Quantum Constructions on Hamiltonians, Codes, and Circuits

Thesis by
Thomas C. Bohdanowicz

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy in Physics

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2022
Defended October 12, 2021

ABSTRACT

This thesis covers three different and largely unrelated projects from my time as Ph.D. student studying quantum information and computation.

In the first chapter, we construct a Hamiltonian whose dynamics simulate the dynamics of every other Hamiltonian up to exponentially long times in the system size. The Hamiltonian is time independent, local, one dimensional, and translation invariant. As a consequence, we show (under plausible computational complexity assumptions) that the circuit complexity of the unitary dynamics under this Hamiltonian grows steadily with time up to an exponential value in system size. This result makes progress on a recent conjecture by Susskind, in the context of the AdS/CFT correspondence, that the time evolution of the thermofield double state of two conformal field theories with a holographic dual has circuit complexity increasing linearly in time, up to exponential time.

In the second chapter, we study *approximate* quantum low-density parity-check (QLDPC) codes, which are approximate quantum error-correcting codes specified as the ground space of a frustration-free local Hamiltonian, whose terms do not necessarily commute. Such codes generalize stabilizer QLDPC codes, which are exact quantum error-correcting codes with sparse, low-weight stabilizer generators (i.e. each stabilizer generator acts on a few qubits, and each qubit participates in a few stabilizer generators). Our investigation is motivated by an important question in Hamiltonian complexity and quantum coding theory: do stabilizer QLDPC codes with constant rate, linear distance, and constant-weight stabilizers exist?

We show that obtaining such optimal scaling of parameters (modulo polylogarithmic corrections) is possible if we go beyond stabilizer codes: we prove the existence of a family of $[[N, k, d, \varepsilon]]$ approximate QLDPC codes that encode $k = \widetilde{\Omega}(N)$ logical qubits into N physical qubits with distance $d = \widetilde{\Omega}(N)$ and approximation infidelity $\varepsilon = O(1/\mathrm{polylog}(N))$. The code space is stabilized by a set of 10-local *noncommuting* projectors, with each physical qubit only participating in $O(\mathrm{polylog}\, N)$ projectors. We prove the existence of an efficient encoding map and show that the spectral gap of the code Hamiltonian scales as $\widetilde{\Omega}(N^{-3.09})$. We also show that arbitrary Pauli errors can be locally detected by circuits of polylogarithmic depth.

Our family of approximate QLDPC codes is based on applying a recent connection between circuit Hamiltonians and approximate quantum codes (Nirkhe, et al., ICALP 2018) to a result showing that random Clifford circuits of polylogarithmic depth yield asymptotically good quantum codes (Brown and Fawzi, ISIT 2013). Then, in order to obtain a code with sparse checks and strong detection of local errors, we use a *spacetime* circuit Hamiltonian construction in order to take advantage of the parallelism of the Brown-Fawzi circuits.

The analysis of the spectral gap of the code Hamiltonian is the main technical contribution of this work. We show that for any depth D quantum circuit on n qubits there is an associated spacetime circuit-to-Hamiltonian construction with spectral gap $\Omega(n^{-3.09}D^{-2}\log^{-6}(n))$. To lower bound this gap we use a Markov chain decomposition method to divide the state space of partially completed circuit configurations into overlapping subsets corresponding to uniform circuit segments of depth $\log n$, which are based on bitonic sorting circuits. We use the combinatorial properties of these circuit configurations to show rapid mixing between the subsets, and within the subsets we develop a novel isomorphism between the local update Markov chain on bitonic circuit configurations and the edge-flip Markov chain on equal-area dyadic tilings, whose mixing time was recently shown to be polynomial (Cannon, Levin, and Stauffer, RANDOM 2017). Previous lower bounds on the spectral gap of spacetime circuit Hamiltonians have all been based on a connection to exactly solvable quantum spin chains and applied only to 1+1 dimensional nearest-neighbor quantum circuits with at least linear depth.

In the third and final chapter, we study the problem of maximum-likelihood (ML) decoding of stabilizer codes under circuit level noise. As progress in the design of proposed fault-tolerant quantum computing architectures moves forward, it is becoming essential to achieve the highest noise suppression possible from the underlying quantum error correcting code. The decoder, which ultimately decides which correction to apply to an encoded state that has suffered an error, is an essential part of this design. So-called maximum likelihood decoders achieve optimal error suppression, but using such a decoder becomes intractable as the size of code grows, therefore sub-optimal decoders which achieve good performance and favorable implementation complexity are used instead. Circuit level noise presents a particular challenge for achieving good performance and practical complexity. We present the construction of a subsystem code called the Circuit History Code which provides an algebraic structure for understanding and classifying circuit level errors. We use this structure to formulate maximum likelihood decoding under circuit level noise as a tensor network contraction. This in turn allows the implementation of *approximate* maximum likelihood decoders which we expect could provide near optimal decoding performance with considerably lower complexity. Using tensor network ML decoders can be useful for benchmarking the performance of efficient decoders being designed for implementation in real experiments, as well as providing options for implementing decoders for codes that would be difficult to decode with conventional methods.

PUBLISHED CONTENT AND CONTRIBUTIONS

The first chapter of this thesis is based on [1]. This work was my first completed project as a graduate student, with the project concept being suggested by my advisor, Fernando Brandão. Other than parts of the introduction, the rest of this manuscript was prepared by myself. The technical construction was designed by myself, based on the construction by Nagaj and Wocjan [72]. Fernando Brandão and Elizabeth Crosson provided helpful guidance in the development of this construction. This work has been presented as a poster at Quantum Information Processing 2018 in Delft, Netherlands and a talk at Quantum Information and String Theory 2019 in Kyoto, Japan. It will be submitted for peer review in the next two months.

