(e,e)\mathrm{Id}$. Therefore, $\mu_n = \lambda^n(e, e)$.

Let K be such that after blocking K tensors, $V_n(\tilde{X}_g)$ and $V_n(\tilde{X}_{g^{-1}})$ can be decomposed into a sum of N and M normal MPOs, respectively. That is, $\forall n \in K\mathbb{N}$,

$$V_n(\tilde{X}_g) = \sum_{i=1}^{N} V_n(\tilde{X}_g^{(i)}), \tag{129}$$

$$V_n(\tilde{X}_{g^{-1}}) = \sum_{i=1}^{M} V_n(\tilde{X}_{g^{-1}}^{(i)}). \tag{130}$$

Then their product, $\lambda^n(g, g^{-1})\mu^n \mathrm{Id}$, can be decomposed into a sum of at least MN not necessarily essentially different normal MPOs,

$$\lambda^n(g, g^{-1})\mu^n \mathrm{Id} = \sum_{i=1}^{N}\sum_{j=1}^{M} V_n(\tilde{X}_g^{(i)})V_n(\tilde{X}_{g^{-1}}^{(j)}). \tag{131}$$

Let L be such that after blocking L tensors, all of these MPOs can be decomposed into normal MPOs: $\forall n \in KL\mathbb{N}$,

$$\lambda^n(g, g^{-1})\mu^n \mathrm{Id} = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{K_{ij}} V_n(Z_g^{ijk}), \tag{132}$$

for some normal tensors Z_g^{ijk}. If $i \neq i'$ or $j \neq j'$, Z_g^{ijk} and $Z_g^{i'j'k'}$ are not necessarily essentially different. Collecting the essentially different terms yields

$$\lambda^n(g, g^{-1})\mu^n \mathrm{Id} = \sum_{i=1}^{R}\sum_{j=1}^{S_i} \xi_j^n V_n(Z_g^i), \tag{133}$$

where R is the number of essentially different terms, Z_g^i are maximal pairwise essentially different subsets of Z_g^{ijk} and S_i is the multiplicity with which Z_g^i appears. Note that

$$\sum_{i=1}^{R} S_i = \sum_{i=1}^{N}\sum_{j=1}^{M} K_{ij}. \tag{134}$$

As essentially different normal MPOs become linearly independent for sufficiently large system sizes (Corollary 7.1), Proposition 10 implies that there can only be one term in this decomposition, that is, $R = 1$ and moreover $S_1 = 1$. As all $K_{ij} \geq 1$, we have $N = M = 1$ and thus $V_n(\tilde{X}_g)$ is normal. Therefore, $V_n(\tilde{X}_g)$ can be described by a normal MPO tensor X_g. □

The central tool in this section is comparing normal and non-normal MPS tensors that generate the same state. We only state the results here, and the proofs are provided in Appendix B.

Proposition 20. Let A be a normal MPS tensor and B be an MPS tensor such that for some $\lambda \in \mathbb{C}$

$$V_n(B) = \lambda^n V_n(A), \quad \forall n \in \mathbb{N}. \tag{135}$$

Then there exist matrices V, W *such that* $V\,W = \mathrm{Id}$ *and* $\forall n \in \mathbb{N}$ *and* $(i_1, i_2, \ldots, i_n) \in \{1, 2, \ldots, d\}^n$,

$$VB^{i_1} \ldots B^{i_n} W = A^{i_1} \ldots A^{i_n}. \tag{136}$$

Definition 7. The pair of operators V, W in Proposition 20 is called a *reduction* from B to A.

Proposition 21. Let V, W be a reduction from B to A. Let $N^i = B^i - WA^iV$. *Then the algebra generated by* N^i *is nilpotent.*

Definition 8. Let V, W be a reduction from B to A. Let $N^i = B^i - WA^iV$. Then the *nilpotency length* of the reduction is the minimal N_0 such that $\forall n \geq N_0$,

$$N^{i_1} \ldots N^{i_n} = 0. \tag{137}$$

The main statement is that any two reductions are related:

Theorem 22. *Let* V, W *and* $\tilde{V}, \tilde{W}$ *be two reductions from B to a normal tensor A. Let the nilpotency length of both reductions be at most* N_0. *Then* $\exists \lambda \in \mathbb{C}$ *such that for any* $n > 2N_0$,

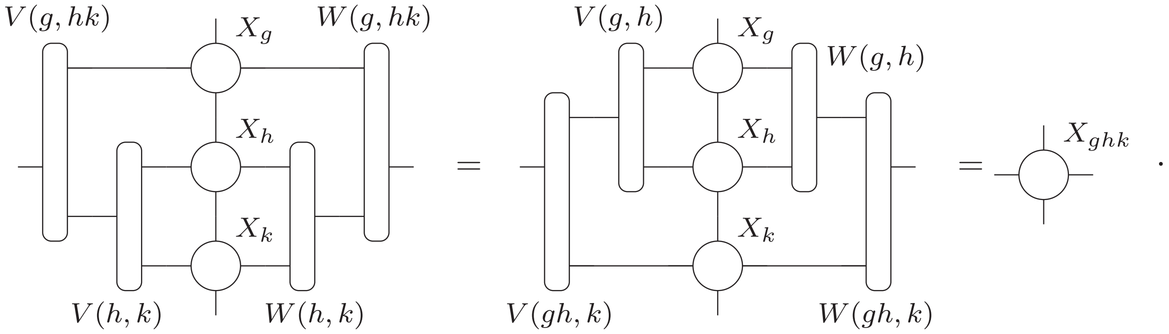
$$VB^{i_1}B^{i_2} \ldots B^{i_n} = \lambda \tilde{V} B^{i_1}B^{i_2} \ldots B^{i_n}, \tag{138}$$

$$B^{i_1}B^{i_2} \ldots B^{i_n} W = \lambda^{-1} B^{i_1}B^{i_2} \ldots B^{i_n} \tilde{W}. \tag{139}$$

Let us now continue how to assign an element of the third cohomology group to a one-block projective MPO representation. This discussion is essentially the same as in Ref. 38. We include here the construction for completeness.

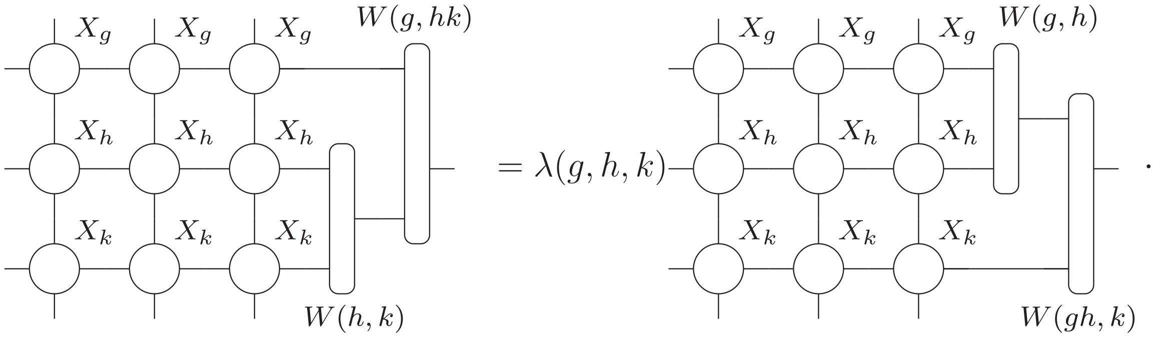
Let $X_{g,h} = \sum_{ijk} X_g^{ij} \otimes X_h^{jk} \otimes |i\rangle\langle k|$ be the MPO tensor describing the product of two MPOs. As $X_{g,h}$ and X_{gh} describe the same state and X_{gh} is injective, $X_{g,h}$ can be reduced to X_{gh} by Proposition 20. Let us fix such a reduction $V(g,h)$, $W(g,h)$ for any pair of group elements. We will assign a complex scalar to these reductions. We show that this scalar forms a three-cocycle. Different reductions then lead to different three-cocycles. We show, however, that their ratio forms a three-coboundary. Therefore, the equivalence class of the scalars is an element from the third cohomology group.

Starting from the reductions $V(g,h)$, $W(g,h)$, there are two natural ways to reduce the product of three MPOs,



(140)

By Theorem 22, there exists a complex scalar $\lambda(g,h,k)\in\mathbb{C}$ such that for any sufficiently long chain,



(141)

We show now that this scalar λ forms a three-cocycle due to associativity of the product. For the fixed reductions $V(g, h)$ and $W(g, h)$, denote the l.h.s. of Eq. (141) as $[g[hk]]$ and the r.h.s. as $[[gh]k]$. Consider a product of four MPOs, $ghkl$, and the following sequence of reductions:

$$[[[gh]k]l] \to [[gh][kl]] \to [g[h[kl]]] \to [g[[hk]l]] \to [[g[hk]]l] \to [[[gh]k]l]. \tag{142}$$

In this sequence, every member can be transformed to the next by changing the reduction of three consecutive group elements. Therefore, every member is related to the previous one by a scalar. Writing out these scalars, we obtain

$$[[[gh]k]l] = \underbrace{\lambda(gh,k,l)^{-1}\cdot\lambda(g,h,kl)^{-1}\cdot\lambda(h,k,l)\cdot\lambda(g,hk,l)\cdot\lambda(g,h,k)}_{=1}\cdot[[[gh]k]l]. \tag{143}$$

As this relation is the defining relation for the three-cocycles, $\lambda : G^3 \to \mathbb{C}^*$ is a three-cocycle, where G acts trivially on $\mathbb{C}^*$.

Note that the above construction depends on the fixed reductions $V(g, h)$, $W(g, h)$ of the product of two operators. In general, changing the reduction also changes the scalar. This change, however, is not arbitrary: we prove now that it forms a three-coboundary. Consider another reduction $\tilde{V}(g, h)$ and $\tilde{W}(g, h)$ with the corresponding three-cocycle $\tilde{\lambda}$. Then, denoting the reduction with $\tilde{V}(g, h)$ and $\tilde{W}(g, h)$ by round brackets (in the sense as above), using Theorem 22,

$$(gh) = \omega(g,h)[gh] \tag{144}$$

for some $\omega(g,h)\in\mathbb{C}$. Therefore, the two scalars λ and $\tilde{\lambda}$ are related as follows:

$$((gh)k) = \omega(g,h)\omega(gh,k)[[gh]k], \tag{145}$$

$$(g(hk)) = \omega(h,k)\omega(g,hk)[g[hk]]. \tag{146}$$

Therefore, the relation between λ and $\tilde{\lambda}$ is

$$\tilde{\lambda}(g,h,k) = \frac{\omega(g,h)\omega(gh,k)}{\omega(h,k)\omega(g,hk)}\lambda(g,h,k). \tag{147}$$

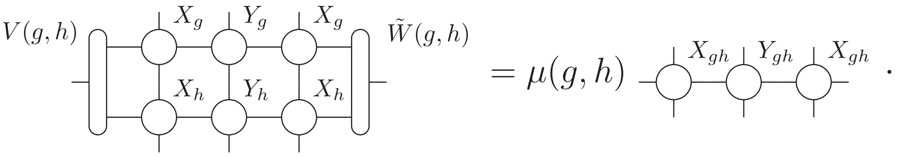
This is the defining relation of three-coboundaries; thus, $\lambda/\tilde{\lambda} : G^3 \to \mathbb{C}^*$ is a three-coboundary. Therefore, λ, by construction, is a three-cocycle defined up to a three-coboundary; thus, by the definition of the cohomology group, it is an element from $H^3(G,\mathbb{C})$.

Next, consider MPO representations that are translationally invariant after blocking two tensors X and Y. The previous method assigns two possibly different labels from $H^3(G,\mathbb{C}^*)$ to the two MPO tensors XY and YX. We will show now that these two labels are in fact equal.

Proposition 23. Let $V_n(X_gY_g)$ be a one-block projective MPO representation of G. Then $V_n(Y_gX_g)$ is also a one-block projective MPO representation of G and their third cohomology label is the same.

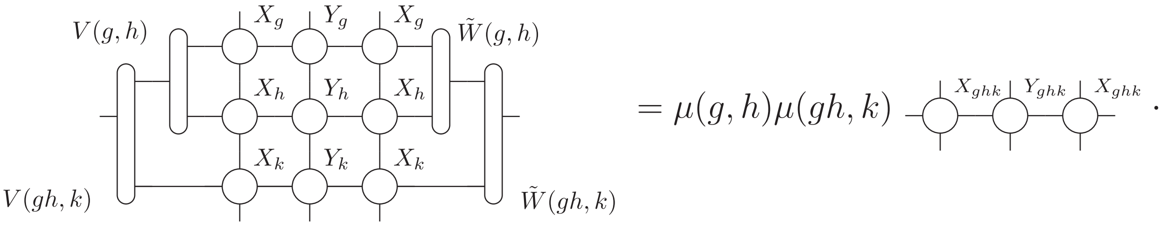
Proof. As $V_n(Y_gX_g)$ is the same MPO as $V_n(X_gY_g)$, but shifted by one lattice site, it is a one-block projective MPO representation. Without loss of generality, one can suppose that both X_gY_g and Y_gX_g are injective: they contain only one block; thus, they can be reduced to injective MPOs. Thus, incorporating the reductions into X_g and Y_g, we obtain two new tensors such that both X_gY_g and Y_gX_g are injective.

Let $V(g, h)$ and $W(g, h)$ be reductions corresponding to the product of X_gY_g and X_hY_h, while $\tilde{V}(g, h)$ and $\tilde{W}(g,h)$ be reductions for the product of Y_gX_g and X_hY_h. Then Proposition 29 in Appendix B implies that $V(g, h)$ and $\tilde{W}(g,h)$ reduces (up to a scalar) a chain of odd number of MPO tensors,



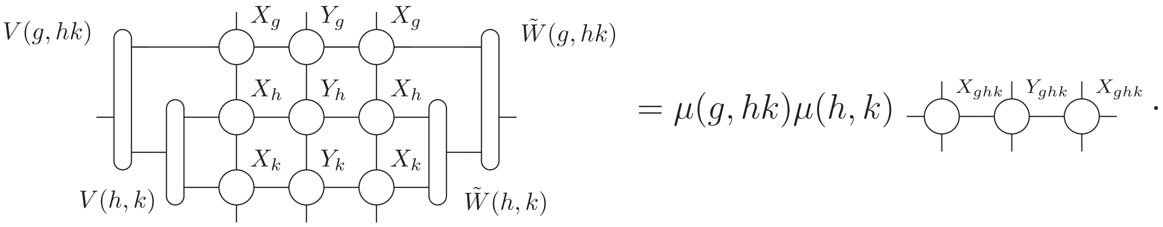
(148)

Therefore, for the product of three MPOs corresponding to g, h, and k and a chain consisting of an odd number of MPO tensors,



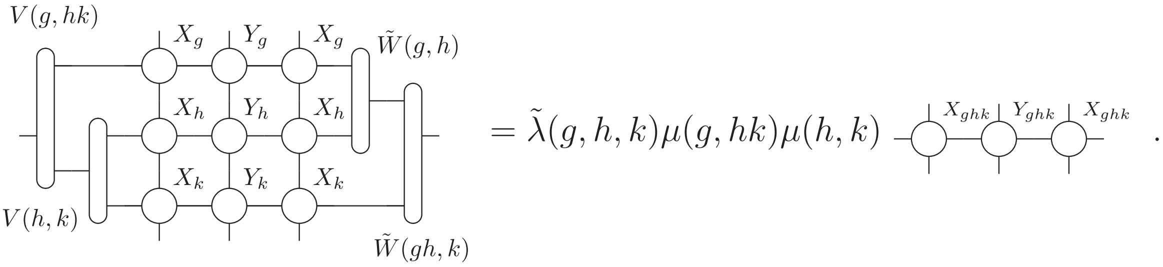
(149)

Similarly,



(150)

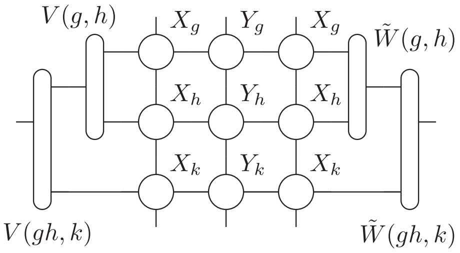
If the above chain is long enough, changing the order of the reductions $\tilde{W}$ changes the above equation only by a scalar $\tilde{\lambda}(g,h,k)$,



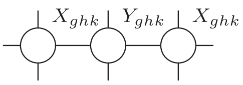
(151)

Similarly, changing the order of the reductions on the left side, we get (notice that the scalar associated with changing the order of the reductions on the left side is the inverse of that on the right side, see

Theorem 22)



$$= \frac{\tilde{\lambda}(g,h,k)\mu(g,hk)\mu(h,k)}{\lambda(g,h,k)}$$



. (152)

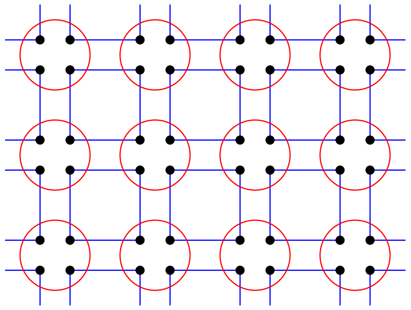
Comparing this equation with Eq. (149), we conclude that

$$\frac{\lambda(g,h,k)}{\tilde{\lambda}(g,h,k)} = \frac{\mu(g,hk)\mu(h,k)}{\mu(g,h)\mu(gh,k)}. \tag{153}$$

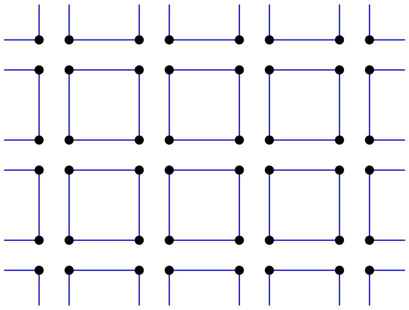
Therefore, the two scalars differ only by a three coboundary. That is, the two third cohomology labels corresponding to X_gY_g and Y_gX_g coincide. □

B. Third cohomology labeling of semi-injective PEPS

We investigate the following setup. Let G be a group and O_g be a faithful (not necessarily unitary) representation of G. Let $|\phi\rangle$ be a four-partite state with full rank one-particle reduced densities. Suppose $\forall g \in G$, O_g is a symmetry of the semi-injective PEPS defined by $|\phi\rangle$ and Id,



$$= \mu_{n,m}(g)$$



, (154)

where the blue squares represent $|\phi\rangle$ and the red circles represent operators O_g.

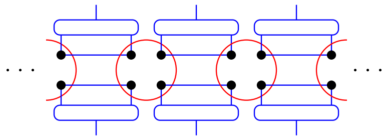
Note that this setup can readily be applied for unitary symmetries of semi-injective PEPSs: let the semi-injective PEPS be defined by the four-partite state $|\phi\rangle$ and an invertible operator A. Let the unitary representation of the symmetry group G be U_g. Then, by inverting A in the symmetry condition, we arrive to Eq. (154) with $O_g = A^{-1}U_gA$.

Proposition 24. If Eq. (154) holds for some n, m ≥ 3, *then it holds for all n, m, and* $\mu_{n,m}(g) = \mu^{nm}(g)$ *for some one-dimensional representation* μ *of G.*

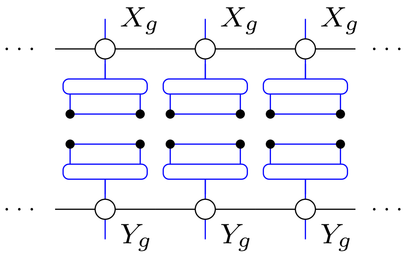
Proof. Apply Proposition 12 and notice that μ is a representation. □

We show now that the action of the symmetries shows up on the boundary as a projective MPO representation of the group G.

Proposition 25. If Eq. (154) holds, then for every $g \in G$ *there are two MPO tensors* X_g *and* Y_g *such that*



$$= \mu^n(g)$$



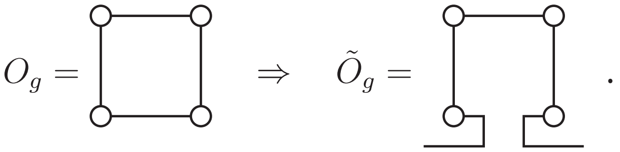
, (155)

and $V_n(Y_g) = \left(V_n(X_g)\right)^{-1}$ *for all n. Moreover,* $V_n(X_g)$ *and* $V_n(Y_g)$ *form projective representations of G with* $V_n(X_g)V_n(X_h) = \lambda^n(g, h)V_n(X_gX_h)$ *for a two-cocycle* λ*. In particular,* $V_n(X_g)V_n(X_h)$ *has only one block in its canonical form.*

Proof. From Theorem 13, the existence of X_g and Y_g is clear. From Corollary 14.1, it is also true that $V_n(X_g)V_n(X_h) = \lambda^n(g, h)V_n(X_gX_h)$. Due to associativity, $\lambda(g, h)\lambda(gh, k) = \lambda(g, hk)\lambda(h, k)$, and thus λ forms a two-cocycle. □

Note that if we allow for blocking, there is a length scale K for which $\lambda^{Kn}(g, h)$ becomes constant 1. On the other hand, the labeling with an element from the third cohomology group $H^3(G, \mathbb{C})$ corresponding to the $g \mapsto X_g$ one-block projective MPO representation of G is a scale-invariant labeling.

In the following, we show that the classification of the boundary MPO representation $V_n(X_g)$ also shows up in the MPO defined by O_g. To see this, we define a translationally invariant (on four sites) MPO from O_g that we call $V_n(\tilde{O}_g)$. Write O_g as an MPO in Eq. (104), and open one of the indices. We call this tensor $\tilde{O}_g$. Pictorially,

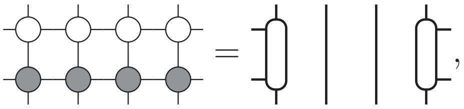


(156)

This MPO plays an important role in the third cohomology labeling of semi-injective PEPSs.

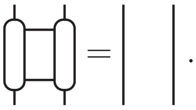
Proposition 26. The MPOs $V_n(\tilde{O}_g)$ *form a one-block projective MPO representation of G. Its third cohomology label is the same as that of* $V_n(X_g)$.

Proof. As the product of the Schmidt vectors of O_g and O_g^{-1} factorizes, the tensor $\sum_j \tilde{O}_g^{ij}\tilde{O}_{g^{-1}}^{jk}$ has the following structure:



(157)

with

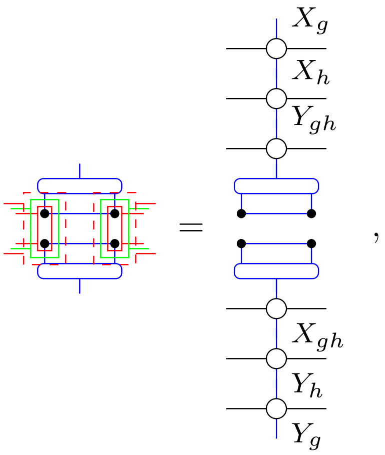


(158)

Therefore, $V_n(\tilde{O}_g)V_n(\tilde{O}_{g^{-1}}) = \text{Id}$, as it is the n-fold product of this tensor.

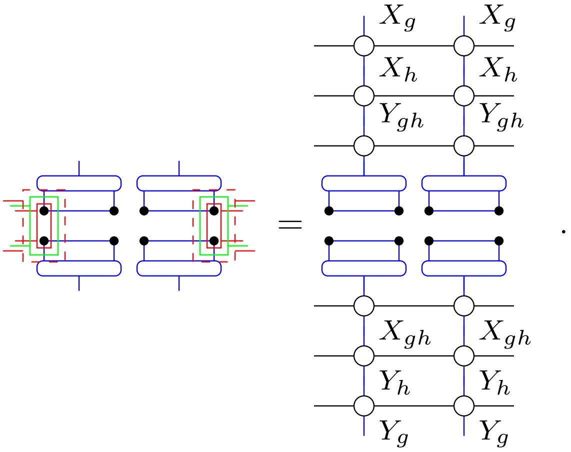
We prove now that $V_n(\tilde{O}_g)V_n(\tilde{O}_h)V_n(\tilde{O}_{(gh)^{-1}}) = \text{Id}$, and thus $V_n(\tilde{O}_g)V_n(\tilde{O}_h) = V_n(\tilde{O}_{gh})$.

Consider the MPS tensor defined by the Schmidt vectors of O_g, O_h, and then $O_{(gh)^{-1}}$ acting on the Schmidt vectors of $|\phi\rangle$. Then, similar to Eq. (98), this tensor can be written as



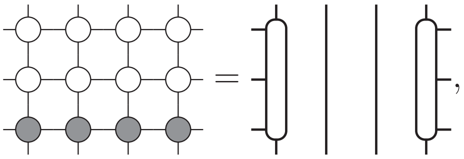
(159)

where the red solid rectangle denotes the Schmidt vectors of O_g, the green one denotes that of O_h, and the dashed one denotes that of $O_{(gh)^{-1}}$. Joining two such tensors, the middle operator is $O_g O_h O_{(gh)^{-1}} = \mathrm{Id}$, so



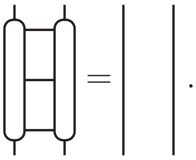
(160)

As the l.h.s. factorizes with respect to the vertical cut, and the r.h.s. factorizes with respect to the horizontal cut, and the one particle reduced densities of $|\phi\rangle$ are full rank, the product of the Schmidt vectors of O_g, O_h, and $O_{(gh)^{-1}}$ also factorizes, and thus



(161)

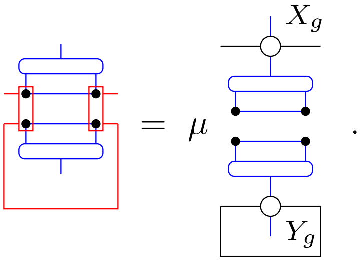
with



(162)

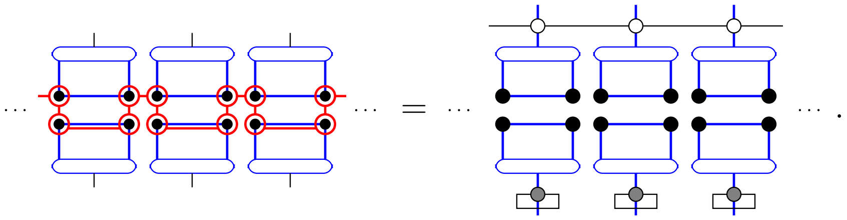
Therefore $V_n(\tilde{O}_g)V_n(\tilde{O}_h)V_n(\tilde{O}_{(gh)^{-1}}) = \mathrm{Id}$, as it is the n-fold product of this tensor. This means that $V_n(\tilde{O}_g)$ is an MPO representation.

As an MPO representation is also a one-block projective MPO representation, one can label this MPO representation with an element from the third cohomology group $H^3(G, \mathbb{C}^*)$. We now show that this label coincides with that of the projective MPO representation of G on the boundary. To see this, partially contract the MPS tensors describing the boundary of the state [defined in Eq. (95)]. That is, contract only the lower indices,



(163)

Notice that the red MPO tensor acting on the l.h.s. is exactly $\tilde{O}_g$. After contracting these tensors, Eq. (163) reads


(164)

By construction, the red MPO appearing on the l.h.s. is $V_n(\tilde{O}_g)$. Therefore, if $V(g, h)$, $W(g, h)$ is a reduction from $V_n(\tilde{O}_g)V_n(\tilde{O}_h)$ to $V_n(\tilde{O}_{gh})$, then it is also a reduction from $V_n(X_g)V_n(X_h)$ to $V_n(X_{gh})$. As the third cohomology is assigned to the MPO representation with the help of these reductions, $V_n(O_g)$ is classified by the same third cohomology class as $V_n(X_g)$. □

The above proof can be repeated for the vertical boundary instead of the horizontal one. This means that the third cohomology label of the vertical boundary is the same as that of $V_n(\tilde{O}'_g)$, where $\tilde{O}'_g = D_g A_g B_g C_g$ if $O_g = A_g B_g C_g D_g$. Proposition 23 implies that the third cohomology labeling of $V_n(\tilde{O}'_g)$ and $V_n(\tilde{O}_g)$ coincides; therefore, the third cohomology labeling of the horizontal and vertical boundary coincides.

VIII. CONCLUSION

In this work, we introduced a new class of PEPSs, semi-injective PEPSs. We showed that semi-injective PEPSs are a generalization of injective PEPSs and that some important examples that are not known to have an injective PEPS description naturally admit a semi-injective PEPS description. We showed that they are unique ground states of their parent Hamiltonian. We also derived a canonical form, i.e., a way to decide locally if two semi-injective PEPSs are equal. One of the necessary conditions is that the boundaries of the two states are related by an invertible MPO. Using this result, the third cohomology labeling of SPT phases extends naturally to semi-injective PEPSs, suggesting that these states are appropriate to capture the relevant physics of SPT phases. We also showed that this third cohomology labeling is directly encoded in the physical symmetry operators. This property enables one to find the third cohomology label locally without considering the boundary of a large system.

ACKNOWLEDGMENTS

The authors would like to thank Barbara Kraus, Frank Verstraete, David Pérez-García, and Vedran Dunjko for helpful discussions. This project has been supported by the European Union through the ERC Starting Grant WASCOSYS (No. 636201).

APPENDIX A: EXAMPLES FOR CANONICAL FORM

In the injective PEPS case, if two tensors generate the same state, then they are related by a product gauge transformation. In the case of semi-injective PEPSs, this is no longer true as the following example shows.

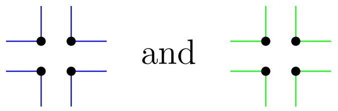
Let A be the following MPS tensor:

$$A^0 = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}, \tag{A1}$$

$$A^1 = \begin{pmatrix} 24 & -10 \\ 17 & -3 \end{pmatrix}. \tag{A2}$$

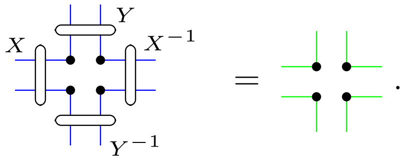
This tensor was constructed in such a way that it is Z symmetric for size 4, but not for longer chains: for the tensor B with $B^0 = A^0$ and $B^1 = -A^1$, $V_4(B) = V_4(A)$, but $V_5(A) \neq V_5(B)$. The tensors A and B are also normal, and after blocking two tensors they become injective. Proposition 7 also implies that A and B are not related by a gauge transform. There is also no gauge relating the tensors after blocking four of them: $\nexists X$: $XBBBBX^{-1} = AAAA$.

Consider two semi-injective PEPSs. Let Ψ_A be defined by $\phi_A = V_4(A)$ and Id, Ψ_B by $\phi_B = V_4(B)$ and Id. By construction, $\Psi_A = \Psi_B$. We will show, however, that the PEPS tensors defined by grouping four MPS tensors,



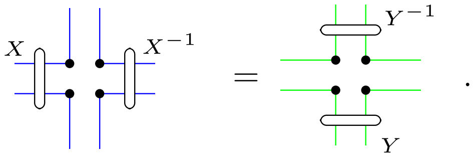
(A3)

are not related by a gauge, where the blue tensors are A and the green ones are B. We prove that by contradiction. Suppose there are such gauges, X and Y,



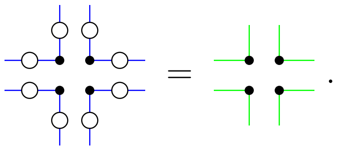
(A4)

Inverting Y and Y^{-1}, we get that



(A5)

Notice that the l.h.s. is the product with respect to the vertical cut, whereas the r.h.s. is the product with respect to the horizontal cut. As A and B become injective after blocking two tensors, both X and Y have to be product operators, and thus



(A6)

But this would mean that after blocking four tensors, $BBBB = XAAAAX^{-1}$ for some gauge X. As this is a contradiction, the two given PEPS tensors generating the same semi-injective PEPS are not related by a gauge.

APPENDIX B: MPS REDUCTIONS

In this section, we present the proofs of the theorems about reductions of MPS used in Sec. VII.

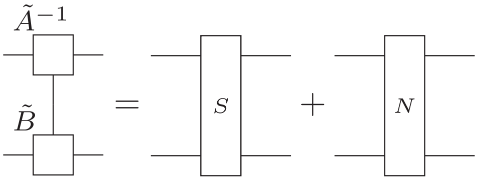
Proposition 20. Let A be a normal MPS tensor and B be an MPS tensor such that for some $\lambda \in \mathbb{C}$,

$$V_n(B) = \lambda^n V_n(A), \quad \forall n \in \mathbb{N}. \tag{135'}$$

Then there exist matrices V, W such that $VW = \mathrm{Id}$ *and* $\forall n \in \mathbb{N}$ *and* $(i_1, i_2, \ldots, i_n) \in \{1, 2, \ldots, d\}^n$,

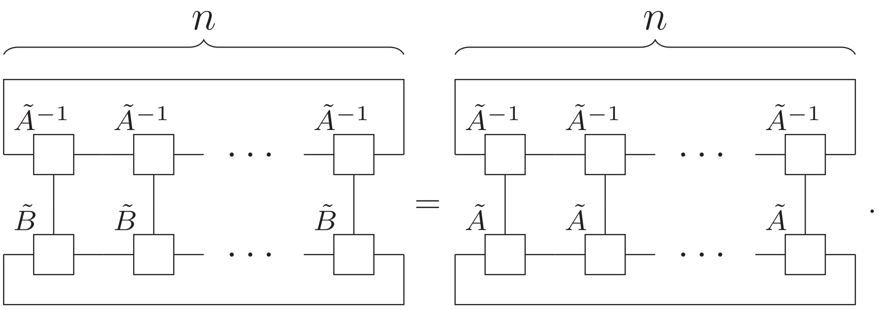
$$VB^{i_1} \ldots B^{i_n} W = A^{i_1} \ldots A^{i_n}. \tag{136'}$$

Proof. Suppose the injectivity length (see Proposition 5) of A is L. Let $\tilde{A}$ and $\tilde{B}$ denote the tensors obtained from A and B by blocking them L times, respectively. Then $\tilde{A}$ has a left inverse, $\tilde{A}^{-1}$. Take the Jordan decomposition of the following matrix:



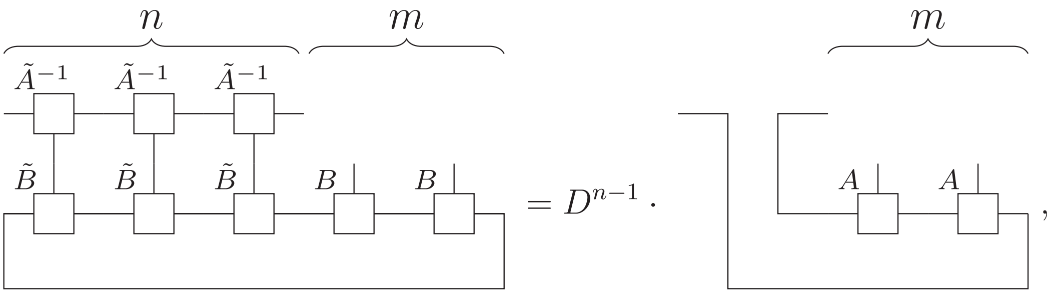
(B1)

where S is semi-simple (diagonalizable), N is nilpotent (upper triangular in the basis in which S is diagonal), and $[S, N] = 0$. B and A generate the same state; thus,



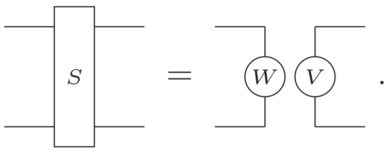
(B2)

The r.h.s is D^n, where D is the bond dimension of A, as it is n times the trace of Id. Using the Jordan decomposition, Eq. (B1), the l.h.s. is $\mathrm{Tr}(S + N)^n = \mathrm{Tr}\, S^n$. Therefore $\mathrm{Tr}\, S^n = D^n$; thus, Proposition 10 implies that the rank of S is 1. $[S, N] = 0$ therefore implies that $SN = NS = 0$. Thus, $(S + N)^n = S^n + N^n = S^n$ if n is larger than the nilpotency rank of N. Then, as A and B generate the same state, for all n and m,



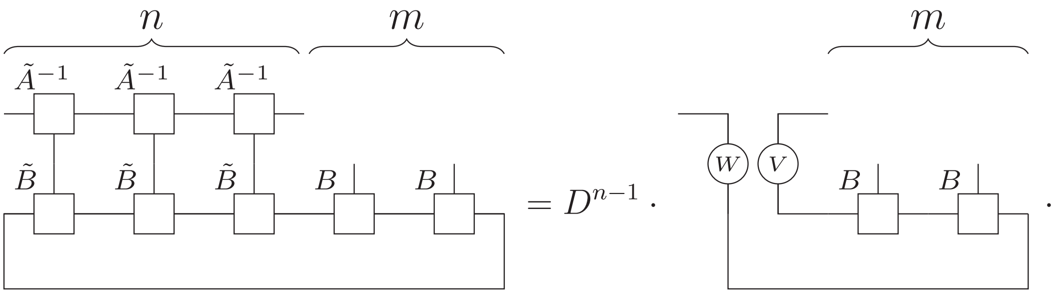
(B3)

where we have used $n - 1$ times that Tr Id = D. As S is rank one, there are matrices V and W such that S can be written as



(B4)

Therefore, as $N^n = 0$, the l.h.s. can be rewritten as



(B5)

Therefore, comparing this with the r.h.s. of Eq. (B3), for all m,

$$VB^{i_1}\dots B^{i_m}W = A^{i_1}\dots A^{i_m}. \tag{B6}$$

For $m = 0$, $VW = \mathrm{Id}$. □

Proposition 21. Let V, W be a reduction from B to A. Let $N^i = B^i - WA^iV$. Then the algebra generated by N^i is nilpotent.

Before proceeding to the proof, we need the following simple statement:

Lemma 27. Let V, W be a reduction from B to an injectie MPS tensor A, $N^i = B^i - WA^iV$. Then for any $m > 0$,

$$VN^{i_1}N^{i_2}\dots N^{i_m}W = 0. \tag{B7}$$

Proof. We prove this by induction on m. For $m = 1$,

$$VN^iW = VB^iW - VWA^iVW = A^i - A^i = 0. \tag{B8}$$

Suppose the statement is true for all $n < m$. Then, writing $N^{i_1} = B^{i_1} - WA^{i_1}V$ and using the induction hypothesis,

$$VN^{i_1}N^{i_2}\dots N^{i_m}W = VB^{i_1}N^{i_2}\dots N^{i_m}W. \tag{B9}$$

Similarly, $N^{i_2},\dots,N^{i_{m-1}}$ can be changed to $B^{i_2},\dots,B^{i_{m-1}}$,

$$VN^{i_1}N^{i_2}\dots N^{i_m}W = VB^{i_1}\dots B^{i_{m-1}}N^{i_m}W. \tag{B10}$$

Writing now $N^{i_m} = B^{i_m} - WA^{i_m}V$, we arrive to

$$VN^{i_1}\dots N^{i_m}W = VB^{i_1}\dots B^{i_m}W - VB^{i_1}\dots B^{i_{m-1}}WA^{i_m}VW = 0. \tag{B11}$$

□

Proof of Proposition 21. B and A generate the same state,

$$\mathrm{Tr}\left\{B^{i_1}B^{i_2}\dots B^{i_n}\right\} = \mathrm{Tr}\left\{A^{i_1}A^{i_2}\dots A^{i_n}\right\}. \tag{B12}$$

Write $B^i = WA^iV + N^i$ and expand the product on the l.h.s. As V and W form a reduction, $VN^{i_1}\dots N^{i_m}W = 0$ for any $m > 0$ and all $i_1, \dots, i_m$ by Lemma 27, and thus all terms cancel except the products of A and the products of N. Therefore,

$$\mathrm{Tr}\left\{N^{i_1}N^{i_2}\dots N^{i_n}\right\} = 0. \tag{B13}$$

This means that $\mathrm{Tr}\{Z\} = 0$ for every element Z in the algebra generated by N^i. Thus, in particular, for every $n > 0$, $\mathrm{Tr}\{Z^n\} = 0$. Therefore the algebra generated by N^i is a nil algebra and thus nilpotent.[50] That is,

$$N^{i_1}N^{i_2}\dots N^{i_n} = 0 \tag{B14}$$

for large enough n. □

Theorem 22. *Let V, W and $\tilde V, \tilde W$ be two reductions from B to a normal tensor A. Let the nilpotency length of both reductions be at most N_0. Then $\exists\lambda \in \mathbb{C}$ such that for any $n > 2N_0$,*

$$VB^{i_1}B^{i_2}\dots B^{i_n} = \lambda\tilde VB^{i_1}B^{i_2}\dots B^{i_n}, \tag{138′}$$

$$B^{i_1}B^{i_2}\dots B^{i_n}W = \lambda^{-1}B^{i_1}B^{i_2}\dots B^{i_n}\tilde W. \tag{139′}$$

Before proceeding to the proof, we need the following calculation that we use repeatedly:

Lemma 28. Let V, W be a reduction from B to a normal tensor A, $N^i = B^i - WA^iV$. Let N_0 be the nilpotency length of the reduction. Then the following equations hold:

$$B^{i_1}B^{i_2}\dots B^{i_n} = \sum_{0\le k\le l\le n} N^{i_1}\dots N^{i_k}WA^{i_{k+1}}\dots A^{i_l}VN^{i_{l+1}}\dots N^{i_n}, \tag{B15}$$

$$VB^{i_1}B^{i_2}\dots B^{i_n} = \sum_{\max(0,n-N_0)\le l\le n} A^{i_1}\dots A^{i_l}VN^{i_{l+1}}\dots N^{i_n}, \tag{B16}$$

$$B^{i_1}B^{i_2}\dots B^{i_n}W = \sum_{0\le k\le \min(N_0,n)} N^{i_1}\dots N^{i_k}WA^{i_{k+1}}\dots A^{i_n}. \tag{B17}$$

Proof. Write $B^{i_j} = WA^{i_j}V + N^{i_j}$ for all j in $B^{i_1}\dots B^{i_n}$ and expand the expression. Using Lemma 27 and the definition of the nilpotency length (Definition 8), we arrive at the desired equations. □

Proof of Theorem 22. Let L be the injectivity length of A and let $m = 2N_0 + L$. Consider

$$C^{i_1\dots i_m} = VB^{i_1}B^{i_2}\dots B^{i_m}\tilde{W}. \tag{B18}$$

Using Lemma 28 with $B^i = WA^iV + N^i$, we have

$$C^{i_1\dots i_m} = \sum_{k=N_0+L}^{m} A^{i_1}\dots A^{i_k}VN^{i_{k+1}}\dots N^{i_m}\tilde{W}, \tag{B19}$$

as $m - N_0 = N_0 + L$. Similarly, using Lemma 28 with $B^i = \tilde{W}A^i\tilde{V} + \tilde{N}^i$, we get

$$C^{i_1\dots i_m} = \sum_{k=0}^{N_0} V\tilde{N}^{i_1}\dots \tilde{N}^{i_k}\tilde{W}A^{i_{k+1}}\dots A^{i_m}. \tag{B20}$$

Note that the MPS tensor at position $k = N_0 + 1$ to $N_0 + L$ is A^{i_k} in both expressions. By assumption, that block is injective. Applying its inverse and comparing the two expressions, we conclude that

$$\sum_{k=0}^{N_0} V\tilde{N}^{i_1}\dots \tilde{N}^{i_k}\tilde{W}A^{i_{k+1}}\dots A^{i_{N_0}} = \lambda A^{i_1}\dots A^{i_{N_0}}, \tag{B21}$$

$$\sum_{k=0}^{N_0} A^{i_1}\dots A^{i_k}VN^{i_{k+1}}\dots N^{i_{N_0}}\tilde{W} = \lambda^{-1}A^{i_1}\dots A^{i_{N_0}} \tag{B22}$$

for some $\lambda \in \mathbb{C}$. But then, using Lemma 28 for $VB^{i_1}B^{i_2}\dots B^{i_n}$ with $B^i = \tilde{W}A^i\tilde{V} + \tilde{N}^i$, we get

$$VB^{i_1}B^{i_2}\dots B^{i_n} = \sum_{k=0}^{N_0}\sum_{l=n-N_0}^{n} V\tilde{N}^{i_1}\dots \tilde{N}^{i_k}\tilde{W}A^{i_{k+1}}\dots A^{i_l}\tilde{V}N^{i_{l+1}}\dots N^{i_n}. \tag{B23}$$

If $n \ge 2N_0$, then $l \ge N_0$. Therefore the left part of the r.h.s. can be replaced using Eq. (B21),

$$VB^{i_1}B^{i_2}\dots B^{i_n} = \lambda \sum_{l=n-N_0}^{n} A^{i_1}\dots A^{i_l}\tilde{V}N^{i_{l+1}}\dots N^{i_n} = \lambda\tilde{V}B^{i_1}B^{i_2}\dots B^{i_n}, \tag{B24}$$

where the last equation holds by using Lemma 28 for $\tilde{V}B^{i_1}B^{i_2}\dots B^{i_n}$ with $B^i = \tilde{W}A^i\tilde{V} + \tilde{N}^i$. Equation (139′) can be proven similarly using Eq. (B22). □

We now consider MPSs that are translationally invariant after blocking two sites.