The second chapter of this thesis is based on [2]. This work started out of a discussion between myself and Chinmay Nirkhe about similarities between techniques being used in [1] and [73], and became a project about approximate LDPC codes after insightful discussion with Elizabeth Crosson and Henry Yuen. The technical development and writing of results in this work came from equal contributions among all four authors. My most important technical contribution to this work was the development and proof of the counting isomorphism underlying the gap analysis. This work earned an accepted talk at Quantum Information Processing 2019 in Boulder, CO, as well as STOC 2019 in Phoenix, AZ.

The third chapter of this thesis is based on yet unpublished work with Steve Flammia, Chris Chamberland and Giacomo Torlai at the Amazon Web Services Center for Quantum Computing as an intern in the summer of 2020 and 2021. We expect to complete this work and submit it for publication by the year's end.

[1] Thomas C. Bohdanowicz and Fernando G. S. L. Brandão. Universal hamiltonians for exponentially long simulation, 2017.

[2] Thomas C. Bohdanowicz, Elizabeth Crosson, Chinmay Nirkhe, and Henry Yuen. Good approximate quantum ldpc codes from spacetime circuit hamiltonians. *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, Jun 2019. doi: 10.1145/3313276.3316384. URL http://dx.doi.org/10.1145/3313276.3316384.

CONTENTS

LIST OF FIGURES

Chapter 1

UNIVERSAL HAMILTONIANS FOR EXPONENTIALLY LONG SIMULATION

1.1 Introduction

In recent years, there has been an explosion of exciting work using ideas from the theory of quantum information and computation to study and understand open questions in areas of physics like high energy theory (holography and quantum gravity in particular) and condensed matter theory (topological phases of matter in particular). The exploration of ideas at the forefront of quantum gravity research using quantum information tools has been particularly fruitful, with many conjectures having been made about what the theory of computation can tell us about a possible fundamental theory of nature. In this work, we explore broad connections between the Hamiltonian Quantum Cellular Automaton model of quantum computation, classical and quantum complexity theory, and a recent conjecture by Susskind which asserts notions of computational complexity as being fundamental in understanding certain aspects of the AdS/CFT correspondence used in string theory and quantum gravity. In particular, we build an explicit family of Hamiltonians which demonstrates the plausability of Susskind's conjecture while also giving the first local, translation-invariant, and time-independent Hamiltonian whose time evolution can be used to simulate the dynamics of any other for times scaling exponentially in the system size. Further, we assert that there is a notion in which the family of Hamiltonians we present are, by construction, the most complex time-independent, local, and translation-invariant Hamiltonians that can exist.

Universality

The notion of *universality* is central in science. At its core is the idea that a simple set of objects can describe the fundamental properties of a much richer class of objects. One particularly successful example in physics is the use of universality in classifying phase transitions. It turns out that critical phenomena can be understood by looking at only a small set of systems and critical exponents associated to them, which are universal in the sense that they can reproduce the critical behavior of every other system [57]. A second example, which is the focus of this paper, is the study of universal quantum dynamics. The goal here is to understand and classify, in several different scenarios, which quantum dynamics can simulate any other [35]. This line of investigation is at the core of the idea of a (universal) quantum computer, which should be able to efficiently simulate the dynamics of any quantum system [38, 41]. It also naturally extends to the quantum domain the study of universality in classical

dynamics, usually considered in the context of cellular automata.

There are several different notions of universality for quantum dynamics. One of the first to be considered was the notion of universal quantum gates in quantum computation, in which one is interested in identifying and classifying sets of quantum gates (simple one or two-qubit unitary operators) that can approximate any unitary operator. This is a rich problem and there is a beautiful theory around it [36, 59], still with many unresolved questions. A noteworthy result is that almost any two qubit gate is universal in the sense that it, together with the ability to swap the qubits, can approximate any other unitary. Therefore, in a well defined sense, one could say that universality is the general rule for quantum gates.

Another notion concerns universal Hamiltonians. Here we are interested in identifying Hamiltonians whose time evolution can be used to simulate the dynamics of any other Hamiltonian. More precisely, we can ask whether we can perform universal quantum computation with the Hamiltonian. One variant of that notion is to consider sets of few-qubit interactions and classify which are universal, given the ability of controlling the interactions in a time-dependent fashion. This notion is closely related to the one discussed for quantum gates above and, similarly, a generic two-qubit Hamiltonian is typically universal for quantum computing in this sense [30].

Yet another interesting variant is to consider a fixed *time-independent* Hamiltonian and ask if it is universal. It is clear that the class of unitaries which can be approximated by the time evolution of a fixed Hamiltonian is rather limited (as the eigenbasis of e^{itH} for a fixed H is always the same). However the situation is more interesting if one allows for some form of encoding of the desired dynamics (represented by an appropriate quantum circuit which approximates them) by preparing a simple initial state of the system in a suitable way. Indeed, Vollbrecht and Cirac [77] gave a construction of a time-independent and translation-invariant model which is universal for quantum computing. A description of the desired quantum circuit to be implemented by the simulator is encoded in the initial state as a particular computational basis state. Later on, Nagaj and Wocjan made refinements to this construction in Ref. [72]. In contrast to the other notions of universality, it is unclear if a typical local Hamiltonian will be universal in this sense. Indeed, apart from a few examples, it is still largely unexplored which Hamiltonians can give universal dynamics through time-independent evolutions. Further, no other simulation scheme for quantum dynamics is capable of simulating dynamics of an n-qubit system for exponentially long times, $t = O(2^n)$. All current schemes have approximation errors which become large at exponentially long times, or would require use space scaling as $O(2^n)$ to avoid this large error. This motivates us to ask the following question:

Problem 1.1.1. Does there exist a universal scheme that can faithfully simulate the dynamics of an n-qubit system for exponentially long timescales $t = O(2^n)$?

In this work we construct a time-independent, geometrically local, translation invariant Hamiltonian whose dynamics can efficiently (in a sense made more precise below) reproduce the dynamics of any other time-independent Hamiltonian for times up to exponentially large in the system size, using space only polynomially large in the size of the system to be simulated.