Proposition 29. Let $A \in \mathbb{C}^{D_1}\otimes\mathbb{C}^{D_2}\otimes\mathbb{C}^{d_1}$ and $B \in \mathbb{C}^{D_2}\otimes\mathbb{C}^{D_1}\otimes\mathbb{C}^{d_2}$ be two tensors such that both AB and BA are normal MPS tensors. Let $C \in \mathbb{C}^{\tilde{D}_1}\otimes\mathbb{C}^{\tilde{D}_2}\otimes\mathbb{C}^{d_1}$ and $D \in \mathbb{C}^{\tilde{D}_2}\otimes\mathbb{C}^{\tilde{D}_1}\otimes\mathbb{C}^{d_1}$ be two tensors such that $V_n(CD) = V_n(AB)$. Then $V_n(BA) = V_n(DC)$ and if V, W are reductions of CD to AB and $\tilde{V}, \tilde{W}$ are reductions of DC to BA, then for a sufficiently long chain,

$$AB\dots BA = \lambda VCD\dots DC\tilde{W}, \tag{B25}$$

$$BA\dots AB = \mu\tilde{V}DC\dots CDW. \tag{B26}$$

Proof. First, notice that $V_n(BA) = V_n(DC)$, as $V_n(BA)$ is $V_n(AB)$ shifted by half a lattice constant, while $V_n(DC)$ is $V_n(CD)$ shifted by half a lattice constant.

Next, using Lemma 28 with $CD = WABV + N_1N_2$, we have

$$V\,\underbrace{CD\dots D}_{2k}\,C\tilde{W} = \sum_{i=0}^{M} \underbrace{AB\dots B}_{2k-2i}\,V\,\underbrace{N_1N_2\dots N_1N_2}_{2i}\,C\tilde{W}, \tag{B27}$$

where M is the injectivity length of the reduction V, W. Similarly, using Lemma 28 with $DC = \tilde{W}BA\tilde{V} + \tilde{N}_1\tilde{N}_2$, we have

$$VC\,\underbrace{D\dots DC}_{2k}\,\tilde{W} = \sum_{i=0}^{\tilde{M}} VC\,\underbrace{\tilde{N}_1\tilde{N}_2\dots\tilde{N}_1\tilde{N}_2}_{2i}\,\tilde{W}\,\underbrace{BA\dots BA}_{2k-2i}, \tag{B28}$$

where $\tilde{M}$ is the injectivity length of $\tilde{V}$, $\tilde{W}$. Therefore, if $2k > 2\,\mathrm{M} + 2\tilde{M} + 2L$, where L is the injectivity length of AB, then

$$V\,\underbrace{CD\dots C}_{2k+1}\,\tilde{W} = \underbrace{AB\dots A}_{2N+1}\,\underbrace{B\dots A}_{2k-1-2N}\,\underbrace{\boxed{}}_{2N+1} = \underbrace{\boxed{}}_{2N+1}\,\underbrace{B\dots A}_{2k-1-2N}\,\underbrace{B\dots A}_{2N+1}\,\cdot \tag{B29}$$

As the middle part is injective, the last equation can hold only if $AB\dots BA = \lambda VCD\dots DC\tilde{W}$. The other equation can be proven similarly. □

APPENDIX C: THE QUBIT CASE

In this section, we characterize how two semi-injective PEPSs defined by $(|\phi_A\rangle, O)$ and $(|\phi_B\rangle, \mathrm{Id})$ with $|\phi_A\rangle, |\phi_B\rangle \in \left(\mathbb{C}^2\right)^{\otimes 4}$ can generate the same state. We restrict ourselves to the case where $|\phi_A\rangle$ and $|\phi_B\rangle$ do not factorize in either direction. Using Corollary 18.1, $|\phi_A\rangle$ and $|\phi_B\rangle$ are SLOCC equivalent, and thus we can suppose $|\phi_A\rangle = |\phi_B\rangle$ (by changing O). Notice that the state $|\xi\rangle$ appearing in the proof of Theorem 18 [see Eq. (125)] is also SLOCC equivalent with ϕ_A. We can thus suppose that $|\phi_A\rangle = |\phi_B\rangle = |\xi\rangle$. Therefore, given $|\phi_A\rangle$, we only need to characterize all pairs of two-body invertible operators such that Eq. (125) holds.

Let us fix $|\phi_A\rangle = |\phi_B\rangle = |\xi\rangle$. We start the investigation with a state such that in the horizontal cut it has Schmidt rank two. As the span of the Schmidt vectors contains a product state and we are only interested in $|\phi_A\rangle$ up to SLOCC equivalence, without loss of generality we can suppose that a basis of its reduced density on the upper two particles is

$$|\Psi_1\rangle = |00\rangle\,, \tag{C1}$$

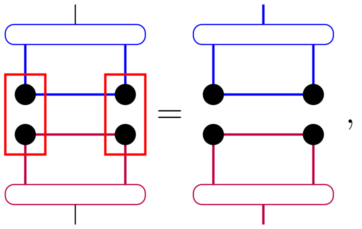
$$|\Psi_2\rangle = a\,|01\rangle + b\,|10\rangle + c\,|11\rangle\,, \tag{C2}$$

whereas a basis of its reduced density on the lower two particles is

$$|\Phi_1\rangle = |00\rangle\,, \tag{C3}$$

$$|\Phi_2\rangle = A\,|01\rangle + B\,|10\rangle + C\,|11\rangle\,. \tag{C4}$$

In this setting, we are looking for invertible two body operators O_1 and O_2 such that



, (C5)

where the left red rectangle represents O_1, the right one represents O_2, while the blue Schmidt vectors are $\Psi_{1/2}$, the purple ones are $\Phi_{1/2}$. This gives four times sixteen equations on the matrix elements of O_1 and O_2. Checking these equations can be done in any computer algebra system (CAS). The following cases can be distinguished:

- $C \neq 0$, $c \neq 0$: In this case, the operators are (α, β, γ are free parameters)

$$O_1 = |00\rangle\langle 00| + \alpha\,|01\rangle\langle 01| + \frac{b}{c}(\alpha - 1)\,|00\rangle\langle 01| + \beta\,|10\rangle\langle 10| + \frac{B}{C}(\beta - 1)\,|00\rangle\langle 10| + \gamma\,|11\rangle\langle 11| + \frac{b}{c}(\gamma - \beta)\,|10\rangle\langle 11| + \frac{B}{C}(\gamma - \alpha)\,|01\rangle\langle 11| + \frac{Bb}{Cc}(1 + \gamma - \alpha - \beta)\,|00\rangle\langle 11|, \tag{C6}$$

$$O_2 = |00\rangle\langle 00| + \frac{1}{\alpha}\,|01\rangle\langle 01| + \frac{a}{c}(\frac{1}{\alpha} - 1)\,|00\rangle\langle 01| + \frac{1}{\beta}\,|10\rangle\langle 10| + \frac{A}{C}(\frac{1}{\beta} - 1)\,|00\rangle\langle 10| + \frac{1}{\gamma}\,|11\rangle\langle 11| + \frac{a}{c}(\frac{1}{\gamma} - \frac{1}{\beta})\,|10\rangle\langle 11| + \frac{A}{C}(\frac{1}{\gamma} - \frac{1}{\alpha})\,|01\rangle\langle 11| + \frac{aA}{Cc}(1 + \frac{1}{\gamma} - \frac{1}{\alpha} - \frac{1}{\beta})\,|00\rangle\langle 11|. \tag{C7}$$

- $C \neq 0$, $c = 0$: In this case $b \neq 0$, otherwise the one particle reduced densities of the state are not full rank. The operators are (α, β, γ are free parameters)

$$O_1 = |00\rangle\langle 00| + |01\rangle\langle 01| + \alpha\,|00\rangle\langle 01| + \beta\,|10\rangle\langle 10| + \frac{B}{C}(\beta - 1)\,|00\rangle\langle 10| + \beta\,|11\rangle\langle 11| + \gamma\,|10\rangle\langle 11| + \frac{B(\beta - 1)}{C}\,|01\rangle\langle 11| + \frac{B\gamma - B\alpha}{C}\,|00\rangle\langle 11|, \tag{C8}$$

$$O_2 = |00\rangle\langle 00| + |01\rangle\langle 01| - \frac{a}{b}\alpha\,|00\rangle\langle 01| + \frac{1}{\beta}\,|10\rangle\langle 10| + \frac{A}{C}(\frac{1}{\beta} - 1)\,|00\rangle\langle 10| + \frac{1}{\beta}\,|11\rangle\langle 11| - \frac{a\gamma}{b\beta^2}\,|10\rangle\langle 11| + (-\frac{A}{C} + \frac{A}{C\beta})\,|01\rangle\langle 11| + \frac{Aa\gamma - Aa\gamma/\beta^2}{Cb}\,|00\rangle\langle 11|. \tag{C9}$$

- $c = 0$, $C = 0$: In this case $B \neq 0$, $b \neq 0$, otherwise the one particle reduced densities of the state are not full rank. The operators are (α, β, γ are free parameters)

$$O_1 = |00\rangle\langle 00| + |01\rangle\langle 01| + \alpha\,|00\rangle\langle 01| + |10\rangle\langle 10| + \beta\,|00\rangle\langle 10| + |11\rangle\langle 11| + \alpha\,|10\rangle\langle 11| + \beta\,|01\rangle\langle 11| + \gamma\,|00\rangle\langle 11|, \tag{C10}$$

$$O_2 = |00\rangle\langle 00| + |01\rangle\langle 01| - \frac{a}{b}\alpha\,|00\rangle\langle 01| + |10\rangle\langle 10| - \frac{B}{A}\beta\,|00\rangle\langle 10| + |11\rangle\langle 11| - \frac{a}{b}\alpha\,|10\rangle\langle 11| - \frac{A}{B}\beta\,|01\rangle\langle 11| + \frac{2Aa\alpha\beta - Aa\gamma}{Bb}\,|00\rangle\langle 11|. \tag{C11}$$

Using this result, we have checked that if the state has a Schmidt rank of at least 3, then O_1 and O_2 can only be product operators.

To find therefore all possible operators O such that the semi-injective PEPS defined by ($|\phi_A\rangle$, O) and by ($|\phi_B\rangle$, Id) are the same (supposing they have a Schmidt rank of at least two along both vertical and horizontal cuts), one has to do the following steps:

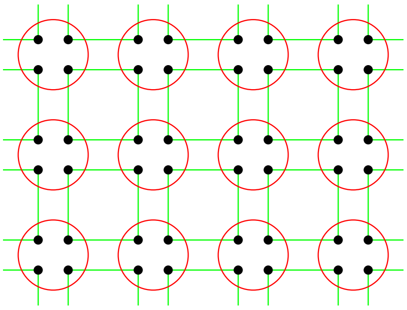
1. Transform $|\phi_A\rangle$ with an invertible product operator O_1 to have $|00\rangle$ in the span of its Schmidt vectors in both the upper and lower two particles.
2. If the Schmidt rank of $|\phi_A\rangle$ is two along the horizontal cut, then take the two-body operators given above, $O_2 \otimes O_3$. Otherwise take $O_2 = O_3 = \text{Id}$.
3. Repeat the previous two steps for the vertical cut, giving an invertible product operator O_4 and two-body invertible operators $O_5 \otimes O_6$.
4. Find all invertible product transformation O_7 such that $|\phi_B\rangle = O_7|\phi_A\rangle$.

Then all possible operators O are given by $\tilde{O}_1(O_2 \otimes O_3)\tilde{O}_1^{-1}\tilde{O}_4(O_5 \otimes O_6)\tilde{O}_4^{-1}\tilde{O}_7$, where $\tilde{O}_i = SO_iS$, and S is the swap operator defined before Theorem 17.

APPENDIX D: G-INJECTIVE TENSORS

In this section, we try to generalize semi-injective PEPSs in a way that it also includes G-injective PEPSs. We try the obvious generalization: if the semi-injective PEPS ($|\phi\rangle$, Id) has symmetries O_g for $g \in G$ for some group G, then $|\phi\rangle$ and $\sum_g O_g$ defines a non-semi-injective PEPS that could be a candidate to include G-injective PEPSs. We present here, however, an example for such a state that behaves very different from G-injective PEPSs.

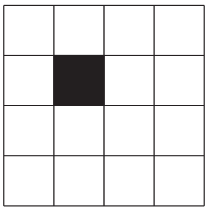
Consider the following state:

$|\Psi\rangle =$  , (D1)

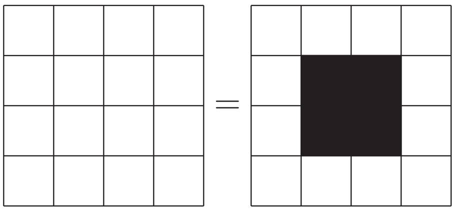
where the green rectangle is a four-partite GHZ state, and the red circle is $O = \mathrm{Id} + Z^{\otimes 4}$.

We will show that on an $n \times n$ torus, there are at least 2^n linearly independent states that are locally indistinguishable from this state. This means that given any local (frustration free) parent Hamiltonian, its ground space is at least 2^n-fold degenerate.

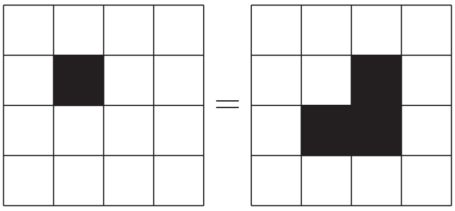
To see this, consider states on the torus that are constructed similar to $|\Psi\rangle$, except that some of the four-partite GHZ states $|\phi^+\rangle = 1/\sqrt{2}(|0000\rangle + |1111\rangle)$ are changed to $|\phi^-\rangle = 1/\sqrt{2}(|0000\rangle - |1111\rangle)$. Such a state will be depicted schematically as a rectangular grid, with squares colored black at all occurrence of $|\phi^-\rangle$. For example, the following figure depicts such a state with one occurrence of $|\phi^-\rangle$,

 . (D2)

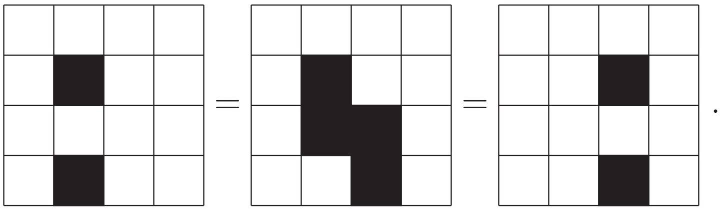
We will see that these states are all locally indistinguishable from $|\Psi\rangle$ and that they span an at least $\exp\{n/2\}$-dimensional space. First notice that $|\phi^-\rangle = Z|\phi^+\rangle$, where Z acts on one of the four particles (any one of them). Due to the special form of O, however, if in a 2×2 block all $|\phi^+\rangle$ are changed to $|\phi^-\rangle$, it does not change the state,

$|\Psi\rangle =$  . (D3)

In fact, inverting the color of all rectangles in any 2×2 rectangle does not change the state. For example,

 . (D4)

A consequence of this is that a pair of black rectangles in the same column (row) can "travel" horizontally (vertically) no matter how far they are separated. As an illustration, let us show how to move two black rectangles in the same column separated by one to the neighboring column,



(D5)

This means that these states are indistinguishable from $|\Psi\rangle$ on any finite (system size independent) region. Inverting the color of all rectangles in any 2×2 rectangle in fact defines an equivalence relation on the colorings of the grid: two colorings are equivalent if and only if they can be transformed to each other by repeatedly inverting the color of all rectangles in 2×2 regions. Equivalent colorings correspond to the same state, whereas inequivalent colorings correspond to perpendicular ones: such states all have the form $(1 + Z^{\otimes 4})^{\otimes n} |\phi^{\pm}\rangle^{\otimes n}$. Expanding this expression, we get a sum of tensor products of $|\phi^+\rangle$ and $|\phi^-\rangle$. Starting from two equivalent colorings, the sum contains the same terms reordered. Starting from inequivalent colorings, all terms differ from each other and thus the states are perpendicular as $\langle\phi^+|\phi^-\rangle = 0$. To see that there are at least 2^n equivalence classes, notice that the parity of black rectangles in each column is an invariant. (And for each parity assignment there is a coloring with that parities.)

[1] M. B. Hastings, "Solving gapped Hamiltonians locally," Phys. Rev. B **73**, 085115 (2006); e-print arXiv:cond-mat/0508554.

[2] I. Arad, A. Kitaev, Z. Landau, and U. Vazirani, "An area law and sub-exponential algorithm for 1D systems," e-print arXiv:1301.1162 [quant-ph] (2013).

[3] A. Molnar, N. Schuch, F. Verstraete, and J. I. Cirac, "Approximating Gibbs states of local Hamiltonians efficiently with projected entangled pair states," Phys. Rev. B **91**, 045138 (2015); e-print arXiv:1406.2973 [quant-ph].

[4] S. R. White, "Density matrix formulation for quantum renormalization groups," Phys. Rev. Lett. **69**, 2863–2866 (1992).

[5] S. Ostlund and S. Rommer, "Thermodynamic limit of the density matrix renormalization for the spin-1 heisenberg chain," Phys. Rev. Lett. **75**, 3537–3540 (1995); e-print arXiv:cond-mat/9503107v1.

[6] F. Verstraete, D. Porras, and J. I. Cirac, "DMRG and periodic boundary conditions: A quantum information perspective," Phys. Rev. Lett. **93**, 227205 (2004); e-print arXiv:cond-mat/0404706v1.

[7] I. Arad, Z. Landau, U. Vazirani, and T. Vidick, "Rigorous RG algorithms and area laws for low energy eigenstates in 1D," Commun. Math. Phys. **356**, 65–105 (2017).

[8] F. Verstraete and J. I. Cirac, "Renormalization algorithms for quantum-many body systems in two and higher dimensions," e-print arXiv:cond-mat/0407066 (2004).

[9] J. Jordan, R. Orus, G. Vidal, F. Verstraete, and J. I. Cirac, "Classical simulation of infinite-size quantum lattice systems in two spatial dimensions," Phys. Rev. Lett. **101**, 250602 (2008); e-print arXiv:cond-mat/0703788v4.

[10] B.-X. Zheng, C.-M. Chung, P. Corboz, G. Ehlers, M.-P. Qin, R. M. Noack, H. Shi, S. R. White, S. Zhang, and G. K.-L. Chan, "Stripe order in the underdoped region of the two-dimensional Hubbard model," Science **358**(6367), 1155–1160 (2017).

[11] P. Corboz, T. Rice, and M. Troyer, "Competing states in the t-J model: Uniform D-wave state versus stripe state," Phys. Rev. Lett. **113**, 046402 (2014); e-print arXiv:1402.2859 [cond-mat.str-el].

[12] P. Corboz, A. M. Läuchli, K. Penc, M. Troyer, and F. Mila, "Simultaneous dimerization and SU (4) symmetry breaking of 4-color fermions on the square lattice," Phys. Rev. Lett. **107**, 215301 (2011); e-print arXiv:1108.2857v3 [cond-mat.str-el].

[13] H. J. Briegel and R. Raussendorf, "Persistent entanglement in arrays of interacting particles," Phys. Rev. Lett. **86**, 910–913 (2001); e-print arXiv:quant-ph/0004051.

[14] F. Verstraete and J. I. Cirac, "Valence bond solids for quantum computation," Phys. Rev. A **70**, 060302(R) (2004); e-print arXiv:quant-ph/0311130v1.

[15] I. Affleck, T. Kennedy, E. H. Lieb, and H. Tasaki, "Valence bond ground states in isotropic quantum antiferromagnets," Commun. Math. Phys. **115**, 477–528 (1988).

[16] I. Affleck, T. Kennedy, E. H. Lieb, and H. Tasaki, "Rigorous results on valence-bond ground states in antiferromagnets," Phys. Rev. Lett. **59**, 799–802 (1987).

[17] N. Schuch, D. Poilblanc, J. I. Cirac, and D. Pérez-García, "Resonating valence bond states in the PEPS formalism," Phys. Rev. B **86**, 115108 (2012); e-print arXiv:1203.4816 [cond-mat.str-el].

[18] S. Yang, T. B. Wahl, H.-H. Tu, N. Schuch, and J. I. Cirac, "Chiral projected entangled-pair state with topological order," Phys. Rev. Lett. **114**, 106803 (2016); e-print 1411.6618v1.

[19] D. Poilblanc, J. I. Cirac, and N. Schuch, "Chiral topological spin liquids with projected entangled pair states," Phys. Rev. B **91**, 224431 (2015); e-print arXiv:1504.05236 [cond-mat.str-el].

[20] N. Schuch, J. I. Cirac, and D. Pérez-García, "PEPS as ground states: Degeneracy and topology," Ann. Phys. **325**, 2153–2192 (2010); e-print arXiv:1001.3807.

[21] O. Buerschaper, "Twisted injectivity in PEPS and the classification of quantum phases," Ann. Phys. **351**, 447–476 (2014); e-print arXiv:1307.7763 [cond-mat.str-el].

[22] M. B. Şahinoğlu, D. Williamson, N. Bultinck, M. Mariën, J. Haegeman, N. Schuch, and F. Verstraete, "Characterizing topological order with matrix product operators," e-print arXiv:1409.2150 [quant-ph] (2014).

[23] N. Bultinck, M. Mariën, D. J. Williamson, M. B. Şahinoğlu, J. Haegeman, and F. Verstraete, "Anyons and matrix product operator algebras," Ann. Phys. **378**, 183–233 (2017); e-print arXiv:1511.08090v2.

[24] M. A. Levin and X.-G. Wen, "String-net condensation: A physical mechanism for topological phases," Phys. Rev. B **71**, 045110 (2005); e-print arXiv:cond-mat/0404617.

[25] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac, "Matrix product state representations," Quantum Inf. Comput. **7**, 401 (2007); e-print arXiv:quant-ph/0608197.

[26] J. I. Cirac, D. Perez-Garcia, N. Schuch, and F. Verstraete, "Matrix product density operators: Renormalization fixed points and boundary theories," Ann. Phys. **378**, 100–149 (2017); e-print arXiv:1606.00608.

[27] X. Chen, Z.-C. Gu, and X.-G. Wen, "Classification of gapped symmetric phases in 1D spin systems," Phys. Rev. B **83**, 035107 (2011); e-print arXiv:1008.3745.

[28] N. Schuch, D. Pérez-García, and J. I. Cirac, "Classifying quantum phases using matrix product states and projected entangled pair states," Phys. Rev. B **84**, 165139 (2011); e-print arXiv:1010.3732.

[29] M. Sanz, M. M. Wolf, D. Pérez-García, and J. I. Cirac, "Matrix product states: Symmetries and two-body Hamiltonians," Phys. Rev. A **79**, 042308 (2009); e-print arXiv:0901.2223 [cond-mat.str-el].

[30] F. Pollmann, A. M. Turner, E. Berg, and M. Oshikawa, "Entanglement spectrum of a topological phase in one dimension," Phys. Rev. B **81**, 064439 (2010); e-print arXiv:0910.1811.

[31] M. Fannes, B. Nachtergaele, and R. F. Werner, "Finitely correlated states on quantum spin chains," Commun. Math. Phys. **144**, 443–490 (1992).

[32] B. Nachtergaele, "The spectral gap for some spin chains with discrete symmetry breaking," Commun. Math. Phys. **175**, 565–606 (1996); e-print arXiv:cond-mat/9410110.

[33] Z.-C. Gu, M. Levin, and X.-G. Wen, "Tensor-entanglement renormalization group approach to topological phases," Phys. Rev. B **78**, 205116 (2008); e-print arXiv:0807.2010v1.

[34] O. Buerschaper, M. Aguado, and G. Vidal, "Explicit tensor network representation for the ground states of string-net models," Phys. Rev. B **79**, 085119 (2009); e-print arXiv:0809.2393v1.

[35] D. Pérez-García, M. Sanz, C. E. Gonzalez-Guillen, M. M. Wolf, and J. I. Cirac, "A canonical form for projected entangled pair states and applications," New J. Phys. **12**, 025010 (2010); e-print arXiv:0908.1674 [quant-ph].

[36] P. Anderson, "Resonating valence bonds: A new kind of insulator?," Mater. Res. Bull. **8**, 153–160 (1973).

[37] F. Verstraete, M. M. Wolf, D. Pérez-García, and J. I. Cirac, "Criticality, the area law, and the computational power of projected entangled pair states," Phys. Rev. Lett. **96**, 220601 (2006); e-print arXiv:quant-ph/0601075.

[38] X. Chen, Z.-X. Liu, and X.-G. Wen, "2D symmetry protected topological orders and their protected gapless edge excitations," Phys. Rev. B **84**, 235141 (2011); e-print arXiv:1106.4752.

[39] D. J. Williamson, N. Bultinck, M. Mariën, M. B. Sahinoglu, J. Haegeman, and F. Verstraete, "Matrix product operators for symmetry-protected topological phases: Gauging and edge theories," Phys. Rev. B **94**, 205150 (2016); e-print arXiv:1412.5604v3.

[40] Z.-C. Gu and X.-G. Wen, "Tensor-entanglement-filtering renormalization approach and symmetry protected topological order," Phys. Rev. B **80**, 155131 (2009); e-print arXiv:0903.1069v2.

[41] Z. Y. Xie, J. Chen, J. F. Yu, X. Kong, B. Normand, and T. Xiang, "Tensor renormalization of quantum many-body systems using projected entangled simplex states," Phys. Rev. X **4**, 011025 (2014); e-print arXiv:1307.5696.

[42] D. E. Evans and R. Hoegh-Krohn, "Spectral properties of positive maps on C*-algebras," J. London Math. Soc. **s2–17**, 345–355 (1978).

[43] M. Wolf, Quantum channels and operations guided tour. Lecture notes available at: http://www-m5.ma.tum.de/foswiki/pub/M5/Allgemeines/MichaelWolf/QChannelLecture.pdf.

[44] R. Orus, "A practical introduction to tensor networks: Matrix product states and projected entangled pair states," Ann. Phys. **349**, 117–158 (2014); e-print arXiv:1306.2164 [cond-mat.str-el].

[45] M. Sanz, D. Perez-Garcia, M. M. Wolf, and J. I. Cirac, "A quantum version of Wielandt's inequality," IEEE Trans. Inf. Theory **56**, 4668–4673 (2010); e-print arXiv:0909.5347.

[46] A. Cadarso, M. Sanz, M. M. Wolf, J. I. Cirac, and D. Perez-Garcia, "Entanglement, fractional magnetization and long-range interactions," Phys. Rev. B **87**, 035114 (2013); e-print arXiv:1209.3898.

[47] G. D. las Cuevas, T. S. Cubitt, J. I. Cirac, M. M. Wolf, and D. Pérez-García, "Fundamental limitations in the purifications of tensor networks," J. Math. Phys. **57**, 071902 (2016); e-print arXiv:1512.05709v2.

[48] C. H. Bennett, S. Popescu, D. Rohrlich, J. A. Smolin, and A. V. Thapliyal, "Exact and asymptotic measures of multipartite pure-state entanglement," Phys. Rev. A **63**, 012307 (2001).

[49] X. Chen, Z.-C. Gu, Z.-X. Liu, and X.-G. Wen, "Symmetry protected topological orders and the group cohomology of their symmetry group," Phys. Rev. B **87**, 155114 (2013); e-print arXiv:1106.4772.

[50] M. Nagata, "On the nilpotency of nil-algebras," J. Math. Soc. Jpn. **4**, 296–301 (1952).

C.2 Computational speedups using small quantum devices

Computational speedups using small quantum devices

Vedran Dunjko, Yimin Ge, and J. Ignacio Cirac

In this work, we consider the question whether small quantum computers with only M qubits can be useful for speeding up certain classical algorithms, even when the problem size is much larger than M. We develop a hybrid quantum-classical algorithm that polynomially speeds up Schöning's algorithm for solving 3SAT involving $n \gg M$ variables.

Many classical 3SAT algorithms significantly faster than brute-force search are known. Their runtimes can usually be characterised by a constant $\gamma \in (0,1)$, meaning that they solve 3SAT in $O^*(2^{\gamma n})$. One of the best and most famous ones is the randomised bounded-error algorithm of Schöning with runtime $O^*(2^{\gamma_0 n})$, where $\gamma_0 \approx 0.415$. Given a quantum computer with only $M \ll n$ qubits, we however first demonstrate that a straightforward approach to speed up this algorithm results in a "threshold effect": if M/n is below some constant value, a naive hybrid algorithm turns out to be *slower* than its classical version.

Our central result is a hybrid algorithm which, given a quantum computer with $M = cn$ qubits, where $c \in (0,1)$ is an arbitrary constant, solves 3SAT with n variables in a runtime of $O^*(2^{(\gamma_0 - f(c) + \varepsilon)n})$, where $f(c) > 0$ is a constant and ε can be made arbitrarily small.

Our hybrid algorithm is a quantum-assisted version of derandomised variants of Schöning's algorithm which use the *space splitting algorithm* to reduce 3SAT to the problem of finding a satisfying assignment within an r-ball (with respect to the Hamming distance) of a trial assignment $\mathbf{x}$, known as Promise-Ball-SAT (PBS). Two recursive algorithms for PBS, called PromiseBall and FastBall with runtimes $O^*(3^r)$ and $O^*(2^r)$, respectively, are known. Both algorithms recursively call themselves with ever smaller values of r. Moreover, using FastBall in the space splitting algorithm leads to an overall 3SAT-solving runtime which essentially matches that of the randomised version. To obtain the hybrid speedup, we first develop a quantum algorithm for PBS called QBall which is quadratically faster than PromiseBall and only uses $O(r \log(n/r) + r + \log n)$ qubits. Then, in FastBall, we replace the classical recursive call with a call to QBall once r becomes small enough for the available M qubits.

The main technical part of this work is the implementation of QBall, detailed in Section A of the supplemental material. While the common strategy of using amplitude amplification to quadratically enhance PromiseBall may be obvious, the main difficulty is to do this using sufficiently few qubits such that genuine speedups arise. Indeed, straightforward quantum algorithms based on PromiseBall use at least $\Omega(n)$ qubits, even for small values of r, rendering them useless when only $M = cn$ qubits are available.

To implement QBall, we first develop a (classical) non-recursive variant of PromiseBall which takes some $\vec{s} \in S := \{0,1,2\}^r$ as input and maps it to a set V of at most r variables with the following property: there exists a satisfying assignment within Hamming distance r of $\mathbf{x}$ if and only if there exists some $\vec{s} \in S$ such that $\mathbf{x}_V$, the assignment obtained from $\mathbf{x}$ by flipping the variables in V, is a satisfying assignment. This allows us to quantum search over S in a runtime of $O^*(3^{r/2})$. However, the straightforward implementation in which all variables are part of the workspace uses $\Omega(n)$ qubits.

Next, we reduce the number of qubits in several steps. First, classical inputs can be hardwired into the quantum circuit instead of taking up input qubits. Second, given a quantum register containing an encoding of V, checking whether $\mathbf{x}_V$ is a satisfying assignment as well as selecting the next variable to be added to V can both be done using only $O(\log n)$ reuseable qubits. However, naive encodings of V use $O(r \log n)$ qubits. It is easy to see that this is still too large, i.e. does not result in a constant decrease of γ but only a subpolynomial speedup.

The last qubit reduction comes from using more efficient data structures to encode V, detailed in Section A of the supplemental material. It is easy to see that $O(r \log(n/r) + r)$ bits suffice in principle to describe V. However, straightforward use of such encodings adds significant obstacles: most operations on such encodings become inherently non-reversible and hence difficult to implement as quantum operations without introducing large ancilla spaces. Straightforward ways to circumvent this issue without using too many qubits turn out to have detrimental runtime overheads. We solve this by splitting V into $O(\log r)$ suitable subsets and uncomputing only the subsets when necessary.

Statement of individual contribution

The work is the result of frequent discussions between Vedran Dunjko, J. Ignacio Cirac, and myself. Vedran Dunjko, who is the principal author of this article, had the idea of exploiting the recursive structure of the derandomised version of Schöning's algorithm to obtain a speedup using few qubits. In particular, it was his idea to use a quantum-enhanced version of PromiseBall sufficiently deep in the recursion tree of FastBall. My contribution to this work was predominantly in carrying out the detailed technical work to make these ideas work. First, I recognised the reversibility obstacle, and subsequently developed the data structures for efficient set-encodings as well as the time- and space-efficient implementations of the accompanying subroutines (detailed in Section A of the supplemental material). Second, I carried out the precise runtime analysis, linking in particular the degree of the speedup to the value of c. Vedran Dunjko was in charge of writing the main text of the article, while I was in charge of writing the supplemental material.

Permission to include:

Vedran Dunjko, Yimin Ge, and J. Ignacio Cirac.
Computational speedups using small quantum devices.
Physical Review Letters, 121, 250501 (2018).

December 2017

APS Copyright Policies and Frequently Asked Questions

(…)

As the author of an APS-published article, may I include my article or a portion of my article in my thesis or dissertation?

Yes, the author has the right to use the article or a portion of the article in a thesis or dissertation without requesting permission from APS, provided the bibliographic citation and the APS copyright credit line are given on the appropriate pages.

(…)

FAQ Version: December 12, 2017

https://journals.aps.org/copyrightFAQ.html 03/12/2019

Editors' Suggestion   Featured in Physics

Computational Speedups Using Small Quantum Devices

Vedran Dunjko,[1,2,*] Yimin Ge,[1,†] and J. Ignacio Cirac[1,‡]
[1]*Max Planck Institut für Quantenoptik, Hans-Kopfermann-Straße 1, 85748 Garching, Germany*
[2]*LIACS, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, Netherlands*

(Received 24 July 2018; revised manuscript received 14 September 2018; published 18 December 2018)

Suppose we have a small quantum computer with only M qubits. Can such a device genuinely speed up certain algorithms, even when the problem size is much larger than M? Here we answer this question to the affirmative. We present a hybrid quantum-classical algorithm to solve 3-satisfiability problems involving $n \gg M$ variables that significantly speeds up its fully classical counterpart. This question may be relevant in view of the current quest to build small quantum computers.

DOI: 10.1103/PhysRevLett.121.250501

Quantum computers use the superposition principle to speed up computations. However, it is not clear if they can be useful when they are, as expected for the foreseeable future, limited in size. The reason is that quantum (and classical) algorithms typically exploit global structures of problems, and restricting superpositions to certain block sizes will break that structure. Thus, for problems where arbitrarily sized quantum computers offer advantages, small quantum computers may be of no significant help given large inputs.

Here, we study this problem and show that this is not generally true: There are relevant algorithms that utilize the global structure, where quantum computers significantly smaller than the problem size can offer significant speedups. More precisely, we focus on the famous algorithm of Schöning for boolean satisfiability and present a modified hybrid quantum-classical algorithm that significantly outperforms its purely classical version, even given small quantum computers.

Satisfiability (SAT) problems are among the basic computational problems, and they naturally appear in many contexts involving combinatorial optimization, like scheduling or planning tasks, and in statistical physics. A prominent SAT problem is 3SAT, which involves clauses with up to three literals. 3SAT is the canonical example of the so-called NP-complete problems, believed to be exponentially difficult even for quantum computers. Nevertheless, quantum computers can still accelerate their solving [1,2], and given their ubiquity, they may become one of the most important applications of quantum computers. However, the best quantum algorithms, which "quantum-enhance" classical SAT solvers [1], require many qubits and are not directly applicable given small quantum computers. There are several possibilities for how to use a limited-size quantum device. For instance, one could speed up smaller, structure-independent subroutines, which occur, e.g., in the preparation phases of algorithms (e.g., whenever a search over a few items is performed, one could utilize Grover's algorithm [3]). However, for "genuine" speedups, e.g., those that interpolate between the runtimes of the fully classical and a fully quantum algorithm, according to the size of the available quantum device, one should attack the actual computational bottlenecks. As we show, if this is done straightforwardly, one may encounter a threshold effect: If the quantum device is too small, i.e., can handle only a small fraction of the instance, a naïve hybrid algorithm turns out to be slower than its classical version. Here, we demonstrate how this effect can sometimes be circumvented, in the context of satisfiability problems. Specifically, we provide a quantum-assisted version of a well-understood classical algorithm, which achieves genuine improvements given quantum computers of basically any size, avoiding the threshold effect [4]. Our results are applicable to broader kSATproblems and, more generally, highlight the characteristics of classical algorithms and methods, which can be exploited to provide threshold-free enhancements.

The 3SAT problems.—In SAT problems, we are given a boolean formula $F\colon\{0,1\}^n \to \{0,1\}$ over n binary variables $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$. The task is to find a satisfying assignment $\mathbf{x}$, i.e., fulfilling $F(\mathbf{x}) = 1$, if one exists. In 3SAT, F is defined by a set of L clauses $\{C_j\}$, where each clause specifies three literals $\{l_1^j, l_2^j, l_3^j, \}$. Each literal specifies one of the n binary variables (x) or a negated variable $(\bar{x})$; for instance, $\{l_1, l_2, l_3\}$ could be $\{x_3, \bar{x}_5, x_8\}$. An assignment of variables $\mathbf{x}$ thus sets the values of all literals. C_j is satisfied by $\mathbf{x}$ if any of the literals in C_j attains the value 1. A formula F, written as $F(\mathbf{x}) = \bigwedge_{j=1}^{L}(l_1^j \vee l_2^j \vee l_3^j)$ using standard logic operator notation, is satisfied by $\mathbf{x}$ if all its clauses are satisfied.

Classical algorithms.—Many classical 3SAT solvers are significantly faster than the brute-force search. Their performance can be characterized by a constant $\gamma \in (0,1)$, meaning that they solve 3SAT in a runtime of $O^*(2^{\gamma n})$ [6]. One of the best and most famous ones is the algorithm of

Schöning [8]. It initializes a random assignment of the variables, then repeatedly finds an unsatisfied clause, randomly selects one literal in that clause, and flips the corresponding variable. This sampling algorithm terminates once a satisfying assignment is reached or once this process is iterated $O(n)$ times. Schöning proved that the probability of this algorithm finding a satisfying assignment (if one exists) is at least $(3/4)^n$, which, by iteration, leads to a Monte Carlo algorithm with expected runtime $O^*(2^{\gamma_0 n})$ with $\gamma_0 \coloneqq \log_2(4/3) \approx 0.415$. A significant speedup of the classical algorithm is a reduction of its value of γ. To study the potential of small quantum computers, we investigate whether small devices suffice to achieve such a reduction.

Straightforward hybrid algorithms for small quantum computers.—In [1] a quantum algorithm inspired by Schöning's method, which exploits amplitude amplification [9], was introduced. It solves 3SAT instances with n variables in runtime $O^*(2^{\gamma_0 n/2})$ and requires $\approx \beta n$ qubits for a $\beta > 2$. Given a quantum computer with only $M \approx \beta m$ qubits ($m \ll n$), one has a few options to achieve speedups. A "bottom up" approach would be to use the quantum algorithm as an m-variable instance solver, which is then used as a subroutine in an overarching classical algorithm. For instance, to tackle the problem of n variables, one could sequentially go through all possible partial assignments of $n - m$ variables. Each partial assignment induces a SAT problem with m variables, which could then be solved on the small quantum device. The runtime of such an algorithm is $O^*(2^{(n-m)+\gamma_0 m/2})$, which highlights the *threshold effect*: The hybrid algorithm becomes *slower* than the classical algorithm of Schöning if $m/n \lesssim 0.74$. Note that, unlike thresholds for speedups induced by prefactors suppressed in an asymptotic analysis and the O^* notation, which may prevent speedups for small instances, the present threshold effect is far more fundamental: If M/n is below a constant value, no speedup is possible even for arbitrarily large problem sizes. Roughly speaking, the main problem of hybrid algorithms using a small device as a subinstance solver is that they break the global structure exploited by the classical algorithm. This results in hybrid algorithms whose runtimes interpolate between a fully quantum runtime and something slower than the classical algorithm—hence the threshold effect. Alternatively, we explore a "top-down" approach, where the most computationally expensive subroutines of the classical algorithm are identified and quantum enhanced.

Our results.—We present a hybrid algorithm that avoids the threshold effect. Specifically, given a quantum computer with $M = cn$ qubits, where $c \in (0,1)$ is an arbitrary constant, our algorithm solves 3SAT with n variables in a runtime of $O^*(2^{(\gamma_0 - f(c) + \varepsilon)n})$, where $f(c) > 0$ is a constant and ε can be made arbitrarily small. The function $f(c)$ is involved, but it is almost linear for small c (see the Supplemental Material [10] for details). Critically, irrespective of its exact form, our result constitutes a polynomial speedup over Schöning's algorithm for arbitrarily small c. Our contribution is primarily conceptual, and we assume an error-free setting.