Circuit Complexity

An a priori unrelated question in the realm of quantum complexity theory concerns the circuit complexity of quantum dynamics. Note that in the following, we use the notation $f(n)$ is $O(g(n))$ to mean that the function $f(n)$ has an asymptotic growth rate no greater than that of the function $g(n)$, and $f(n)$ is $\Omega(g(n))$ to mean that the function $f(n)$ has an asymptotic growth rate no less than that of $g(n)$. Further, we will use $\mathrm{poly}(n)$ to denote the set of functions growing at rate n^c for any $c > 0$. The circuit complexity of a unitary operator consists of the minimum number of two-qubit gates needed to (approximately) construct the unitary. Formally, we can use the following definition:

Definition 1.1.2. Given a unitary operator $U \in U(2^n)$ acting on n-qubits and a precision $\varepsilon \geqslant 0$, the *circuit complexity of* U to precision ε, $C_\varepsilon(U)$ is the minimum number of 2-qubit gates from the generating set $\{H, P, Toffoli\}$ needed to approximate U to entry-wise precision ε. Similarly, for an n-qubit quantum state $|\psi\rangle$, we define the *circuit complexity of* $|\psi\rangle$ to precision ε, $C_\varepsilon(|\psi\rangle)$, to be the minimum number of 2-qubit gates from the generating set $\{H, P, Toffoli\}$ needed to build a quantum circuit on $m \geqslant n$ qubits such that when applied to $|0\rangle^{\otimes m}$ we obtain a state ρ such that $\|\rho - |\psi\rangle\langle\psi| \otimes |0^{m-n}\rangle\langle 0^{m-n}|\|_{\mathrm{tr}} \leqslant \varepsilon$.

Large circuit complexity is a general feature of unitary operators. Indeed a simple counting argument shows that most unitaries on n qubits will have circuit complexity which grows as $O(2^n)$ [2]. Moreover in Ref. [17] it was shown that most quantum circuits with m gates have circuit complexity $\Omega(m^{1/11})$. It is much harder to prove circuit complexity lower bounds for the evolution of a fixed Hamiltonian (or even for the application of a fixed unitary many times). In fact proving a superpolynomial lower bound on circuit complexity for exponentially long times in the system size n would result in a major breakthrough in classical complexity theory (as shown in Ref. [2], it would imply that the computational complexity class PSPACE is not contained in the class BQP/poly, a conclusion which appears to be far beyond the reach of current proof techniques in theoretical computer science).

It turns out that the concept of universality naturally connects with circuit complexity. If one aims to find a unitary whose dynamics has large circuit complexity, it would be natural to consider one whose time evolution is universal, capable of simulating any other evolution (with a reasonable overhead). Indeed, from the result of Ref. [17], it

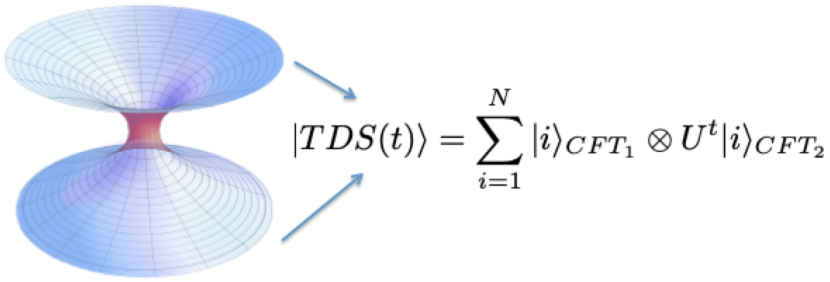


Figure 1.1: A cartoon depiction of a non-traversal AdS wormhole whose boundary consists of a pair of conformal field theories which share the state $|TDS(t)\rangle$.

directly follows that the circuit complexity of the unitary evolution of the Vollbrecht-Cirac universal Hamiltonian [77] on n qubits, for large enough time, is $\Omega(\text{poly}(n))$. However it is not clear how to obtain larger lower bounds (i.e. superpolynomial in n) for long times, e.g. exponential in n, using a physically reasonable Hamiltonian, even under reasonable computational complexity assumptions. We state this as a formal problem that we address in this work.

Problem 1.1.3. Does there exist a local, time-independent and translation-invariant Hamiltonian on n local quantum systems of a fixed dimension whose time evolution can grow to have circuit complexity $\Omega(2^n)$?

This is a natural problem to consider on its own (e.g. it was considered in Ref. [11] under the name of "fast-forwarding" and connected to the time-energy uncertainty relation), but it has recently gained a renewed interest due to a possible unexpected application in the context of quantum gravity and holography, which we now briefly review.

Holography and Susskind's Conjecture

The AdS/CFT correspondence [69] posits the equivalence between a theory of quantum gravity in Anti-de-Sitter (AdS) space (i.e. gravity with a negative cosmological constant) with $d+1$ dimensions (called the bulk theory) and a quantum conformal field theory (CFT) with d dimensions (called the boundary theory). This idea suggests that gravity is an emergent phenomenon (as it does not appear explicitly in the CFT description) and has become one of the most influential ideas in physics in the last twenty years. The correspondence is usually phrased as a dictionary between properties of the two theories. A physical quantity in one theory should have a partner in the other which behaves similarly. Yet there is one quantity of the quantum gravity picture for which it is hard to find a suitable partner in the CFT picture. This is the volume of a non-traversible wormhole (or Einstein-Rosen bridge) in the AdS space connecting two boundary CFTs (see Figure 1.1). One can argue

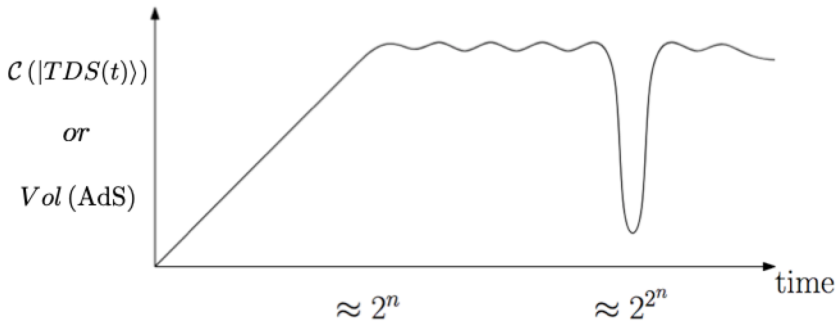


Figure 1.2: This graph shows the time growth of a non-traversible wormhole in AdS space, which is proposed to be the time growth of the circuit complexity of the thermofield double state of the dual conformal field theories, $C_\varepsilon(|TDS(t)\rangle)$.