Algorithm description.—Our hybrid algorithm is a quantum-assisted version of de-randomized variants of Schöning's algorithm [7,11], reviewed next. Given a bitstring $\mathbf{x} \in \{0,1\}^n$, let $B_r(\mathbf{x})$ denote the r-ball centered at $\mathbf{x}$, i.e., the set of all bitstrings $\mathbf{y}$ differing from $\mathbf{x}$ in at most r positions (i.e., their Hamming distance is $\leq r$). Then, relying on coding theory, the space of possible assignments is covered by a number of r-balls. Given this covering set, specified by the centers of the balls, the algorithm sequentially checks whether there exists a satisfying assignment within each of the r-balls. This "space-splitting" algorithm reduces SAT to the problem of finding a satisfying assignment within an r-ball, called Promise-Ball-SAT (PBS). A deterministic algorithm PROMISEBALL$(F, r, \mathbf{x})$ for PBS was introduced in [11]. This is a simple, recursive divide-and-conquer algorithm: On input it takes a formula, specified by a set of clauses with at most three literals, a radius, and a center $\mathbf{x}$. The algorithm first checks some conditions for (un)satisfiability [if $r \leq 0$ and $F(\mathbf{x}) = 0$ or if any clause is empty], or if $\mathbf{x}$ is a satisfying assignment. Otherwise, in the recursive step, it finds the first unsatisfied clause C and calls PROMISEBALL$(F_{|l=1}, r-1, \mathbf{x})$ for every literal $l \in C$ (the variables, literals, and clauses are enumerated in some prespecified order). Here, $F_{|l=1}$ denotes the formula obtained by setting the variable corresponding to l to the value ensuring $l = 1$; i.e., all clauses involving l $(\bar{l})$ are removed (truncated). This algorithm solves PBS in time $O^*(3^r)$. For comparison, Schöning sampling solves it in $O^*(2^r)$. The overall runtime of the space-splitting algorithm of [11] can be expressed as a function of the runtime of the PBS-solving subroutine (see Supplemental Material [10]). What is relevant is that, whenever a PBS solver with runtime $O^*(2^r)$ (e.g., randomized Schöning) is used in the space-splitting algorithm, we recover Schöning's runtime with $\gamma_0 \approx 0.415$.

Note that every recursive call in PromiseBall reduces r by 1. To make use of a small quantum device, one can, once r becomes small enough, use a quantum algorithm for PBS instead of a classical call. This leads to a general approach to speed up algorithms, which recursively call themselves (or other subroutines) with ever-decreasing instance sizes, using small quantum devices. We call this the *standard hybrid approach*. However, there are two obstacles to consider. First, since PromiseBall is significantly slower than the algorithm of Schöning, this still leads to a threshold: M would have to be a large fraction of n to gain an advantage. Second, straightforward quantum implementations of PromiseBall require too many qubits $[\Omega(n)]$, even if r is small. While it is not difficult to reduce this to $O(r \log n)$, the resulting hybrid algorithm, although avoiding a threshold, would have a very low, subpolynomial, advantage and not yield a true improvement of γ. We circumvent this by using more involved memory structures combined with specialized algorithms which algorithmically delete unneeded information, leading to much better memory requirements of $O(r\log(n/r)+r+\log n)$, and a true speed-up.

To summarize, we next provide the following algorithms: (1) a nonrecursive variant of PromiseBall, which on input only takes r ternary choices of which literals to flip; (2) a straightforward reversible implementation of (1) needed for quantization, with two parts—QBALL_1 transforms the choices to corresponding variable labels, and QBALL_2 checks whether flipping the chosen variables in $\mathbf{x}$ (ball center) satisfies the formula; (3) a method to significantly reduce the memory requirements of storing a set of labels in the second algorithm, from $O(r\log n)$ to $O(r\log(n/r)+r)$, while preserving reversibility. Algorithms (1)–(3), combined with amplitude amplification, then form the quantum algorithm qball for PBS, which, combined with a faster classical PBS-based solver [7], forms our final algorithm (4).

Space- and time-efficient quantum algorithm for PBS.—First, we specify a nonrecursive (classical) variant of PromiseBall, which does not manipulate the entire formulas explicitly. This avoids an immediate need for $\Omega(n)$ qubits for "carrying" the formula as input. Afterwards, we optimize the memory required for a reversible implementation and use amplitude amplification to achieve a faster quantum algorithm.

The structure of PromiseBall yields a ternary tree of depth r, induced by the up to three choices of literals in the recursive step of the algorithm. Thus, a sequence of choices $s_1,\dots,s_r\in\{1,2,3\}$ specifies a leaf in the tree and hence the subset of literals whose values have been flipped. Thus, the algorithm PromiseBall induces a mapping from $s_1,\dots,s_r$ to a set of at most r variables to be flipped, denoted $V=\{v_1,\dots,v_{r'}\}$, where $r'\le r$. The nonrecursive algorithm computes the list of variables V indexed by the sequence $\vec{s}=s_1,\dots,s_r$, generates the candidate assignments $\mathbf{x}_V$ realized by flipping the values of the variables specified by V, and checks if they satisfy the formula. This subroutine can be executed in polynomial time. The nonrecursive PromiseBall simply goes through all 3^r sequences $\vec{s}$, yielding the runtime $O^*(3^r)$.

This can be turned into a quantum algorithm QBALL_{12}, realizing the mapping:

$$|s_1,\dots,s_r\rangle|0\rangle|0\rangle \xrightarrow{\mathrm{QBALL}_{12}} \underbrace{|s_1,\dots,s_r\rangle\mathrlap{\overbrace{\phantom{|V\rangle|F(\mathbf{x}v)\rangle}}^{\mathrm{QBALL}_2}}|V\rangle}_{\mathrm{QBALL}_1}|F(\mathbf{x}v)\rangle, \qquad (1)$$

where the first part (QBALL_1) generates the indices of the variables in V, and the second part (QBALL_2) verifies whether F is satisfied by $\mathbf{x}_V$. The full quantum algorithm for PBS, which we call qball, then uses amplitude amplification to find one sequence $\vec{s}$ which yields a satisfying assignment, using $O(3^{r/2})$ calls to $\mathrm{QBALL}_{1,2}$, each with polynomial runtime. We next give the basic ideas for how to implement the algorithm space efficiently (for details see the Supplemental Material [10]). For ease of presentation, we first show how QBALL_2 can be realized straightforwardly, using many ancillas, before reducing their number. A straightforward implementation of QBALL_2 would utilize n additional qubits and assign them the value $\mathbf{x}$. Then, the circuit would iterate through the registers specifying V [naïvely requiring $O(r\log n)$ qubits] and introduce (controlled) negations to those ancillary qubits selected by the values in V. This would finalize the variable presetting stage and set the input to the formula to $\mathbf{x}_V$. Next, we would sequentially evaluate each clause, by associating a gate controlled by the variable qubits corresponding to the variables occurring in the clause. This controlled gate applies the appropriate negations to realize the literals, increasing a counter if a clause is satisfied. After doing this for each clause, the output qubit is flipped only if the counter equals L, meaning all clauses are satisfied. Such a circuit uses $O(\log(L)+n)$ ancillas. Since $L=O(\mathrm{poly}(n))$, the key problem is the contribution of n. This is simplified by noting that, although the circuits our algorithms generate depend on F, $\mathbf{x}$ and r, we can w.l.o.g. assume that $\mathbf{x}=(0,\dots,0)$ and subsume the negations directly into F [12]. Next, since the clauses are evaluated sequentially, we only require three variable-specific ancillas, specifying the bit values appearing in the current clause: For each clause, each of the three ancillas corresponds to the three variables occurring in that clause. The variable presetting stage is now done individually for each clause C_j: Before clause evaluation, the circuit iterates through the V-specifying register and flips the kth ancilla if the specification matches the kth variable within the clause C_j. The three ancillas are uncomputed after evaluating C_j and can be reused. This requires only $O(\log n)$ additional qubits.

The algorithm QBALL_1 is more involved. QBALL_1 comprises a main loop which sequentially adds one variable specification v_i to the already specified set as follows: The ith circuit block takes the specifications of the first $i-1$ variables $v_1,\dots,v_{i-1}$ as inputs, iterates through all clauses, and evaluates each clause C_j (in a manner similar to QBALL_2), using the values $v_1,\dots,v_{i-1}$ to correctly preset the clause-specific input. If the clause is not satisfied, it uses the value of s_i to select the specification of one variable occurring in C_j, taking into account that variables which have already been flipped cannot be selected again, and storing this specification as v_i. For reversibility, additional counters are used, but these can be uncomputed and recycled. The final compression relies on efficient encodings of V, which, as an ordered list, would use $O(r\log n)$ qubits. Since the ordering does not matter, instead of storing the positions, we can store the relative shifts $v_1, v_2-v_1,\dots,v_{r'}-v_{r'-1}>0$ of a sorted version of the list, using no more digits than necessary and a separation symbol to denote the next number. Since these values add up to at most n, one can see that $O(r\log(n/r)+r)$ qubits suffice for this encoding. This structure indeed encodes a set, erasing the ordering. However, straightforward algorithms that use encodings of sets instead of lists encounter reversibility problems. To illustrate this, note that in the process of adding a new variable to V, one must realize the two steps $|\{v_1,\ldots,v_{i-1}\}\rangle|0\rangle \mapsto |\{v_1,\ldots,v_{i-1}\}\rangle|v_i\rangle$, i.e., finding the new element, and $|\{v_1,\ldots,v_{i-1}\}\rangle|v_i\rangle \mapsto |\{v_1,\ldots,v_i\}\rangle|0\rangle$, i.e., placing it into the set, and, critically, freeing the ancillary qubits for the next step. However, this is irreversible since the information about which element was added last is lost. The full ordering information requires $O(r\log r)$ additional qubits, nullifying all advantages. Of course, one could instead realize the reversible operation $|\{v_1,\ldots,v_{i-1}\}\rangle|v_i\rangle|0\rangle \mapsto |\{v_1,\ldots,v_{i-1}\}\rangle|v_i\rangle|\{v_1,\ldots,v_i\}\rangle$, followed by applying the inverse of the entire circuit up to this point to uncompute $|\{v_1,\ldots,v_{i-1}\}\rangle|v_i\rangle$, but this would result in an exponential instead of polynomial runtime of QBALL_1. We circumvent this issue by splitting V into $O(\log r)$ sets of sizes $1,2,4,\ldots$, and the loading of each larger block is followed by an algorithmic deletion of all smaller blocks. This ensures the overall number of qubits needed for this encoding is still $O(r\log(n/r)+r)$, at the cost of $2^{O(\log(r))}$ additional steps, which is just polynomial. These structures and primitives lead to an overall space- and time-efficient implementation of QBALL_1 (see Supplemental Material [10] for details). Combining these subroutines with a quantum search over $\vec{s}$, we obtain the algorithm QBALL, solving PBS in time $O^*(3^{r/2})$ and using $O(r\log(n/r)+r+\log n)$ qubits. In particular, QBALL outperforms the randomized algorithm of Schöning for PBS (2^r vs. $2^{\log_2(3)r/2}\approx 2^{0.79r}$).

Hybrid algorithm for 3SAT.—We could now use the standard hybrid approach for PromiseBall, i.e., call QBALL instead of PromiseBall when r is sufficiently small. However, this still leads to a threshold. This is resolved by using an improved deterministic algorithm for PBS [7], where coding theory is applied to cover the space of choice vectors $\vec{s}$. This yields an algorithm with runtime $O^*((2+\epsilon)^r)$, where ϵ can be chosen arbitrarily small—thus, the runtime of this algorithm for PBS essentially matches the runtime of Schöning. While we do not need the details of this algorithm, the critical point is that, like PromiseBall, it recursively calls itself to solve PBS with ever smaller values of r (sequentially reduced by a quantity depending on ϵ). Now, we can apply the standard hybrid approach. Since QBALL beats the runtime of this improved classical algorithm for PBS, the hybrid algorithm is faster than Schöning's, and, unlike using PromiseBall, there is no threshold induced by slow classical algorithms.

To estimate the runtime of our algorithm, note that since QBALL only requires $O(r\log(n/r)+r+\log n)$ qubits, a device with $M=cn$ qubits can solve PBS for $r=\beta(c)n$ for some $\beta(c)>0$. Since the hybrid algorithm replaces a classical subroutine of runtime $O^*(2^r)$ with a subroutine of runtime $O^*(3^{r/2})$ in a recursion tree below depth $r=\beta(c)n$, the runtime of the hybrid algorithm beats that of the classical one by a factor of $O^*((\sqrt{3}/2)^{\beta(c)n})$. Thus, the combined runtime of the hybrid algorithm is $O^*(2^{(\gamma_0-f(c)+\epsilon)n})$ for $f(c)=\log_2(2/\sqrt{3})\beta(c)\approx 0.21\beta(c)$. The forms of β and f are involved, but in the relevant regime of small c, $f(c)$ achieves almost linear scaling, specifically $f(c)=\Theta(c/\log(c^{-1}))$ (for details see Supplemental Material [10]).

Conclusions.—We have shown that a small quantum computer can speed up relevant classical algorithms even for significantly larger inputs. While obvious for structureless scenarios (e.g., unstructured search), when considering algorithms that use the problem's structure, like Schöning's algorithm, speedups are nontrivial: The way the problem is partitioned must maintain the algorithm's structure to avoid thresholds. Our algorithm achieves a significant speedup, namely, a reduction of the relevant parameter γ, characterizing runtimes of the form $O^*(2^{\gamma n})$. The speedup holds relative to a variant of Schöning's algorithm. Our results, however, generalize to other algorithms based on Schöning's approach (e.g., Refs. [13,14], since those rely on better initial assignments) and to the variants handling kSAT ($k>3$). Historically, the best classical SAT solvers with provable bounds are based either on the ideas of Schöning or on the approach of [15], which includes the current record holder [16]. It would be interesting to see whether this second class of algorithms is also amenable to the types of enhancements achieved here. The broader question of this work is becoming increasingly more relevant given the current progress in prototypes of small quantum computers [17–19]. As our contribution is conceptual, we assume an error-free scenario. Still, our results may also have pragmatic relevance. Indeed, while the number of physical qubits of implementations is rapidly growing, the number of protected logical qubits we may expect in the near term is likely to be very limited. In practice, both the overheads and the noise may be bottlenecks to exploit small devices in general [20]. Thus, it would be particularly interesting to optimize the overheads of our algorithm. Specifically, any methods to decrease the number of gates and ancillas would increase the tolerance of our scheme. Finally, an in-depth analysis of optimal device-specific implementations could further help make our algorithms suitable for near-term realizations.

V. D. acknowledges support from the Alexander von Humboldt Foundation. J. I. C. and Y. G. acknowledge the ERC Advanced Grant QENOCOBA under the EU Horizon 2020 program (Grant Agreement No. 742102).

[*]v.dunjko@liacs.leidenuniv.nl
[†]yimin.ge@mpq.mpg.de
[‡]ignacio.cirac@mpq.mpg.de

[1] A. Ambainis, ACM SIGACT News **35**, 22 (2004).

[2] A. Ambainis, K. Balodis, J. Iraids, M. Kokainis, K. Prūsis, and J. Vihrovs, arXiv:1807.05209.

[3] L. K. Grover, in *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing, STOC '96* (ACM, New York, NY, USA, 1996), pp. 212–219.

[4] Our scenario is related to, but distinct from, Ref. [5] which considers how a classical computer can help simulate a given quantum algorithm which requires only slightly more qubits than are available. In contrast, we change the algorithm, i.e., propose hybrid algorithms which use a significantly smaller quantum device, to speed up a fixed classical algorithm. The two approaches are also complementary in their applicability: The main results of [5] are most powerful for shallow computations and could only be directly applied to SAT solving if one has an exponentially sized quantum computer to begin with, as the computation may be exponentially deep.

[5] S. Bravyi, G. Smith, and J. A. Smolin, Phys. Rev. X **6**, 021043 (2016).

[6] The O^* notation suppresses the terms that contribute only polynomially; see, e.g., Ref. [7].

[7] R. A. Moser and D. Scheder, in *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing, STOC '11* (ACM, New York, NY, USA, 2011), pp. 245–252.

[8] U. Schöning, in *Proceedings of the 40th Annual Symposium on Foundations of Computer Science* (IEEE Computer Society, Washington, DC, 1999), pp. 410–414.

[9] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp, arXiv:quant-ph/0005055.

[10] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.121.250501, for the details of the algorithms and of the runtime analysis.

[11] E. Dantsin, A. Goerdt, E. A. Hirsch, R. Kannan, J. Kleinberg, C. Papadimitriou, P. Raghavan, and U. Schöning, Theor. Comput. Sci. **289**, 69 (2002).

[12] Let $F_{\mathbf{x}}$ be the formula obtained by negating any literal in F which corresponds to a variable to which $\mathbf{x}$ assigns the value 1, so that $F_{\mathbf{x}}(0, \ldots, 0) = F(\mathbf{x})$. By subsuming $\mathbf{x}$ into F, we mean that we use $F_{\mathbf{x}}$ instead of F.

[13] T. Hofmeister, U. Schöning, R. Schuler, and O. Watanabe, in *STACS 2002*, edited by H. Alt and A. Ferreira (Springer, Berlin, Heidelberg, 2002), pp. 192–202.

[14] K. Iwama, K. Seto, T. Takai, and S. Tamaki, in *Algorithms and Computation*, edited by O. Cheong, K.-Y. Chwa, and K. Park (Springer, Berlin, Heidelberg, 2010), pp. 73–84.

[15] R. Paturi, P. Pudlák, M. E. Saks, and F. Zane, J. ACM **52**, 337 (2005).

[16] T. Hertli, SIAM J. Comput. **43**, 718 (2014).

[17] IEEE spectrum, IBM edges closer to quantum supremacy with 50-qubit processor, https://spectrum.ieee.org/tech-talk/computing/hardware/ibm-edges-closer-to-quantum-supremacy-with-50qubit-processor.

[18] American Physical Society meeting, Engineering superconducting qubit arrays for quantum supremacy, http://meetings.aps.org/Meeting/MAR18/Session/A33.1.

[19] Intel newsroom. 2018 ces: Intel advances quantum and neuromorphic computing research, https://newsroom.intel.com/news/intel-advances-quantum-neuromorphic-computing-research.

[20] J. Preskill, Quantum **2**, 79 (2018).

Computational speedups using small quantum devices: Supplemental Materials

Vedran Dunjko,[1, 2, *] Yimin Ge,[1, †] and J. Ignacio Cirac[1, ‡]

[1] *Max Planck Institut für Quantenoptik, Hans-Kopfermann-Str. 1, 85748 Garching, Germany*
[2] *LIACS, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands*

(Dated: November 29, 2018)

In the supplemental materials we provide the details referred to in the main text. In Section A, we provide the detailed exposition of the central quantum algorithm QBall. In Section B, we provide the details of the full hybrid algorithm and a detailed runtime analysis.

A. Detailed quantum algorithm for PBS

In this section, we describe the quantum algorithm for PBS. Let $r \geq 0$ be an integer and F a 3SAT formula over n variables. The problem is to decide whether there exists a satisfying assignment of F with hamming weight at most r.

As explained in the main text, the most involved part of the algorithm is a subroutine called QBall$_1$ which takes as input $s_1, \ldots, s_r \in \{1,2,3\}$ and outputs a set $V = V(\vec{s})$ of at most r variables with the property that F has a satisfying assignment with hamming distance at most r if and only if there exists an $\vec{s} \in \{1,2,3\}^r$ such that the assignment $\mathbf{x}(V(\vec{s}))$, obtained by setting all variables in $V(\vec{s})$ to 1 and all other ones to 0, is a satisfying assignment.

The quantum algorithm for PBS is a quantum version of the non-recursive variant of the PromiseBall algorithm described in the main text, which is quantized using amplitude amplification. We will therefore first develop a classical *reversible* circuit (which depends on F and r) for the QBall$_1$ subroutine, which maps $\vec{s}$ to a space-efficient encoding of the set $V(\vec{s})$, using at most $O(r\log(n/r) + r + \log n)$ ancilla bits. Although the algorithm below is, for now, purely classical (and reversible), we will nevertheless use bra/ket notation for simplicity.

1. *Key ideas of* QBall$_1$

Before going into the particulars of the algorithm, it is worthwhile to highlight the key ideas we utilize, and explain why we go into such detail to explain a relatively simple algorithm. As mentioned in the main text, given r, QBall$_1$ can be summarized as follows:

```
Set $V = \emptyset$.
For $i = 1$ to $r$ do begin
    Set $\mathbf{x}(V)$ to be the assignment obtained by setting variables in $V$ to 1 and all other ones to 0.
    Find the first clause in $F$ which is not satisfied by $\mathbf{x}(V)$.
    If such a clause exists, use $s_i$ to select the next variable $v_i$. Else, set $v_i$ to a dummy variable (index beyond $n$).
    Update $V := V \cup \{v_i\}$.
end.
```

The output V of the above process collects specifications of r variables, and the variables with indices below or equal to n specify the satisfying assignment, if one exists. We use dummy variables as this avoids the need for controlled operations which could change the algorithm's behaviour (*e.g.* terminate the loop once no suitable clause is found), or which require more memory to implement reversibly. The majority of subtleties in our algorithms pertain precisely to that latter requirement: utilizing as few ancillas as possible while at the same time maintaining reversibility.

As mentioned in the main text, one of the key issues is dealing with the size of the representation of V. If V is represented as an ordered list, we end up using $O(r\log(n))$ bits, which is too much for our purposes. On the other hand, V can be represented as a set, *i.e.* without storing any ordering (*e.g.* the order in which they were added to V), in which case we only require $O(r\log(n/r) + r)$ bits (we only need to store the shifts between the indices of the

[*] v.dunjko@liacs.leidenuniv.nl

[†] yimin.ge@mpq.mpg.de

[‡] ignacio.cirac@mpq.mpg.de

variables once they are ordered). This is sufficient for our purposes, however, now the process of adding one variable to V in line 6 is no longer reversible. More precisely, any reversible operation necessarily produces registers which, at least implicitly, still contain information about which element in V was just added. If these additional registers are not reused, the algorithm uses too much memory. However, in order to reuse those registers, we need to uncompute information, which may be computationally expensive. Note that since all the operations we utilize will later be quantized and called in an overarching amplitude amplification process, we must use only reversible operations – no measurements with classical feedback can be used.