that the volume of the wormhole will grow linearly with time up to exponential time (see Figure 1.2) [75]. But most of the traditionally considered physical properties of the CFT (which usually involve only few-body operators) will equilibrate much sooner (on the timescale of the scrambling time). One property of the CFT system which does seem to have the same behaviour as the volume of the wormhole is the circuit complexity of the joint quantum state [1] of the two CFTs associated with the boundary of the wormhole. This state is given by the time evolution of the so-called 'thermofield double state' (TDS), which at infinite temperature reduces to a maximally entangled state of the two CFTs. Indeed, in analogy with the behavior of the wormhole volume, we expect the circuit complexity of the state to grow linearly with time up to exponential in the size of the system, at which point it has to saturate (see Figure 1.2). Then at recursion times (which are doubly exponential in system size) the circuit complexity will have sharp oscillations before stabilizing again at an exponentially large value. Such a connection prompted Susskind to conjecture that the circuit complexity of the CFT state is the dual property of the volume of the wormhole. Although fascinating, Susskind's proposal is still somewhat speculative; it is an interesting and challenging task to make it more concrete. Progress was made by Aaronson and Susskind [3]. They considered a toy model of the problem for which concrete results about the circuit complexity could be established. Let V describe one discrete step of the time evolution of the CFT. The CFT thermofield double state (at infinite temperature and regularizing the theory to have finite dimension N) after t time steps is given by

$$|\mathrm{TDS}(t)\rangle := \frac{1}{\sqrt{N}} \sum_i V^t |i\rangle_{\mathrm{CFT},\,1} \otimes (V^T)^t |i\rangle_{\mathrm{CFT},\,2} \tag{1.1.1}$$

$$= \frac{1}{\sqrt{N}} \sum_i |i\rangle_{\mathrm{CFT},\,1} \otimes U^t |i\rangle_{\mathrm{CFT},\,2} \tag{1.1.2}$$

[1] Analogously to the circuit complexity of a unitary, the circuit complexity of a state is the minimum number of two-qubit gates (from a given universal set of gates) needed to (approximately) create the state.

with $U = (V^T)^2$, where V^T is the transpose of V. For Susskind's correspondence to hold, we need the circuit complexity of $|\mathrm{TDS}(t)\rangle$ to grow linearly in time up to an exponential value in the system size $\log N$. Aaronson and Susskind abstracted away the fact that U is associated to the dynamics of a CFT and asked if there is any (efficiently implementable) unitary U for which one can show that the corresponding $|\mathrm{TDS}(t)\rangle$ has complexity growing up to exponential with the number of applications t. They proved that choosing U as a step function of a reversible and computationally-universal cellular automaton achieves the goal (under certain computational complexity assumptions). A natural open question is whether one can prove something similar for the evolution of the CFT. Short of that, can we get closer to this goal? Can we find a local Hamiltonian whose evolution can replace the step function of the cellular automaton in the Aaronson-Susskind reasoning? This motivates the following more difficult variation of Problem 3:

Problem 1.1.4. Can we find a local, time-independent, translation-invariant Hamiltonian whose time evolution generates a circuit complexity whose growth mimics that of the volume of a non-traversible AdS wormhole? That is, can we come up with an H such that a single time step, $U = e^{iH}$, when applied t times, generates a unitary U^t whose circuit complexity $C_\varepsilon(U^t)$ grows *linearly* with t until it saturates at exponentially long times $t = O(2^n)$?

1.2 Main Results

In this paper we consider universality for fixed Hamiltonians up to exponential times. All Hamiltonians will be normalized to have $\|H\| = 1$. Our desired notion of universality is captured by the following definition [2]:

Definition 1.2.1. A family of Hamiltonians $\{H_m\}_{m\in\mathbb{N}}$, indexed by the number of qudits m they act on, is called a *Universal Hamiltonian Family* if for any k-local Hamiltonian $H \in \mathcal{B}(\mathbb{C}^{2^n})$ on n qubits (with $k \leqslant O(\log(n))$) and time t, there are $\mathrm{poly}(n)$-sized quantum circuits D and E, $m = \mathrm{poly}(n, \log t)$, and $t' = \mathrm{poly}(t, n)$ such that

$$\|e^{iHt} - (I^{\otimes n} \otimes \langle 0^{m-n}|)De^{iH_m t'}E(I^{\otimes n} \otimes |0^{m-n}\rangle)\| < 1/\mathrm{poly}(n). \quad (1.2.1)$$

Previous results [72, 77] achieved a weaker notion of universality. In the notation of Definition 1.2.1, the size m of the Hamiltonians in Refs. [72, 77] scales as $m = \mathrm{poly}(n, t)$, which prevents efficient simulation for exponentially long times because it would require preparing an exponentially large state encoding the dynamics. So, in other words, our construction allows exponentially long simulation time using only polynomial space and without requiring any active control or intervention during the simulation. Note also that this notion of universality is incomparable to

[2] This definition was suggested to us by Dorit Aharonov [4].

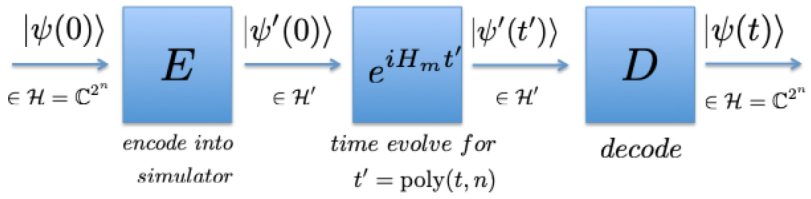


Figure 1.3: Schematic of our simulator. The input is a particular starting state $|\psi(0)\rangle$ from a Hilbert space $\mathcal{H}$ upon which the Hamiltonian H which we wish to simulate acts. After decoding, we obtain the desired time-evolved state $|\psi(t)\rangle = e^{iHt}|\psi(0)\rangle$.

the notion considered in [35]. Rather than reproducing a large class of properties of the system being simulated (e.g. spectrum of Hamiltonian, etc.) our simulation scheme only simulates the *dynamics* and is weaker in that sense. See Figure 1.3. However, our simulation faithfully simulates dynamics for times up to exponentially large in the system size, whereas the construction of Ref. [35] can not, and is in that sense stronger. Although Definition 1.2.1 is natural when discussing universal Hamiltonians, we can also consider the following stronger definition, which will be useful to us:

Definition 1.2.2. A family of Hamiltonians $\{H_m\}_{m\in\mathbb{N}}$, indexed by the number of qudits m they act on, is called a *Circuit Universal Hamiltonian Family* if for every $\mathrm{poly}(n)$-sized circuit U on n qubits and time t, there are $\mathrm{poly}(n)$-sized quantum circuits E (for encoding) and D (for decoding), $m = \mathrm{poly}(n, \log t)$, and $t' = \mathrm{poly}(t, n)$ such that

$$\|U^t - (I^{\otimes n} \otimes \langle 0^{m-n}|)De^{iH_m t'}E(I^{\otimes n} \otimes |0^{m-n}\rangle)\| < 1/\mathrm{poly}(n). \quad (1.2.2)$$

From the Hamiltonian simulation results of Ref. [16], it follows that a *Circuit Universal Hamiltonian Family* is also a *Universal Hamiltonian Family*. Our main result is the following:

Theorem 1.2.3. *There is a* Circuit Universal Hamiltonian Family *in one spatial dimension with translation-invariance and local spin dimension of 14580. The encoding circuit consists of preparing computational basis product states. The decoding circuit consists of making a measurement in the computational basis and resetting computational basis product states to the all zero state.*

It is an open question to decrease the local dimension of the model; we expect substantial improvements to be possible. In words, we explicitly construct a local term $h_{l,l+1}$ acting on two qudits, each of dimension 14580, such that the family of Hamiltonians

$$H_m = \frac{1}{m}\sum_{l=1}^{m-1} h_{l,l+1} \quad (1.2.3)$$

is universal in the sense of Definition 1.2.2.

Theorem 1.2.3 can be used to argue that the circuit complexity of $e^{iH_m t}$ must grow to a superpolynomial value. The starting point is the result of Atia and Aharonov that unless PSPACE = BQP, there must exist a 2-sparse row computable Hamiltonian whose long-time dynamics has superpolynomial circuit complexity (see Theorem 6 of Ref. [11]). This result together with Theorem 1.2.3 gives:

Corollary 1.2.4. *Unless PSPACE = BQP, the circuit complexity of e^{itH_m} is superpolynomial for $t = 2^{O(m)}$.*

An interesting feature of Corollary 1.2.4 is that we can identify a *single* Hamiltonian – more precisely a fixed interaction term forming a translation-invariant model for each length – which can be shown to have growing circuit complexity (under certain computational complexity assumptions).

We can also make progress on Susskind proposal discussed before. Define

$$|\mathrm{TDS}_{H_m}(t)\rangle := \frac{1}{\sqrt{N}} \sum_i |i\rangle \otimes e^{itH_m}|i\rangle. \tag{1.2.4}$$

Then we have:

Corollary 1.2.5. *Assuming PSPACE $\subset$ PP/poly, the circuit complexity of $|TDS_{H_m}(t)\rangle$ is superpolynomial for $t = 2^{O(m)}$. Assuming that there are no subexponential PP circuits for PSPACE, the circuit complexity of e^{itH_m} for $t = 2^{O(m)}$ is $2^{\Omega(m)}$.*

The proof follows directly from Theorem 7.1.2 of Ref. [2], by using H_m to simulate the step function of the universal cellular automaton used there. The corollary makes partial progress on Susskind's proposal, by finding a quantum evolution which shares more features with the dynamics of a CFT (locality and translation-invariance) than the cellular automaton of Ref. [3]. We leave as open questions demonstrations that our construction can replicate important features and symmetries expected of a CFT Hamiltonian, and whether or not it can be used to create a state whose circuit complexity actually grows *linearly* in t from $t = 0$ up to exponential times (under plausible complexity theoretic assumptions).

Sketch of Construction

The construction of our circuit universal Hamiltonian family (CUHF) that proves Theorem 1.2.3 makes use of the concept of a Hamiltonian Quantum Cellular Automaton (HQCA), as described by Nagaj and Wocjan in [72]. The formal definition of an HQCA is as follows:

Definition 1.2.6. A Hamiltonian Quantum Cellular Automaton (HQCA) is a local, time-independent, translation-invariant Hamiltonian H on a lattice of qudits which carries out quantum computation via the following sequence of steps:

1. The input of the computation as well as information describing the computation to be performed (which is described by some unitary operator U) is encoded in the state of the qudits.
2. The qudits undergo continuous time evolution under H for some time t.
3. A simple basis state measurement on a subset of the qudits collapses, with high probability, the state of the whole system to one where an appropriate subset of the qudits contains the desired output of the quantum computation.

Such a Hamiltonian H is termed a cellular automaton because the local terms which sum up to make H can usually be thought of as transitions between different basis states for a local part of the lattice, which themselves correspond to reversible transition rules of some classical cellular automaton. This will be evident in the description of our construction.