To exemplify the problem, it is possible to implement the operation

$$|V_{i-1}\rangle\,|v_i\rangle\,|0\rangle \mapsto |V_{i-1}\rangle\,|v_i\rangle\,|V_{i-1}\cup\{v_i\}\rangle\,, \tag{1}$$

followed by the deletion of the first register in the state $|V_{i-1}\rangle$, which can be performed by reversing whatever circuit was used to generate it. This would realize the transformation

$$|V_{i-1}\rangle\,|v_i\rangle\,|V_{i-1}\cup\{v_i\}\rangle \mapsto |V_i\rangle\,|0\rangle\,|0\rangle\,, \tag{2}$$

where $V_i = V_{i-1}\cup\{v_i\}$. This constitutes an algorithmic deletion procedure, where we have deleted all unnecessary information so that the ancillas can be reused. However, it is easy to see that the process of calling the inverse of the entire circuit up to step $i-1$, which is (recursively) also done for each $i' < i$, has a runtime of $O^*(2^i)$. Since i goes up to r, the resulting full quantum algorithm for PBS (after using amplitude amplification) would have a runtime of $O^*(2^r \cdot 3^{r/2})$. This would be (significantly) worse than the original classical algorithm.

We circumvent this problem by using a more involved memory structure which allows for more efficient deletion. Specifically, we split V into approximately $\log_2 r$ subsets of sizes $l = 1, 2, 4, \ldots$, storing all r variable specifications in total. This memory structure is still space-efficient, in that it still requires $O(r\log(n/r) + r)$ bits. However, such a structure can be used for efficient deletion. The basic idea is as follows. At any given point, the set V_{i-1} of size $i-1$ is split into subsets corresponding to the binary representation of $i-1$. Suppose we now want to add v_i to V_{i-1}. If there is no set of size 1 yet (*i.e.*, $i-1$ is even), we simply add $\{v_i\}$ to the representation. Otherwise, we merge the two sets of size 1 into a set of size 2 and algorithmically delete the two sets of size 1 by inverting the process which computed them. Then, if there is already a subset of size 2, we merge the two subsets of size 2 into a set of size 4 and algorithmically delete the two sets of size 2, and so on.

As in the naïve algorithmic deletion procedure, the uncomputation is performed recursively. However, in the naïve case, the depth of the recursion was i, whereas in the more efficient algorithm, the recursion depth is only $O(\log i)$. This results in a computational overhead of $2^{O(\log r)}$, *i.e.* polynomial, and not exponential, in r.

In the following sections, these basic ideas are presented in more detail. The high-level descriptions of all the subroutines required to realize the elementary steps to execute the overall algorithm are provided in Table I. Each subroutine must satisfy the same two critical properties we just discussed for the highest level of the algorithm description. First, each subroutine must be economic with respect to the number of ancillary bits used. Second, since each subroutine is used many times, it is also critical that the ancillary bits are always reset (uncomputed), so they can be reused, and that the deletion is sufficiently efficient. To ensure both properties are maintained, we provide the descriptions of all subroutines in full detail.

2. Algorithm overview

The basic idea for the QBall$_1$ algorithm to calculate $V(\vec{s})$ from $\vec{s}$ consists of a main loop going through $i = 1, \ldots, r$, and for each i attempts to find another variable v_i to be added to V in the following way. The algorithm finds the first clause C_j that is not satisfied by setting all variables in V to 1 (we assume that there is some predefined order of the clauses $C_1, \ldots, C_L$). We then select the next variable v_i to be the ${s_i}^{\text{th}}$ variable in C_j that is not already in V. If no such variable exists (*i.e.*, C_j contains less than s_i variables not in V) or there exists no unsatisfied clause under the current V, then the algorithm adds a dummy variable x_{n+i} to V. Naïvely and without any further processing, this would lead to an encoding of V that simply stores the ordered list $v_1, \ldots, v_r$, taking $O(r\log n)$ bits in total. As mentioned in the main text, this would ultimately lead to a qubit requirement of the corresponding quantum algorithm that is too large to give a strong speed up. We therefore first describe how to efficiently encode a set of variables.

Let $S = \{y_1, \ldots, y_k\}$ be a set, where $0 < y_1 < \cdots < y_k \le 2n$ are integers. Define $|\mathrm{Enc}\, S\rangle$ to be a sequence of at most $O(k\log(n/k) + k)$ trits encoding $(y_1, y_2 - y_1, \ldots, y_k - y_{k-1})$, where each number is encoded on the binary subspace of the trits using no more digits than necessary and the third symbol of the trit subspace serves as a separation character between successive numbers. Note that since the numbers add up to at most n, the number of trits required for this

Main subroutines:	
$\mathsf{Calculate}_i$	$\vert s_i\rangle\,\vert\mathrm{EffEnc}\,V_{i-1}\rangle\,\vert 0\rangle \mapsto \vert s_i\rangle\,\vert\mathrm{EffEnc}\,V_{i-1}\rangle\,\vert v_i\rangle$ Given a trit s_i and (an efficient encoding of) the current set of variables, produces the next variable v_i to be added to V.
Merge_i	$\vert\vec{s}\rangle\,\vert\mathrm{EffEnc}\,V_{i-1}\rangle\,\vert v_i\rangle \mapsto \vert\vec{s}\rangle\,\vert\mathrm{EffEnc}\,V_i\rangle$ Adds the next variable to the set while maintaining efficient encoding, and uncomputes the value of the added variable.
Key ancillary subroutines:	
$\mathsf{Extract}_k$	$\vert\mathrm{Enc}\,S\rangle\,\vert j\rangle\,\vert 0\rangle \mapsto \vert\mathrm{Enc}\,S\rangle\,\vert j\rangle\,\vert y_j\rangle$ Extracts the j^{th} element (w.r.t. ascending ordering) from the encoding of the set S of known size k. Critical subroutine in $\mathsf{Calculate}_i$.
Shift_k	$\vert\mathrm{Enc}\,S\rangle\,\vert j\rangle\,\vert 0\rangle \mapsto \vert\mathrm{Enc}\,S\rangle\,\vert j\rangle\,\vert y_j - y_{j-1}\rangle$ Used in $\mathsf{Extract}_k$. The set-encoding stores shifts between neighbouring elements, and this subroutine extracts them. In essence, it counts special separation characters delineating different numbers in the set encoding.
$\mathsf{Contains}_k$	$\vert\mathrm{Enc}\,S\rangle\,\vert v\rangle\,\vert 0\rangle \mapsto \vert\mathrm{Enc}\,S\rangle\,\vert v\rangle\,\vert v \in S?\rangle$ Checks if v is present in the set S, given encoding of S. Required for checking if a clause is satisfied when flipping the values of literals if they occur in S, and also when selecting the next variable, as we can only select those which have not been chosen previously. Uses $\mathsf{Extract}_k$.
$\mathsf{Check}_{j,i}$	$\vert\mathrm{EffEnc}\,V_{i-1}\rangle\,\vert 0\rangle \mapsto \vert\mathrm{EffEnc}\,V_{i-1}\rangle\,\vert C_j(V_{i-1})\rangle$ Checks if the j^{th} clause is satisfied by $\mathbf{x}(V_{i-1})$. One of the key subroutines.
Select_i	$\vert\mathrm{EffEnc}\,V_{i-1}\rangle\,\vert s_i\rangle\,\vert L+1-j\rangle\,\vert 0\rangle \mapsto \vert\mathrm{EffEnc}\,V_{i-1}\rangle\,\vert s_i\rangle\,\vert L+1-j\rangle\,\vert v_{j,i}(V_{i-1},s_i)\rangle$ Once a clause C_j is identified to not be satisfied, this subroutine uses the choice s_i and V_{i-1} to select a variable in C_j to be added next.
Append_k	$\vert v\rangle\,\vert\mathrm{Enc}\,S\rangle\,\vert 0\rangle \mapsto \vert v\rangle\,\vert\mathrm{Enc}\,S \cup \{v\}\rangle$ Adds an element to the set S and computes the encoding of the new set. The added element is assumed to be larger than the largest element in S.
Union_{k_1,k_2}	$\vert\mathrm{Enc}\,S_1\rangle\,\vert\mathrm{Enc}\,S_2\rangle\,\vert 0\rangle \mapsto \vert\mathrm{Enc}\,S_1\rangle\,\vert\mathrm{Enc}\,S_2\rangle\,\vert\mathrm{Enc}\,S_1 \cup S_2\rangle$ Generates an efficient encoding of the union of two efficiently encoded sets. In essence, uses $\mathsf{Extract}_k$ on both sets to compare elements, storing the smaller, and keeping a position counter for both. Uses Append_k.

TABLE I. Summary of subroutines for the space-efficient reversible implementation to solve PBS. The two main subroutines ran in succession find the next variable, add it to the set, and, critically, free the ancillary registers so they can be reused.

encoding is indeed at most

$$\begin{aligned}\lceil\log_2 y_1\rceil + \lceil\log_2(y_2-y_1)\rceil + \cdots + \lceil\log_2(y_k-y_{k-1})\rceil + 2k &\le \log_2 y_1 + \log_2(y_2-y_1) + \cdots + \log_2(y_k-y_{k-1}) + 3k \quad (3)\\ &\le k\log_2\frac{y_k}{k} + 3k \quad (4)\\ &\le k\log_2\frac{n}{k} + 4k, \quad (5)\end{aligned}$$

where the second line follows from concavity of the logarithm. Note that this is significantly less than the naïve encoding of S which uses $O(k\log n)$ bits. For concreteness, we will assume that after the last separation character, the remaining trits are in 0.

This gives us an efficient way of storing a set of at most r elements. However, encoding V as $|\mathrm{Enc}\,V\rangle$ at all stages of the algorithm is problematic, because reversibility would be lost (indeed, the map $|\mathrm{Enc}\,V_{i-1}\rangle \mapsto |\mathrm{Enc}\,V_i\rangle$ is not injective). We therefore store the set of variables during the algorithm as follows. Let $V_i = \{v_1,\dots,v_i\}$ be the set of variables obtained after the i^{th} round. Note that $|V_i| = i$. We now divide V_i into subsets $V_{i,1},\dots,V_{i,m_i}$ of sizes which are powers of two corresponding to the binary expansion of i. For example, if $i = 13$ (binary expansion 1101), we have $V_{i,1} = \{v_1,v_2,v_3,v_4,v_5,v_6,v_7,v_8\}$, $V_{i,2} = \{v_9,v_{10},v_{11},v_{12}\}$ and $V_{i,3} = \{v_{13}\}$.

To efficiently store V_i, we efficiently encode the set as $|\mathrm{EffEnc}\,V_i\rangle := |\mathrm{Enc}\,V_{i,1}\rangle \dots |\mathrm{Enc}\,V_{i,m_i}\rangle$. Note that the number of bits required for $|\mathrm{EffEnc}\, V_i\rangle$ is at most

$$\sum_{l=0}^{\lceil \log_2 r \rceil} \left(2^l \log_2 \frac{n}{2^l} + 2^{l+2}\right) = \sum_{l=0}^{\lceil \log_2 r \rceil} 2^l \log_2 \frac{n}{r} + \sum_{l=0}^{\lceil \log_2 r \rceil} 2^{l+2} + \sum_{l=0}^{\lceil \log_2 r \rceil} 2^l \log_2 \frac{r}{2^l} \tag{6}$$

$$\leq 4r \log_2 \frac{n}{r} + 16r + 2r \sum_{l=0}^{\lceil \log_2 r \rceil} \frac{1}{2^{\lceil \log_2 r \rceil - l}} (\lceil \log_2 r \rceil - l) \tag{7}$$

$$\leq 4r \log_2 \frac{n}{r} + 20r \tag{8}$$

$$= O(r \log(n/r) + r), \tag{9}$$

where in (7), the first two terms follow from $\sum_{l=0}^{\lceil \log_2 r \rceil} 2^l \leq 4r$ and the last term follows from $2^{\lceil \log_2 r \rceil} \geq r > 2^{\lceil \log_2 r \rceil - 1}$, and (8) follows from

$$\sum_{l=0}^{\lceil \log_2 r \rceil - l} \frac{1}{2^{\lceil \log_2 r \rceil}} (\lceil \log_2 r \rceil - l) < \sum_{j=0}^{\infty} \frac{j}{2^j} = 2. \tag{10}$$

For a given $i \in \{1, \ldots, r\}$, the circuit is divided into a calculate phase and a merge phase. The calculate phase performs

$$\mathsf{Calculate}_i \, |s_i\rangle \, |\mathrm{EffEnc}\, V_{i-1}\rangle \, |0\rangle = |s_i\rangle \, |\mathrm{EffEnc}\, V_{i-1}\rangle \, |v_i\rangle \,, \tag{11}$$

i.e., finds the next variable to be added to V. The merge phase performs

$$\mathsf{Merge}_i \, |\vec{s}\rangle \, |\mathrm{EffEnc}\, V_{i-1}\rangle \, |v_i\rangle = |\vec{s}\rangle \, |\mathrm{EffEnc}\, V_i\rangle \,, \tag{12}$$

i.e., converts the efficient encoding of the old set into the efficient encoding of the new set. It achieves this by using a series of set merge operations comprising calculating the representation of the set-union and the uncomputation of original the set representations.

While the main ideas of both phases are straightforward, the primary challenge is to develop these algorithms in the form of circuits that are both reversible and at the same time require few ancillas. For the latter, it is imperative that at the end of any subroutine, all ancillas, which can be assumed to be initially in the 0 state, are returned to 0 so that they can be reused. Note that not achieving this would increase the ancilla requirement for each call of any such subroutine and would likely result in the algorithm requiring too many ancillas. For convenience, we therefore introduce the following notion.

Definition 1. Let $f : \{0,1\}^q \to \{0,1\}^q$ be a bijection. We say that f can be *implemented reversibly using l ancillas and g gates* if there exists a (classical) reversible circuit of at most g elementary gates which implements the map

$$|x\rangle \, |0\rangle^{\otimes l} \mapsto |f(x)\rangle \, |0\rangle^{\otimes l} \,. \tag{13}$$

Proposition 1.

(i) $\mathsf{Calculate}_i$ can be implemented reversibly using $O(\log n)$ ancillas and $O(\mathrm{poly}(n))$ gates.

(ii) Merge_i can be implemented reversibly using $O(r \log(n/r) + r + \log n)$ ancillas and $O(\mathrm{poly}(n))$ gates.

In the next two subsections, we prove Proposition 1 by describing each phase in detail. Table I gives an overview of the key subroutines involved.

3. The calculate phase

In this part of the algorithm for a given $i \in \{1, \ldots, r\}$, we calculate the ${s_i}^{\text{th}}$ variable in the first unsatisfied clause. This part of the algorithm has two components. First, finding the first clause that is not satisfied, and second, selecting the corresponding variable.

On input, we have $|s_i\rangle \, |\mathrm{EffEnc}\, V_{i-1}\rangle$. Note that the sizes of the sets are known in advance. We first show how given a set, we can efficiently and reversibly extract its elements using few bits. For a set $S = \{y_1, \ldots, y_k\}$ of integers $0 < y_1 < \cdots < y_k \leq 2n$ of known size k, define the reversible operation $\mathsf{Extract}_k \, |\mathrm{Enc}\, S\rangle \, |j\rangle \, |0\rangle = |\mathrm{Enc}\, S\rangle \, |j\rangle \, |y_j\rangle$ for $j \in \{1, \ldots, k\}$. We will assume that any set S we consider has at most $r \leq n$ elements.

Lemma 1. *$\mathsf{Extract}_k$ can be implemented reversibly using $O(\log n)$ ancilla bits and $O(\mathrm{poly}(n))$ gates.*

Proof. We first show how to implement the operation $\mathsf{Shift}_k \left|\mathrm{Enc}\, S\right\rangle \left|j\right\rangle \left|0\right\rangle = \left|\mathrm{Enc}\, S\right\rangle \left|j\right\rangle \left|y_j - y_{j-1}\right\rangle$, where by convention $y_0 = 0$. This operation comprises three parts: first compute the position of the $(j-1)^{\mathrm{st}}$ separation character in $\left|\mathrm{Enc}\, S\right\rangle$ and the number of trits to the j^{th} separation character (we call this operation $U_{1,k}$), second copy the relevant trits to a separate register (we call this operation $U_{2,k}$), and third uncompute all other ancillas by running $U_{1,k}^{-1}$.

We introduce three counters, called the block position counter, the length counter and the j-counter, going from 0 to $\lceil k \log_2(n/k)\rceil + 4k$, 0 to $\lceil \log_2 n\rceil + 1$, and 0 to k respectively, and all initially set to 0. Note that all three counters use no more than $O(\log n)$ bits. To implement $U_{1,k}$, we do the following steps.

First, we add k to the j-counter to set it to k. Then, for each trit in $\left|\mathrm{Enc}\, S\right\rangle$ sequentially, we do the following: Controlled on the j-counter being larger than $k - j + 1$, increase the block position counter by one; then controlled on the trit being a termination symbol, decrease j-counter by one. This ends with the j-counter being 0, and the block position counter containing the position of the j^{th} block.

Note, the implementation of operations controlled on the state of counters can be done by a number comparison, and the latter can be implemented reversibly with one ancilla bit.

Next, for each trit in $\left|\mathrm{Enc}\, S\right\rangle$, we sequentially do the following: If the j-counter is equal to $j-1$, increase the lenght counter by one; if the trit is a termination symbol, increase the j-counter register by one.

Since the length counter is only increased in the j^{th} block, it stores its length. The j-counter now has the (known) value k and we can reset it to 0.

This implements $U_{1,k}$, which performs

$$U_{1,k} \left|\mathrm{Enc}\, S\right\rangle \left|j\right\rangle \left|0\right\rangle \left|0\right\rangle = \left|\mathrm{Enc}\, S\right\rangle \left|j\right\rangle \left|\mathrm{pos}(S,j)\right\rangle \left|\mathrm{len}(S,j)\right\rangle \tag{14}$$

and resets all ancillas to 0, where $\mathrm{pos}(S,j)$ is the position of the $(j-1)^{\mathrm{st}}$ separation character and $\mathrm{len}(S,j) = \mathrm{pos}(S,j+1) - \mathrm{pos}(S,j)$ is the number of trits to the j^{th} separation character.

To implement $U_{2,k}$, it will be convenient to define the family of $U(j,l)$ copy-unitaries. Each such unitary is a sequence of l CNOT gates, which copy the trits from position $j+1$ to position $j+l$ as the last l bits of the shift-output register. Each such unitary costs only l CNOTS. The $c - U(j,l)$ is the controlled family of such unitaries, which is controlled on the states of the block position and the length counter, and the corresponding $U(j,l)$ is only activated if the block position counter and the length counter are equal to j and l, respectively. For completeness, these are only well-defined if $j + l \leq \lceil k \times \log_2(n/k)\rceil + 4k$, *i.e.*, the lenght of the register containing $\left|\mathrm{Enc}\, S\right\rangle$. If the labels are out-of-bounds, we substitute them with identities. Although they copy trits, they will only be used as to act on the binary subspace, as we will not be copying the termination symbols. $U_{2,k}$ is then given by

$$U_{2,k} = \prod_{j=1}^{\lceil \log_2(n/k)\rceil + 4k} \prod_{l=1}^{\lceil \log_2 n\rceil + 1} c - U(j,l). \tag{15}$$

Note that only one $U(j,l)$ will be activated, namely the one specified by the position and length registers. In total, $U_{2,k}$ contains $O((\log(n/k) + k)\log(n)) = O(\log(n)^2)$ gates, and uses no ancillas. Applying $U_{1,k}^{-1}$ then completes the implementation of Shift_k.

Finally, $\mathsf{Extract}_k$ can be implemented by introducing an additional counter from 1 to k, and k calls to controlled-Shift_k, each controlled on the counter being $\leq j$, followed by incrementing that counter. This way, each call adds $y_l - y_{l-1}$ to the output register if $l \leq j$. This finally leaves the output register in $\left|y_j\right\rangle$ and all ancillas in 0, as desired. □

This extraction subroutine can be used to check if a given variable index $v \in \{1, \ldots, 2n\}$ is contained in a set of known size. To this end, define the (reversible) operation $\mathsf{Contains}_k$ which performs $\mathsf{Contains}_k \left|\mathrm{Enc}\, S\right\rangle \left|v\right\rangle \left|0\right\rangle = \left|\mathrm{Enc}\, S\right\rangle \left|v\right\rangle \left|v \in S?\right\rangle$, where the last bit is 1 if $v \in S$ and 0 otherwise.

Lemma 2. *$\mathsf{Contains}_k$ can be implemented reversibly using $O(\log n)$ ancilla bits and $O(\mathrm{poly}(n))$ gates.*

Proof. We introduce a counter from 0 to k and $O(\log n)$ additional ancilla bits. Then, for each $j = 1, \ldots, k$, we use Lemma 1 to extract the j^{th} element, controlled on the counter being in 0. We then check whether the extracted element is equal to v, and store the result in a separate temporary result bit. We then uncompute the controlled extraction and, controlled on the result bit being in 1, increment the counter. After doing this for all $j = 1, \ldots, k$, the result bit is in 1 if the element is in S and 0 if not. We apply a CNOT of the temporary result bit to the output bit and then run the inverse of the entire circuit up to that point to reset the ancillas. □

Next, for a clause C_j, let

$$C_j(V) = \begin{cases} 1 & C_j \text{ satisfied by } \mathbf{x}(V) \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

We furthermore define a subroutine $\mathsf{Check}_{j,i}$ which performs

$$\mathsf{Check}_{j,i} \left|\mathrm{EffEnc}\, V_{i-1}\right\rangle |0\rangle = \left|\mathrm{EffEnc}\, V_{i-1}\right\rangle |C_j(V_{i-1})\rangle\,. \tag{17}$$

Note the dependence of $\mathsf{Check}_{j,i}$ on i reflects that the routine is adjusted slightly for different set sizes appearing in $|\mathrm{EffEnc}\, V_i\rangle$ (which are known in advance).