Our CUHF $\{H_m\}_{m\in\mathbb{N}}$ will actually be a family of universal HQCA on a $1D$ chain of m qudits, which satisfy the following properties in addition to those in the definition above:

- The local qudit dimension is constant.
- A quantum circuit U with C gates acting on N qubits can be applied t times through time evolution via Hamiltonian H_m on a chain of $m = \text{poly}(N, C, \log t)$ qudits.

The starting point of our construction is that of Nagaj and Wocjan [72], which produces a $1D$ CUHF with constant local qudit dimension $d = 20$, for which the size of the chain scales as $m = \text{poly}(N, C)$. A version of this construction is described in detail in section 1.3, where we also identify and highlight some important features of their construction that are essential to preserve in our eventual final construction [3].

We build an HQCA which only encodes U into the input state and a number t encoded in binary, but which is able to apply U repeatedly up to t times after waiting for a time $t' = \text{poly}(N, C, t)$ [1]. Therefore the size of the system needed for the simulation is only $m = \text{poly}(N, \log t)$. In the following sections, we proceed to augment Nagaj-Wocjan construction [72] step by step to add in more functionality until we achieve the desired $m = \text{poly}(N, \log t)$ scaling for the simulation size.

[3] E.g., the definition and consequences of having a uniquely orthogonally generated set of states for an HQCA, see section 1.3

The innovations which allow us to obtain our final construction dictate the organization of the remainder of the paper:

1. The Nagaj-Wocjan construction can only apply its encoded unitary U once. So, first, we modify the Nagaj-Wocjan construction so that when the HQCA reaches its end state where U has been applied, it can undergo transitions to 'reset' itself and allow for another application of U without undoing the original application of U. This basically has the effect of allowing the Nagaj-Wocjan construction to run over and over again rather than just once. This is described in Section 1.4.

2. Next, we augment the construction to include a binary clock in the qudit chain which counts the number of times that U has been applied. This allows states of the chain with different numbers of U applications to be orthogonal so that they can be distinguished by a simple measurement. Binary clock constructions of a similar nature also appear in [8, 34, 51], in the context of proving hardness/uncomputability results for the ground-energy/gap of translation-invariant models. This is described in Section 1.5.

3. Finally, we augment the construction with the addition of a target register that allows one to specify a certain desired number of applications of U which is stored in the start state. Once the clock has counted this number of applications of U, the HQCA continues evolving forward in time similarly to the way in which it did before, but it no longer makes any applications of U to the qubits. This allows one to have complete control over how many times U is applied to the qubits, and is crucial to being able to guarantee that after waiting a reasonable amount of time, performing a simple measurement on a subset of the chain is likely to yield a state with the desired number of applications of U. This is described in Section 1.6.

The main challenge in achieving each of these results was to carefully engineer transition rules such that starting from the appropriate start state, for every state that can be obtained from the start state through application of transition rules, exactly one forward transition rule and one reverse transition rule is applicable to that state. This guarantees that the sequence of states obtained from the start state through application of the HQCA transition rules can map directly onto position eigenstates of a single particle on a chain, so that the analysis of the time evolution under the HQCA reduces to that of a single particle continuous-time quantum walk on a line (see Sections 1.3 and 1.6).

1.3 Construction I: Applying U Once

We begin with the modest goal of applying U to our qubits once ($x = 1$) using the time evolution of a simple time-independent and translation-invariant Hamiltonian,

where the sequence of gates that implements U is stored in the start state of the qudit chain that we embed our qubits into for the simulation. The construction described in this section only differs from the $d = 20$ HQCA construction of Nagaj and Wocjan [72] in notation (and only slightly at that). We do, however, formalize and elaborate on some of the essential ideas underlying the success and utility of their construction which will be important in subsequent sections when we expand their construction, the most important of which being the definition of what it means for a sequences of quantum states to be 'uniquely orthogonally generated' by a sequence of transition rules of an HQCA (see Section 1.3). The construction described in this section will be referred to as 'construction I' throughout the rest of the paper.

Gateset

The universal gate set we will use to describe a quantum circuit for U is $\mathcal{G} = \{W, S, I\}$ where each gate acts on a local pair of left and right qubits and

- W is a controlled rotation by $\pi/2$ about the y-axis of the Bloch sphere (where the left qubit always controls the right)
- S is the two-qubit swap gate
- I is shorthand for the two-qubit identity gate $I \otimes I$.

We note that because all of these gates leave the state $|00\rangle$ invariant, to use these gates to simulate the application of an arbitrary U on a state of the form $|0^n\rangle$, some number of qubits in the $|1\rangle$ state must be provided in order to simulate gates that act non-trivially on the $|0\rangle$ state. This overhead is polynomial in the number of gates that U would have in a circuit using a more traditional gateset, so we ignore it and assume that the state that our gates act on has the correct number of qubits in $|1\rangle$ states for the simulation to proceed.

For the convenience of the construction, the way that U breaks down into a sequence of gates from $\mathcal{G}$ will be as follows: we write the unitary as a gate sequence from $\mathcal{G}$ in the form

$$\begin{aligned} U = (U_{K,1} \dots U_{K,N-1}) II (U_{K-1,1} \dots U_{K-1,N-1}) II \dots \\ \dots II (U_{1,1} \dots U_{1,N-1}) I. \end{aligned} \tag{1.3.1}$$

Here we say that U is expressed as K rounds of $N-1$ gates from $\mathcal{G}$. Note that the use of the term 'round' of gates differs here from the usual usage. Normally, a round of gates from some gate set is some tensor product of $m \leqslant N$ single or two-qubit gates with the property that each of the N qubits has exactly one gate acting on it (identity counted as a gate). This allows the number of rounds making up U to count the time complexity of applying the circuit. This is not true of our rounds. The gate

$U_{k,m} \in \mathcal{G}$ is a two qubit gate acting on qubits m and $m+1$. So, two successive gates in a round, $U_{k,m}$ and $U_{k,m+1}$ will possibly each act non-trivially on qubit $m+1$. Thus, the number of rounds, K, in our circuit description of U will in general under-count the depth or time complexity of applying U as written, but it will undercount at most by a factor of N. The reason we write our circuit this way is for the convenience of constructing our desired HQCA. For ease of discussion, we will use the term *depth* of U to refer to the numbe of rounds, K, in the above sense rather than the traditional sense.