Lemma 3. *$\mathsf{Check}_{j,i}$ can be implemented reversibly using $O(\log n)$ ancilla bits and $O(\mathrm{poly}(n))$ gates.*

Proof. We introduce three additional ancilla bits storing the values of the three variables in the clause. Then, for each variable and each set in $|\mathrm{EffEnc}\, V_{i-1}\rangle$, we use Lemma 2 to check if the variable is in the set. The result is copied onto the variable bit using a CNOT, and the $\mathsf{Contains}_k$ operation is then reversed. Note that the sets can be assumed to be disjoint. Then, the variable bits are flipped according to the literals in the clause. The clause is then evaluated and the result stored in a separate bit. We then apply the inverse of the circuit to reset the ancillas. □

Next, we show how to select a variable from a given clause. For a clause C_j with variables x_a, x_b, x_c with $0 < a < b < c \leq n$, let $v_{j,i}(V_{i-1}, s_i)$ be the $s_i{}^{\text{th}}$ smallest number in $\{a,b,c\}\backslash V_{i-1}$. If $s_i > |\{a,b,c\}\backslash V_{i-1}|$, then $v_{j,i}(V_{i-1}, s_i) = n+i$. We now define the reversible operation Select_i which performs

$$\mathsf{Select}_i \left|\mathrm{EffEnc}\, V_{i-1}\right\rangle |s_i\rangle |L+1-j\rangle |0\rangle = \left|\mathrm{EffEnc}\, V_{i-1}\right\rangle |s_i\rangle |L+1-j\rangle |v_{j,i}(V_{i-1}, s_i)\rangle \tag{18}$$

for $j = 1, \ldots, L$ and

$$\mathsf{Select}_i \left|\mathrm{EffEnc}\, V_{i-1}\right\rangle |s_i\rangle |0\rangle |0\rangle = \left|\mathrm{EffEnc}\, V_{i-1}\right\rangle |s_i\rangle |0\rangle |n+i\rangle\,. \tag{19}$$

Lemma 4. *Select_i can be implemented reversibly using $O(\log n)$ ancilla bits and $O(\mathrm{poly}(n))$ gates.*

Proof. First, we add $n+i$ to the result register controlled on the counter input register being in 0. Then, for each j, we do the following operations controlled on the counter input register being in $L+1-j$: Use Lemma 2 to check which variables are contained in V_{i-1} and store the result in three variable ancilla bits (similarly to the implementation $\mathsf{Check}_{j,i}$ in Lemma 3). Then, for each combination of values of the variable ancilla bits and s_i, add the value of $v_{j,i}$ to the result register controlled on the values of the variable register and s_i. We then apply the inverse of the controlled-$\mathsf{Contains}_k$ operations to uncompute the variable ancilla bits. □

To implement $\mathsf{Calculate}_i$, we first define $G_{j,i}$ to be the following circuit: On input we have $|\mathrm{EffEnc}\, V_{i-1}\rangle$, a counter from 0 to L using $O(\log L)$ bits, a result bit and $O(\log n)$ workspace ancillas. $G_{j,i}$ first performs a controlled-$\mathsf{Check}_{j,i}$, controlled on the counter being in 0, on the register containing $|\mathrm{EffEnc}\, V_{i-1}\rangle$ and $O(\log n)$ of the workspace ancillas. Then, we perform a CNOT onto the result bit, controlled on the ancilla containing $C_j(V_{i-1})$ being 0. We then run controlled-$\mathsf{Check}_{j,i}^{-1}$, controlled on the counter being in 0, to uncompute the workspace ancillas. Then, we add 1 to the counter, controlled on the result bit being 1.

The implementation of $\mathsf{Calculate}_i$ is now as follows.

1. Run $G_{1,i} \ldots G_{L,i} \left|\mathrm{EffEnc}\, V_{i-1}\right\rangle |0\rangle |0\rangle$. It is easy to see that this results in $|\mathrm{EffEnc}\, V_{i-1}\rangle |L+1-j_{\min}\rangle |1 - F(\mathbf{x}(V_{i-1}))\rangle$, where $j_{\min}$ is the smallest j such that C_j is unsatisfied under $\mathbf{x}(V_{i-1})$ if such a clause exist, and $L+1$ otherwise, and $F(\mathbf{x}(V))$ is 1 if $\mathbf{x}(V)$ is a satisfying assignment for F and 0 otherwise.

2. Run Select_i on $|\mathrm{EffEnc}\, V_{i-1}\rangle |s_i\rangle |L+1-j_{\min}\rangle$ and $O(\log n)$ workspace ancillas. This produces $|\mathrm{EffEnc}\, V_{i-1}\rangle |s_i\rangle |L+1-j_{\min}\rangle |v_i\rangle$, where v_i is the index of the variable to be added to V.

3. Apply the inverse of step 1. This results in $|\mathrm{EffEnc}\, V_{i-1}\rangle |s_i\rangle |v_i\rangle$ and all other ancilla bits being reset to 0, as desired.

This proves Proposition 1(i). □

4. The merge phase

The main tool of the merge phase is a reversible operation Union_{k_1,k_2} that calculates the union of two sets S_1, S_2 of (known) sizes $k_1, k_2 \leq r$,

$$\mathsf{Union}_{k_1,k_2} \left|\mathrm{Enc}\, S_1\right\rangle \left|\mathrm{Enc}\, S_2\right\rangle \left|0\right\rangle = \left|\mathrm{Enc}\, S_1\right\rangle \left|\mathrm{Enc}\, S_2\right\rangle \left|\mathrm{Enc}\, S_1 \cup S_2\right\rangle . \tag{20}$$

Lemma 5. *Union_{k_1,k_2} can be implemented reversibly with $O(K \log(n/K) + K + \log n)$ ancilla bits and $O(\mathrm{poly}(n))$ gates, where $K = k_1 + k_2$.*

The basic idea is to simply extract the ${j_1}^{\mathrm{th}}$ and ${j_2}^{\mathrm{th}}$ elements of S_1, S_2 at a time, where j_1 and j_2 are the current values of two counter registers, appends the smaller of those elements to the output set, and increase either j_1 or j_2 depending on which one was added. This results in all elements in $S_1 \cup S_2$ being added to the output set in increasing order. As such, we first show how to efficiently append an element to a set of known size.

Lemma 6. *Let $S = \{v_1, \ldots, v_k\}$ be a set with $0 < v_1 < \cdots < v_k < 2n$ and let $v \in \{v_k + 1, \ldots, 2n\}$. Then the operation*

$$\mathsf{Append}_k \left|v\right\rangle \left|\mathrm{Enc}\, S\right\rangle \left|0\right\rangle = \left|v\right\rangle \left|\mathrm{Enc}\, S \cup \{v\}\right\rangle \tag{21}$$

can be implemented reversibly using $O(\log n)$ ancilla bits and trits and $O(\mathrm{poly}(n))$ gates.

Proof. The proof is very similar to the proof of Lemma 1, the steps are as follows.

1. Use $\mathsf{Extract}_k$ to extract the value of the largest element $v_k \in S$ onto a separate register (note that k is known).
2. Calculate the difference of v with that value and store that difference in a separate register of $O(\log n)$ bits which we call the difference register. The ancilla workspace now contains $\left|v_k\right\rangle \left|v - v_k\right\rangle$.
3. Introduce a counter from 0 to $O(\log \log n)$ which we call the length counter, and an additional control bit initially in 0, and use them to find the number of relevant binary digits in the difference register (*i.e.*, $\lceil \log_2(v - v_k) \rceil + 1$) as follows. Starting from the most significant digit of the difference register, in turn do the following for each bit in the difference register: first, controlled on the bit being 1 and end the length counter being 0, flip the control bit. Then, controlled on the control bit being 1, add 1 to the length counter. After doing this for all bits in the difference register, the length counter will be in $\lfloor \log_2(v - v_k) \rfloor + 1$ and the control bit in 1 (since we assume $v > v_k$). Flip the control bit to reset it. The ancilla workspace now contains $\left|v_k\right\rangle \left|v - v_k\right\rangle \left|\lfloor \log_2(v - v_k) \rfloor + 1\right\rangle$.
4. Use the operation $U_{1,k}$ defined in the proof of Lemma 1 to find the position of the last (*i.e.*, k^{th}) separation character in $\left|\mathrm{Enc}\, S\right\rangle$. The ancilla workspace now contains $\left|v_k\right\rangle \left|v - v_k\right\rangle \left|\lfloor \log_2(v - v_k) \rfloor + 1\right\rangle \left|\mathrm{pos}(S, k+1)\right\rangle$.
5. Define a family of unitaries $V(j, l)$ which is a sequence of l CNOTs copying the first l bits in the difference register to the binary subspace of the $(j+1), \ldots, (j+l)^{\mathrm{th}}$ trits of the set register, followed by flipping the $(j+l+1)^{\mathrm{st}}$ trit in the set register into a separation character (note that we assume that all these trits are initially in 0). This operation is only well-defined for $j + l < \lceil (k+1) \log_2(n/(k+1)) + 4(k+1) \rceil$. For out of bounds values of j, l, we define $V(j, l)$ to be the identity. Define $c - V(j, l)$ to be $V(j, l)$ controlled on the append position counter being j and the length counter being l. Apply

$$\prod_{j=1}^{\lceil (k+1) \log_2(n/(k+1)) + 2(k+1) \rceil} \prod_{l=1}^{\lceil \log_2 n \rceil} c - V(j, l). \tag{22}$$

   This appends the value of $v - v_k$ (without leading zeros) and a separation character after the last separation character in the set register.
6. To reset the ancilla workspace, first apply the inverse of step 4, but replacing $U_{1,k}^{-1}$ with $U_{1,k+1}^{-1}$. This resets the ancilla register containing $\left|\mathrm{pos}(S, k+1)\right\rangle$. Then apply the inverse of step 3 and step 2. This resets the ancilla registers containing $\left|\lfloor \log_2(v - v_k) \rfloor + 1\right\rangle$ and $\left|v - v_k\right\rangle$, respectively. Finally apply the inverse of step 1, but replacing $\mathsf{Extract}_k^{-1}$ with $\mathsf{Extract}_{k+1}^{-1}$. This resets the ancilla register containing v_k and thus all remaining ancillas.

Note that in the last step, the changes from $U_{1,k}^{-1}$ to $U_{1,k+1}^{-1}$ and $\mathsf{Extract}_k^{-1}$ to $\mathsf{Extract}_{k+1}^{-1}$, respectively, are necessary because after step 5, the set register contains a set of size $k+1$ and not k. □

Proof of Lemma 5. We introduce the following ancilla registers: two counters from 1 to k_1 and k_2, respectively, called the first and second j-counter, respectively, two registers of $O(\log n)$ bits called the first and second candidate registers, respectively, K trits called comparison trits and $O(K\log(n/K)+K)$ bits to temporarily store $|\mathrm{Enc}\, S_1 \cup S_2\rangle$.

As explained above, the basic idea is to simply extract the elements of S_1, S_2 corresponding to the current values of the j-counters, then add the smaller of those element to the output set, and increase the corresponding j-counter. Additional care however has to be taken to ensure that the algorithm still runs correctly when all elements of one of the sets have been added. As such, the algorithm is as follows. Sequentially, for each $j = 0, \dots, K-1$, do the following:

1. For $b = 1, 2$ in turn, run controlled-$\mathsf{Extract}_{k_b}$ on $|\mathrm{Enc}\, S_b\rangle$, the b^{th} j-counter and the b^{th} candidate register, where the operation is controlled on the $(3-b)^{\text{th}}$ j-counter not being in $j - k_1$. Note that since the sum of the two j-counters is always equal j at all stages, the latter condition is equivalent to all elements of S_b having already been added.

2. For $b = 1, 2$ in turn, add $2n+1$ to the b^{th} candidate register, controlled on the $(3-b)^{\text{th}}$ j-counter being in $j - k_1$. This ensures that when all elements of one of the sets have already been added, the value in the candidate register corresponding to the other set is always smaller.

3. We compare the values of the two candidate registers and determine the smaller one. To do that, note that it is easy to efficiently and reversibly implement the minimum finding operation $\mathsf{Min}\, |v_1\rangle\, |v_2\rangle\, |0\rangle = |v_1\rangle\, |v_2\rangle\, |m(v_1, v_2)\rangle$, where
$$m(v_1, v_2) = \begin{cases} 1 & v_1 < v_2 \\ 2 & v_2 < v_1 \\ 0 & v_1 = v_2, \end{cases} \tag{23}$$
using only $O(\log\log n))$ ancilla bits. We use the j^{th} comparison trit as the third register when calling Min.

4. For $b = 1, 2$ in turn, we call controlled-Append_j on the b^{th} candidate register and the temporary set register, controlled on the j^{th} comparison trit being b.

5. We apply the inverse of the operations in steps 1-2. This resets both candidate registers to 0.

6. For $b = 1, 2$ in turn, we add 1 to the b^{th} j-counter controlled on the j^{th} comparison trit being in b.

After doing this for $j = 0, \dots, K-1$, we copy the state of the temporary set register onto the output register and then apply the inverse of the entire circuit up to that point. This resets the comparison trits and both j-counters. □

We now describe how to implement Merge_i.

The first step is to convert $|v_i\rangle$ into $|\mathrm{Enc}\{v_i\}\rangle$ using Append_0. We then apply a sequence of *merge operations* of sets that follows the pattern of a binary addition: suppose we expand $i-1$ in binary and add 1 to it using the standard addition procedure. Then, every carry bit corresponds to merging two sets. More formally, let $g \geq 0$ be the largest integer such that 2^g divides i (g specifies the the first non-zero position in the binary expansion of i from the least significant bit position). Then, the sequence of merge operations is as follows. For $l = 0, \dots, g-1$, we merge the sets $\{v_{i-2^{l+1}+1}, \dots, v_{i-2^l}\}$ and $\{v_{i-2^l+1}, \dots, v_i\}$.

Each merge of two sets $S_1 = \{v_{i-2^{l+1}+1}, \dots, v_{i-2^l}\}$, $S_2 = \{v_{i-2^l+1}, \dots, v_i\}$ has two parts. First, we use Lemma 5 to compute $|\mathrm{Enc}\, S_1 \cup S_2\rangle$. Second, we uncompute $|S_1\rangle\, |S_2\rangle$ by running the inverse of the circuit from the end of the merge phase for $i - 2^{l+1}$. Note that since $S_{1,2}$ always contain successive variables up to v_i, this is possible.

We illustrate this procedure for $i = 20$. Note that the binary expansion of $i-1$ is $19 = 16+2+1$, so $V_{19,1} = \{v_1, \dots, v_{16}\}$, $V_{19,2} = \{v_{17}, v_{18}\}$, $V_{19,3} = \{v_{19}\}$. Our aim is to go from
$$|\vec{s}\rangle\, |\mathrm{EffEnc}\, V_{19}\rangle\, |\mathrm{Enc}\{v_{20}\}\rangle = |\vec{s}\rangle\, |\mathrm{Enc}\{v_1, \dots, v_{16}\}\rangle\, |\mathrm{Enc}\{v_{17}, v_{18}\}\rangle\, |\mathrm{Enc}\{v_{19}\}\rangle\, |\mathrm{Enc}\{v_{20}\}\rangle \tag{24}$$
to
$$|\vec{s}\rangle\, |\mathrm{Enc}\{v_1, \dots, v_{16}\}\rangle\, |\mathrm{Enc}\{v_{17}, v_{18}, v_{19}, v_{20}\}\rangle = |\vec{s}\rangle\, |\mathrm{EffEnc}\, V_{20}\rangle\,. \tag{25}$$
Note that reversibility is preserved since the operation will also involve the $|\vec{s}\rangle$ register. First, we call $\mathsf{Union}_{1,1}$ to compute
$$\mathsf{Union}_{1,1}\, |\mathrm{Enc}\{v_{19}\}\rangle\, |\mathrm{Enc}\{v_{20}\}\rangle\, |0\rangle = |\mathrm{Enc}\{v_{19}\}\rangle\, |\mathrm{Enc}\{v_{20}\}\rangle\, |\mathrm{Enc}\{v_{19}, v_{20}\}\rangle\,. \tag{26}$$

We then uncompute $|\mathrm{Enc}\{v_{19}\}\rangle\,|\mathrm{Enc}\{v_{20}\}\rangle$ by running the inverse of the part of the circuit from the end of merge phase of $i = 18$. Indeed, that part of the computation mapped $|\vec{s}\rangle\,|\mathrm{Enc}\{v_1,\dots,v_{16}\}\rangle\,|\mathrm{Enc}\{v_{17},v_{18}\}\rangle\,|0\rangle\,|0\rangle$ to $|\vec{s}\rangle\,|\mathrm{Enc}\{v_1,\dots,v_{16}\}\rangle\,|\mathrm{Enc}\{v_{17},v_{18}\}\rangle\,|\mathrm{Enc}\{v_{19}\}\rangle\,|\mathrm{Enc}\{v_{20}\}\rangle$. Thus, running the inverse of this part of the circuits results in $|\vec{s}\rangle\,|\mathrm{Enc}\{v_1,\dots,v_{16}\}\rangle\,|\mathrm{Enc}\{v_{17},v_{18}\}\rangle\,|\mathrm{Enc}\{v_{19},v_{20}\}\rangle$. Next, we call $\mathsf{Union}_{2,2}$, which produces

$$\mathsf{Union}_{2,2}\,|\mathrm{Enc}\{v_{17},v_{18}\}\rangle\,|\mathrm{Enc}\{v_{19},v_{20}\}\rangle\,|0\rangle = |\mathrm{Enc}\{v_{17},v_{18}\}\rangle\,|\mathrm{Enc}\{v_{19},v_{20}\}\rangle\,|\mathrm{Enc}\{v_{17},v_{18},v_{19},v_{20}\}\rangle\,. \tag{27}$$

We then uncompute $|\mathrm{Enc}\{v_{17},v_{18}\}\rangle\,|\mathrm{Enc}\{v_{19},v_{20}\}\rangle$ by running the inverse of the part of the circuit from the end of the merge phase for $i = 16$. This results in

$$|\vec{s}\rangle\,|\mathrm{Enc}\{v_1,\dots,v_{16}\}\rangle\,|\mathrm{Enc}\{v_{17},v_{18},v_{19},v_{20}\}\rangle\,, \tag{28}$$

which is the desired result.

What remains to be seen is the runtime scaling of Merge_i. We show now that the runtime of each merge operation (comprising calculating the union of two sets and uncomputing these sets) scales polynomially with the number of elements involved. To see this, we first show that the runtime M_l of the operation mapping $|\vec{s}\rangle\,|\mathrm{EffEnc}\,V_{i-2^l}\rangle\,|0\rangle$ to $|\vec{s}\rangle\,|\mathrm{EffEnc}\,V_{i-2^l}\rangle\,|\mathrm{Enc}\{v_{i-2^l+1},\dots,v_i\}\rangle$ for any i is bounded by $M_l = O^*(4^l)$, where we assume that $|\mathrm{EffEnc}\,V_{i-2^l}\rangle$ has no sets of 2^l elements or less. This can be seen by induction. The claim is trivial for $l = 0$. Suppose now that this is true for any i and any $l' < l$. Then, the operation that maps $|\vec{s}\rangle\,|\mathrm{EffEnc}\,V_{i-2^{l+1}}\rangle\,|0\rangle$ to $|\vec{s}\rangle\,|\mathrm{EffEnc}\,V_{i-2^{l+1}}\rangle\,|\mathrm{Enc}\{v_{i-2^{l+1}+1},\dots,v_i\}\rangle$ comprises two parts. First, we map $|\vec{s}\rangle\,|\mathrm{EffEnc}\,V_{i-2^{l+1}}\rangle\,|0\rangle$ to $|\vec{s}\rangle\,|\mathrm{EffEnc}\,V_{i-2^{l+1}}\rangle\,|\mathrm{Enc}\{v_{i-2^{l+1}+1},\dots,v_{i-2^l}\}\rangle = |\vec{s}\rangle\,|\mathrm{EffEnc}\,V_{i-2^l}\rangle$. This operation takes runtime M_l. Next, we map $|\vec{s}\rangle\,|\mathrm{EffEnc}\,V_{i-2^l}\rangle\,|0\rangle$ to $|\vec{s}\rangle\,|\mathrm{EffEnc}\,V_{i-2^l}\rangle\,|\mathrm{Enc}\{v_{i-2^l+1},\dots,v_i\}\rangle$, which also takes runtime M_l. Next, we call $\mathsf{Union}_{2^l,2^l}$ to compute

$$\begin{aligned}\mathsf{Union}_{2^l,2^l}\,&|\mathrm{Enc}\{v_{i-2^{l+1}+1},\dots,v_{i-2^l}\}\rangle\,|\mathrm{Enc}\{v_{i-2^l+1},\dots,v_i\}\rangle\,|0\rangle = \\ &|\mathrm{Enc}\{v_{i-2^{l+1}+1},\dots,v_{i-2^l}\}\rangle\,|\mathrm{Enc}\{v_{i-2^l+1},\dots,v_i\}\rangle\,|\mathrm{Enc}\{v_{i-2^{l+1}+1},\dots,v_i\}\rangle\,.\end{aligned} \tag{29}$$

The runtime of this call can be bounded by a polynomial $p(n)$ that is independent of l or i. Finally, to uncompute $|\mathrm{Enc}\{v_{i-2^{l+1}+1},\dots,v_{i-2^l}\}\rangle\,|\mathrm{Enc}\{v_{i-2^l+1},\dots,v_i\}\rangle$, we apply the inverse of the two operations before the call to the union, each taking runtime M_l. This implies $M_{l+1} \le 4M_l + p(n)$, which clearly gives $M_l = O^*(4^l)$, as claimed. In particular, this implies that merging $\{v_{i-2^{l+1}+1},\dots,v_{i-2^l}\}$ and $\{v_{i-2^l+1},\dots,v_i\}$, comprising of first calculating their union and then uncomputing the original sets, takes runtime at most $O^*(4^l)$.