Notice also that in the above description, each round of gates is padded with an additional I on either side, except for the final round which does not have an I on the left side. The reason for this is that it makes the construction of our desired HQCA more convenient, and will be explained below. With the above convention for describing U in terms of $\mathcal{G}$, a unitary U of depth K acting on N qubits has a circuit sequence of length $K(N+1)-1$.

Hilbert Space

Here we describe the Hilbert space C of the 1D chain that our HQCA Hamiltonian H will act on. We imagine our chain as consisting of two registers $C = P \otimes D$, each with L local sites $C_j = P_j \otimes D_j$, indexed from left to right. We will describe orthonormal basis states for each local site Hilbert space in terms of symbols:

- The program register P, whose local sites are P_j, stores the sequence of gates that describe U and controls its application to the work qubits. The local hilbert space is 10 dimensional, with basis symbols $\{W, S, I, \overrightarrow{W}, \overrightarrow{S}, \overrightarrow{I}, \blacktriangleleft, \triangleleft, \ldots, \rightarrow\}$. So, $P = \mathbb{C}^{10}$.
- The data register D, whose local sites D_j are qubits described by computational basis symbols $\{0, 1\}$, is simply a chain of qubits, N of which are the 'work qubits' to which we want to apply the unitary U. So, $D = \mathbb{C}^2$.

The symbols in the basis states for P_j have the following interpretation

- W, S, I: representing unitary gates in the program sequence that will apply U,
- $\overrightarrow{W}$, $\overrightarrow{S}$, $\overrightarrow{I}$: marked characters in the program sequence, used to propagate the active spot to the front (left) of the program sequence,
- $\blacktriangleleft$: apply gate symbol,
- $\triangleleft$: shift program forward,

- $\rightarrow$: a control symbol indicating that we're in the process of shifting the program sequence to the right,
- $\mathbf{.}$: empty spot (before/after the program).

The way these symbols will interact will become clearer when illustrating the transition rules defined below. The length of the chain will be $L = (2K-1)(N+1)+2$ where K is the number of rounds/depth of the circuit description of U, and N is the number of qubits that U is being applied to. The N qubits which U is being applied to will be referred to as the *work qubits*, and their location in the chain C will be the sites D_j of the data register with $j = (K-1)(N+1)+1+n$ for $n = 1...N$.

Initial State

The rest of the construction will be described with a simple illustrative example in mind, where we are applying a $K = 2$-round unitary $U = (S_{12}W_{23})II(W_{12}S_{23})I$ to $N = 3$ work qubits. Details regarding certain choices about the structure of the initial state will make sense after describing the transition rules and running through applying them to this example initial state. This chain will have $L = 14$, and the initial state is written as a product state over the sites and registers: $|\psi_0\rangle = \bigotimes_{j=1}^{14}(|p_j\rangle \otimes |d_j\rangle)$ where $|p_j\rangle$ is the state of the jth site of the program register and $|d_j\rangle$ is the state of the jth site of the data register. The following table describes the initial state:

j	1	$\cdots$											$\cdots$	L
p_j	$\rightarrow$	S	W	I	I	W	S	I	.	.	.	.	.	.
d_j	1	0	0	0	1	w_1	w_2	w_3	1	0	0	0	1	0

(1.3.2)

Note the following about the structure of the initial state:

- The individual initial states of the work qubits have here been labelled w_1, w_2 and w_3 to remind the reader where they are located.
- The work qubits are bordered by data qubits in the 1 state on either side
- The work qubits are further padded to the left by the sequence 10^N $K-1$ times, and to the right by the sequence 0^N1 $K-1$ times
- The final data qubit is in the state 0

The reason for this structure will become clearer in the description and application of the transition rules to this example, and will be commented upon further below.

Transition Rules

We now describe the forward transition rules of our HQCA. Application of these forward transition rules to our initial state $|\psi_0\rangle$ will create a sequence of $T_I + 1$ mutually orthogonal states $|\psi_t\rangle_{t=0}^{T_I}$. Note that T_I is a function of N and K. The transition rules are deliberately constructed such that for each $|\psi_t\rangle$, there is always one single unique forward transition rule that can be applied to it, which results in the state $|\psi_{t+1}\rangle$. For every forward transition rule there is a unique reverse transition rule (the Hermitian conjugate of the forward transition rule), and for each $|\psi_t\rangle$, there is always one single unique reverse transition rule that can be applied to it, which results in $|\psi_{t-1}\rangle$. No reverse transition rule will be applicable to $|\psi_0\rangle$, and no forward transition rule will be applicable to $|\psi_{T_I}\rangle$.

The transition rules are summarised as follows:

$$1: \quad \boxed{\rightarrow}\boxed{A} \quad \rightarrow \quad \boxed{\cdot}\boxed{\overrightarrow{A}} \tag{1.3.3}$$

$$2: \quad \boxed{\overrightarrow{A}}\boxed{B} \quad \rightarrow \quad \boxed{A}\boxed{\overrightarrow{B}} \tag{1.3.4}$$

$$3: \quad \boxed{\overrightarrow{A}}\boxed{\cdot} \quad \rightarrow \quad \boxed{A}\boxed{\rightarrow} \tag{1.3.5}$$

$$\begin{aligned} 4a: \quad & \begin{array}{|c|c|}\hline \rightarrow & \cdot \\ \hline 1 & \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \blacktriangleleft & \cdot \\ \hline 1 & \\ \hline \end{array} \\ 4b: \quad & \begin{array}{|c|c|}\hline \rightarrow & \cdot \\ \hline 0 & \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \triangleleft & \cdot \\ \hline 0 & \\ \hline \end{array} \end{aligned} \tag{1.3.6}$$

$$\begin{aligned} 5a: \quad & \begin{array}{|c|c|}\hline A & \blacktriangleleft \\ \hline \psi_{j,j+1} & \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \blacktriangleleft & A \\ \hline A(\psi_{j,j+1}) & \\ \hline \end{array} \\ 5b: \quad & \boxed{A}\boxed{\triangleleft} \quad \rightarrow \quad \boxed{\triangleleft}\boxed{A} \end{aligned} \tag{1.3.7}$$

$$\begin{aligned} 6a: \quad & \begin{array}{|c|c|}\hline \cdot & \blacktriangleleft \\ \hline 1 & \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \cdot & \rightarrow \\ \hline 1 & \\ \hline \end{array} \\ 6b: \quad & \begin{array}{|c|c|}\hline \cdot & \triangleleft \\ \hline 0 & \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \cdot & \rightarrow \\ \hline 0 & \\ \hline \end{array} \end{aligned} \tag{1.3.8}$$