Since $i \le r$, we need to do this operation at most once for each $l \in \{0,\dots,\lceil\log_2 r\rceil\}$. Thus, the runtime of Merge_i is at most

$$O^*\left(\sum_{l=0}^{\lceil\log_2 r\rceil} 4^l\right) = O^*(\mathrm{poly}(r)) = O(\mathrm{poly}(n)). \tag{30}$$

This proves Proposition 1(ii). □

5. *Quantum algorithm for PBS*

To summarize, we have proven the following

Proposition 2 (Classical reversible circuit for QBALL$_1$)**.** *The map $|\vec{s}\rangle\,|0\rangle \mapsto |\vec{s}\rangle\,|\mathrm{EffEnc}\,V(\vec{s})\rangle$ can be implemented reversibly using $O(r\log(n/r) + r + \log n)$ ancillas and $O(\mathrm{poly}(n))$ gates.*

For completeness, we also show how to (classically) reversibly imeplement QBALL$_2$.

Proposition 3 (Classical reversible circuit for QBALL$_2$)**.** *The map $|\mathrm{EffEnc}\,V(\vec{s})\rangle\,|0\rangle \mapsto |\mathrm{EffEnc}\,V\rangle\,|F(\mathbf{x}(V))\rangle$ can be implemented reversibly using $O(\log n)$ ancillas and $O(\mathrm{poly}(n))$ gates.*

Proof. The algorithm is similar to $\mathsf{Calculate}_i$. Let $G_{j,i}$ be defined as in the implementation of $\mathsf{Calculate}_i$. We can assume that $V = V(\vec{s})$ has exactly r elements. The algorithm is as follows.

1. Run $G_{1,r+1}\dots G_{L,r+1}\,|\mathrm{EffEnc}\,V_r\rangle\,|0\rangle\,|0\rangle$. This results in $|\mathrm{EffEnc}\,V_r\rangle\,|L+1-j_{\min}\rangle\,|1-F(\mathbf{x}(V_{i-1}))\rangle$, where $j_{\min}$ is the smallest j such that C_j is unsatisfied under $\mathbf{x}(V_r)$ if such a clause exist, and $L+1$ otherwise.

2. Use a CNOT to copy the last bit onto the output bit, invert it, and apply the inverse of step 1 to reset the ancillas. □

As mentioned above, this can now easily be turned into a quantum algorithm for PBS.

Theorem 1 (Quantum algorithm for PBS)**.** *There exists a quantum algorithm that solves PBS in runtime $O^*(3^{r/2})$, using at most $O(r\log(n/r) + r + \log n)$ qubits.*

Proof. First, quantize the classical reversible circuits of Proposition 2–3 by turning each (reversible) classical gate into its quantum equivalent. Then, initialise the $|\vec{s}\rangle$ register into

$$\frac{1}{\sqrt{3^r}} \sum_{s_1,\ldots,s_r=1}^{3} |s_1,\ldots,s_r\rangle \tag{31}$$

and apply (quantum) QBall$_1$ followed by QBall$_2$. The latter produces the state

$$\frac{1}{\sqrt{3^r}} \sum_{s_1,\ldots,s_r=1}^{3} |s_1,\ldots,s_r\rangle\, |V(\vec{s})\rangle\, |F(V(\vec{s}))\rangle\,. \tag{32}$$

The last step is to run amplitude amplification (or alternatively, fixed point search [1]) to increase the overlap with 1 on the last qubit. This uses at most $O(\sqrt{3^r})$ repetitions of QBall$_{1,2}$. The overall runtime of the quantum algorithm is therefore $O^*(3^{r/2})$. □

B. Hybrid algorithm and runtime analysis

In this section, we provide the details of the full quantum-enhanced algorithm to solve 3SAT using a small quantum device. In Section B 1 we outline the basic ideas and provide a broad overview. In Section B 2, we look at general runtime properties of the space splitting algorithm which reduces 3SAT to PBS. In Section B 3, we summarise the classical algorithm to solve PBS from [2], which is more efficient than PromiseBall. In Section B 4, we show how that algorithm can be quantum enhanced using QBall and derive its runtime. The full runtime of the hybrid algorithm to solve 3SAT is finally derived in Section B 5.

1. *Key ideas on the hybrid algorithm*

As mentioned in the main text, the results of [3] show that any speedup for PBS leads to a faster algorithm for SAT solving using the space splitting algorithm. This connection is given quantitatively with Eq. (33) in Section B 2 below. Specifically, any algorithm which solves PBS faster than Schöning's algorithm (recall that Schöning's randomized sampling algorithm can equivalently be used to solve PBS) will also outperform Schöning's algorithm for SAT solving.

In the previous section of this Supplemental Materials, we have provided the details of the quantum algorithm which space-efficiently solves the PBS faster than Schöning's algorithm. This algorithm can be used for any r, provided that a quantum device with sufficiently many qubits is available. However, in the space-restricted scenario, we have to resort to other methods. The limit on the number of available qubits translates into a maximum value of r for which we can run the quantum algorithm. To solve PBS for larger values of r, we use a classical algorithm for PBS that recursively calls itself for ever smaller values of r, until we reach a value that the quantum device can handle. We call this general approach the standard hybrid approach. Specifically, we focus on the classical PBS algorithm provided in [2], which is a de-randomization of the algorithm of Schöning, and which we review in Section B 3. This algorithm has two features critical for our purposes. First, this algorithm is (essentially) as fast as the original randomized algorithm of Schöning. This ensures that no thresholds emerge, relative to the algorithm of Schöning, as any speedup for PBS implies a speedup of the original Schöning runtime for SAT solving [4]. Second, it is a recursive divide-and-conquer algorithm: it works by calling itself on an instance smaller relative to all relevant parameters. This is in general a non-trivial demand: the PBS problem involves at least two relevant parameters: r and the size of the formula (which is essentially lower bounded by the number of variables n). The runtime of the algorithm for PBS is exponential only with respect to r, and not n [5], so recursing over r makes sense classically. However, since we assume that we only have a (very) limited-size quantum machine, reducing the PBS instance size only relative to r would not in general suffice. One would naïvely expect to still be required to represent the entire formula on the quantum device, which is impossible since we assume that we have (significantly) fewer qubits than n. However, as explained in the main text, we do not actually need to carry the representation of the formula as input. This is critical for the standard hybrid approach, where the quantum routine is invoked when the instance is small enough to be applicable.

A final contribution of this Section is the overall runtime analysis, provided in Sections B 4 and B 5. The key technical subtleties of this analysis are two-fold. First, a quantum machine which can solve PBS over n variables up to radius r requires somewhat more than r (qu)bits. On the other hand, the expressions quantifying the runtime incorporate the quantities related to the value of r that can be handled. This implies that any expression which quantifies the total runtime relative to a size of the quantum device must include the explicit functional relationship between r, n and the number of qubits we need to run the quantum algorithm for PBS with parameters n, r. The second issue has to do with details of the algorithm in [2], where the recursive calls reduce the instance sizes with respect to non-unit steps of $\Delta > 1$, where Δ influences the efficiency of the overall algorithm. Due to this, the precise analysis of the achieved speed-up via the standard hybrid approach is slightly more involved than for the basic, slower, PromiseBall algorithm presented in the main text. These technical points are elaborated in detail in Section B 4 and B 5.

2. *Runtime properties of the space splitting algorithm*

In [3], it was shown that using the space splitting algorithm (which reduces 3SAT to PBS), 3SAT over n variables can be solved in time

$$T(n) = \underbrace{q_d(n)(2^{3n/d}+2^{(1-h(\rho))n})}_{\text{Cover Set preparation}}+\underbrace{q_d(n)2^{(1-h(\rho))n}}_{\text{Cover Set size}}\times\underbrace{T_2(n,\rho)}_{\text{PBS cost}}, \tag{33}$$

where ρ (the fraction specifying the radius of the balls via $r = \rho n$) and d are parameters which can be optimized, q_d is a polynomial depending on d, $h(\rho) = -(\rho \log_2(\rho) + (1-\rho)\log_2(1-\rho))$ is the binary entropy function, and $T_2(n, \rho)$ is the runtime of the algorithm used to solve PBS with n variables and radius $r = \rho n$.

Schöning's algorithm can also be understood as a PBS solver. It can be shown that Schöning sampling, starting from a center $\mathbf{x}$, which is at Hamming distance r from a satisfying assignment, produces a satisfying assignment with probability at least 2^{-r} [2]. By iteration, we obtain a PBS solver with runtime $O^*(2^r)$.

Since d can be chosen large enough such that the dominating term of $T(n)$ is $2^{(1-h(\rho))n} \times T_2(n, \rho)$ [6], in order to quantify the improvement given a quantum-enhanced subroutine, it suffices to optimize this term with respect to ρ. If (the dominating part of) T_2 is of the form $T_2(n, \rho) = O^*(2^{\zeta\rho n})$, the optimum is obtained by just minimizing $1 - h(\rho) + \zeta\rho$. Important values of ζ are $\zeta = \log_2(3)$ and $\zeta = 1$, corresponding to PBS solved by the deterministic algorithm of [3] and the basic (and also fast deterministic [2]) Schöning's algorithm, attaining optima at values $\rho = 1/4$, and $\rho = 1/3$, respectively. For completeness, the overall effective values γ, for the overall algorithm using PBS routines as listed are approximately 0.585 and 0.415, where the latter matches the runtime of Schöning's (original) algorithm.

3. *Fast deterministic classical PBS solver*

In [2], an improved deterministic classical PBS solver was introduced, with run time in $O^*((2+\epsilon)^r)$, where ϵ depends on tunable protocol parameters. We call this algorithm FastBall. These parameters can be chosen such that ϵ is arbitrarily small or even decaying in r, as explained later. As mentioned, the key idea of this algorithm is to also split the space of choices, selecting which literal will be flipped in the recursive call, into covering balls. For the convenience of the reader, here we present the details of FastBall, adapted from [2]. Let $t \in \mathbb{N}$ be a parameter (influencing ϵ in the overall runtime), and $\mathcal{C} \subseteq \{1, \dots, k\}^t$ be a $k-$arry covering code with radius t/k (where k specifies the clause upper bound in the kSAT problem to be solved). Since t and k are constants, the optimal code can be found in constant time. The key steps and performance aspects of the algorithm are given next, abbreviated and adapted from [2].

```
 1: procedure FastBall($F, \mathbf{x}, r, \mathcal{C}$)
 2:     If $\mathbf{x}$ satisfies $F$
 3:         return $\mathbf{x}$
 4:     else if $r = 0$
 5:         return FALSE
 6:     else
 7:         $G \leftarrow$ a maximal set of pairwise disjoint $k$-clauses of $F$ unsatisfied by $\mathbf{x}$
 8:         if $|G| < t$,                                                    ▷ Case 1
 9:             for each assignment $\beta$ of variables in $G$
10:                 call PromiseBall($F_{|\beta}, \mathbf{x}, r$),
11:         else if $|G| \geq t$                                             ▷ Case 2
12:             $H \leftarrow \{C_1, \ldots, C_t\} \subseteq G$
13:             for each $w \in \mathcal{C}$
14:                 call FastBall($F, \mathbf{x}[H, w], r - \Delta, \mathcal{C}$)   ▷ $\Delta := t - 2t/k$
15: end procedure
```

The parameter Δ influences ϵ, as will be clarified presently. The expression $\mathbf{x}[H, w]$ corresponds to a modified assignment, where the variables selected by the code-word w from the subset of variables occurring in the subset of clauses H have been flipped.

The objective of the algorithm is to achieve the run time of essentially $O^*((k-1)^r)$, which would be the runtime achieved by the randomized algorithm of Schöning, used as a PBS solver. We now briefly discuss the runtime of this algorithm.

In the case the problem is such that the algorithm always encounters Case 1 in line 8, the authors in [2] show that each $F_{|\beta}$ has no clauses of size larger than $k-1$. In this case, Proposition 7 in [2] shows that PromiseBall can solve the problem in $O^*((k-1)^r)$, yielding the overall runtime is $O^*(2^{kt}(k-1)^r)$. Since 2^{kt} is a constant, this is $O^*((k-1)^r)$, achieving the objective.

The more complex case involves occurrences of Case 2 (in line 11). Here, recurrence calls occur, which may encounter Case 1 deeper in the tree, or not. The slowest case occurs when we remain in Case 2 throughout recurrence calls. If we set $\Delta = t - 2t/k$, it was shown in [2] that the runtime of FastBall, which, up to polynomial factors, is the number of leaves in the recursion tree, is $O^*\left((t^2(k-1)^\Delta)^{r/\Delta}\right) = O^*\left((t^{2/\Delta}(k-1))^r\right)$. Since $t^{2/\Delta}$ goes to 1 as t grows, for any $\epsilon > 0$ we can choose $t = t(\epsilon)$ such that $(t^{2/\Delta}(k-1))^r \leq (k-1+\epsilon)^r$, which is the main result. Note that t can also be chosen to be a very slowly growing function of n, $t \approx \log_k \log_2(n)$, which guarantees that the runtime of the algorithm, intuitively, approaches the expression of the form $O^*((k-1)^r)$ as n grows (see [7]).

4. *Quantum speedup of* FastBall *with a small quantum device*

The algorithm FastBall recursively calls itself on smaller instances, where the value of r is reduced in steps of Δ (where $\Delta = t/3$ for 3SAT), and Δ is a function which depends on ϵ alone – hence if ϵ is fixed, Δ is a constant.

At each recurrence step, the algorithm first checks whether a criterion which would ensure that PromiseBall would terminate in $O^*((k-1)^r)$ ($O^*(2^r)$ for 3SAT) steps, is satisfied (Case 1). In that case, the algorithm runs PromiseBall, the quantum enhancement of which was investigated in the main body of the paper. The faster runtime in that case is ensured by the property that the formula in question actually has at most $(k-1)$ (2 for 3SAT) variables per unsatisfied clause. In that case, a quantum enhancement is achieved by the standard hybrid approach, where QBall is run as soon as the instance becomes small enough. Note that since the relevant formula has only $k-1$ variables per clause, QBall can be adapted in this case to yield a runtime of $O^*((k-1)^{r/2})$ if r is small enough. Then, we obtain an interpolated time between Schöning-level performance, and something quadratically faster. We do not need to delve on further details, since this is not the worst-case performance of the algorithm, which occurs if Case 2 persists. We cannot beforehand know at which step, if at all, the criteria for Case 1 will be satisfied, so we call QBall in Case 2 as well, as soon as the recursive step in line 14 calls an instance with sufficiently small r. This QBall-enhanced version of FastBall is a hybrid algorithm which we call QFastBall.

We now estimate the runtime of QFastBall. Recall that we assume a quantum computer with $M = cn$ qubits, where $c \in (0,1)$ is an arbitrary constant. To obtain the runtime of QFastBall, we first determine the largest value of $\tilde{r} = \tilde{r}(c, n)$ such that such a quantum computer can solve PBS of radius $\tilde{r}$.

Recall that QBall requires $O(r\log(n/r) + r + \log n)$ qubits to solve PBS with n variables and radius r. Suppose the exact scaling of this number of qubits is $Ar\ln(n/r) + Br + O(\log n)$ for constants $A, B > 0$ [8]. Let $\beta(c) > 0$ be such that $A\beta(c)\ln(1/\beta(c)) + B\beta(c) = c$. Then, the quantum device can solve PBS for all $r \leq \tilde{r} = \beta(c)n - O(\log n)$. For completeness, it can be shown that $\beta(c) = -c/(AW_{-1}(-ce^{-B/A}/A))$, where W_{-1} is the -1 branch of the Lambert

W function (see Fig. 1 for a plot of $\beta(c)$). We do not need the precise form of $\beta(c)$, it is easy to see, however, that for small values of c, $\beta(c) = \Theta(c/\log(1/c))$ [9].

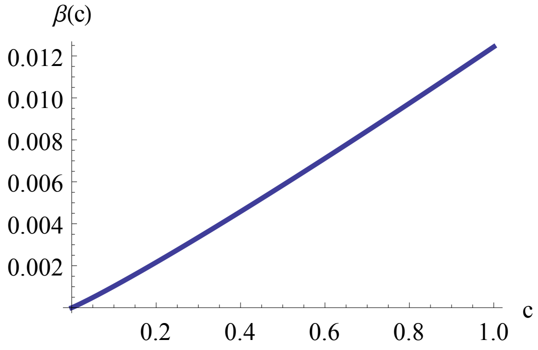


FIG. 1. Plot of $\beta(c)$ for $A \approx 6.93, B = 50$ [8].

Suppose now that QFastBall calls QBall with radius r_{call}. Clearly, $\tilde{r} \geq r_{\text{call}} > \tilde{r} - \Delta$ for each call to QBall. Note that the value of r_{call} could be different for each call to QBall depending on whether we call it deep in PromiseBall (Case 1) or in Case 2. The number of calls to QBall is however at most $O((2t^{2/\Delta})^{r-(\tilde{r}-\Delta)})$, since the number of leaves in the recursion tree, which has depth at most $r - (\tilde{r} - \Delta)$, before QBall is called, is bounded by this quantity. The runtime of QFastBall is therefore given by the product of this quantity and the runtime $O^*(3^{r_{\text{call}}/2})$ of QBall, *i.e.*, at most $O^*((2t^{2/\Delta})^{r-(\tilde{r}-\Delta)} \cdot 3^{\tilde{r}/2}) = O^*((2t^{2/\Delta})^{r-\tilde{r}} \cdot 3^{\tilde{r}/2})$, since Δ and t are constants. Plugging in the expression for $\tilde{r}$ and noting that the $O(\log n)$ contribution is absorbed by the O^* notation, we obtain a runtime of

$$T_{2,\text{QFastBall}}(\rho, n) = O^*\left((2+\epsilon)^{\rho n}\left(\frac{\sqrt{3}}{2+\epsilon}\right)^{\beta(c)n}\right) \tag{34}$$

$$= O^*\left((2+\epsilon)^{\rho n} 2^{-f(c)n}\right) \tag{35}$$

for solving PBS with n variables and radius r using QFastBall with a quantum device of $M = cn$ qubits, where as before ϵ can be made arbitrarily small, and $f(c) = (1 - \log_2 \sqrt{3})\beta(c) \approx 0.21\beta(c)$, where in the last step, we bounded $\sqrt{3}/(2+\epsilon) < \sqrt{3}/2$.

5. *Total runtime for 3SAT*

We now estimate the runtime of the entire algorithm for solving 3SAT using the space splitting algorithm in combination with QFastBall. Substituting (35) into (33), and recalling that we can choose $d = O(1)$ such that the second term in (33) is the dominating term of $T(n)$, we obtain

$$T(n) = O^*\left(2^{(1-h(\rho))n}(2+\epsilon)^{\rho n} 2^{-f(c)n}\right). \tag{36}$$

Note that the last factor in (36), induced by the quantum enhancement, is independent of ρ, and that the first two factors together constitute the runtime of the space splitting algorithm using just FastBall. The optimal value of ρ for (36) is therefore the same as the optimal value of ρ for just using FastBall, which (up to corrections in ϵ) is $\rho = 1/3$. Using this value, we obtain a total runtime of

$$T(n) = O^*\left(2^{(\gamma_0 + \varepsilon - f(c))n}\right), \tag{37}$$

where $\varepsilon = \log_2(1 + \epsilon/2)/3$ can be made arbitrarily small, as stated in the main body of the paper. □

[1] T. J. Yoder, G. H. Low, and I. L. Chuang, Phys. Rev. Lett. **113**, 210501 (2014).

[2] R. A. Moser and D. Scheder, in *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11 (ACM, New York, NY, USA, 2011) pp. 245–252.

[3] E. Dantsin, A. Goerdt, E. A. Hirsch, R. Kannan, J. Kleinberg, C. Papadimitriou, P. Raghavan, and U. Schöning, Theoretical Computer Science **289**, 69 (2002).

[4] More specifically, since we quantum-enhance the algorithm of [2], our approach circumvents a threshold phenomenon relative to this algorithm. Since the algorithm of [2] is referred to as a de-randomization of the algorithm of Schöning, with matching runtime, we for simplicity talk about avoiding the threshold relative to the algorithm of Schöning itself.

[5] For the interested reader we point out that the fine-graining of complexity of a given problem with respect to multiple parameters of a given instance, as is the case here, is studied by so-called *parameterized complexity theory*.

[6] Since d appears in $2^{3n/d}$, to make sure this contribution is not a dominating term in the overall complexity, setting $d = 15$ will suffice as $2^{n/5} \leq 2^{\gamma_q n}$).

[7] `http://users-cs.au.dk/dscheder/SAT2012/searchball.pdf`.

[8] The implementation given in Section A yields $A \approx 6.93$ and $B = 50$ using a straightforward of encoding each trit using two qubits. We remark however that these numbers can be improved significantly.

[9] More precisely, there exist constants $\tilde{c}, X, Y > 0$ such that for all $c \in (0, \tilde{c})$, $Xc/\log(1/c) < \beta(c) < Yc/\log(1/c)$.