Here, $A,\ B \in \mathcal{G}$, and $\psi_{j,j+1}$ refers to the state of qubits j and $j+1$ and $A(\psi_{j,j+1})$ refers to the gate A applied to the state of qubits j and $j+1$. We will name this set of forward transition rules $\mathcal{F}_I$

Active Site

A subset of the basis states for P_j whose symbols contain directional arrows of some form, $\mathcal{A}_I = \{\rightarrow, \overrightarrow{W}, \overrightarrow{I}, \overrightarrow{S}, \blacktriangleleft, \triangleleft\}$, are called *active symbols*. By construction, the initial state and all states resulting from applying a sequence of transition rules to the initial state will always contain exactly one active symbol, the site at which it is located being referred to as the *active site* of the chain. All transition rules for the HQCA involve the propagation of the active site to a neighbouring site and/or transforming one active symbol into another. This feature will be helpful in analysing the construction below.

Illustration

Now we illustrate the use of these transition rules on our example initial state to demonstrate features of the construction and how the HQCA evolves. We start with initial state $|\psi_0\rangle$ described by the table in equation 1.3.2:

$$|\psi_0\rangle = \begin{bmatrix} \rightarrow & S & W & I & I & W & S & I & . & . & . & . & . & . \\ 1 & 0 & 0 & 0 & 1 & w_1 & w_2 & w_3 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \tag{1.3.9}$$

Applying the sequence of forward transition rules 1, 2 (6 times), and 3, we reach the state,

$$|\psi_8\rangle = \begin{bmatrix} . & S & W & I & I & W & S & I & \rightarrow & . & . & . & . & . \\ 1 & 0 & 0 & 0 & 1 & w_1 & w_2 & w_3 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{1.3.10}$$

At this point, transition rule 4a applies, turning the $\rightarrow$ symbol into the gate application symbol $\blacktriangleleft$, giving

$$|\psi_9\rangle = \begin{bmatrix} . & S & W & I & I & W & S & I & \blacktriangleleft & . & . & . & . & . \\ 1 & 0 & 0 & 0 & 1 & w_1 & w_2 & w_3 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{1.3.11}$$

Now, rule 5a is applied 3 times, which applies the first round of gates to the work qubits:

$$|\psi_{12}\rangle = \begin{bmatrix} . & S & W & I & I & \blacktriangleleft & W & S & I & . & . & . & . & . \\ 1 & 0 & 0 & 0 & 1 & S_{23}\,(W_{12}\,(w_1 w_2 w_3)) & & & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{1.3.12}$$

Applying rule 5a four more times will have propagated the active site as far back to the left as it can go, giving

$$|\psi_{16}\rangle = \begin{bmatrix} . & \blacktriangleleft & S & W & I & I & W & S & I & . & . & . & . & . \\ 1 & 0 & 0 & 0 & 1 & S_{23}\,(W_{12}\,(w_1 w_2 w_3)) & & & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \tag{1.3.13}$$

and applying rule 6a yields

$$|\psi_{17}\rangle = \begin{bmatrix} . & \rightarrow & S & W & I & I & W & S & I & . & . & . & . & . \\ 1 & 0 & 0 & 0 & 1 & S_{23}\,(W_{12}\,(w_1 w_2 w_3)) & & & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{1.3.14}$$

So, we see that applying this *unique* sequence of 17 forward transitions has moved the active site $\rightarrow$ from the far left of the gate sequence over the to the far right of the gate sequence (equation 1.3.10), converted the active symbol $\rightarrow$ into the active symbol $\blacktriangleleft$ which is the gate applying symbol (equation 1.3.11), which then moved back to the left applying gates to the qubits below, shifting the gate sequence a single site to the right as it went (equations 1.3.12, 1.3.13 and 1.3.14). We will refer to this single back and forth movement of the active site as a right-moving *oscillation* of the active site. The halfway point of an oscillation, just before the $\rightarrow$ symbol turns around to move left as $\blacktriangleleft$ (as illustrated in equation 1.3.10) is called the *turning point* of the oscillation.

From the above illustration of the first oscillation, one can now see that applying this same sequence of 17 forward transitions over and over again will simply generate further right moving oscillations of the active site that proceed nearly identically, shifting the entire gate sequence one space to the right with each completed oscillation, until the entire gate sequence has been shifted all the way to the right. For example, after the second right-moving oscillation, the state of the chain will be

$$|\psi_{34}\rangle = \left[\begin{array}{cccccccccccccc} \cdot & \cdot & \rightarrow & S & W & I & I & W & S & I & \cdot & \cdot & \cdot & \cdot \\ 1 & 0 & 0 & 0 & 1 & S_{23}\,(W_{12}\,(w_1w_2w_3)) & & & 1 & 0 & 0 & 0 & 1 & 0 \end{array}\right], \quad (1.3.15)$$

and after the third oscillation,

$$|\psi_{51}\rangle = \left[\begin{array}{cccccccccccccc} \cdot & \cdot & \cdot & \rightarrow & S & W & I & I & W & S & I & \cdot & \cdot & \cdot \\ 1 & 0 & 0 & 0 & 1 & S_{23}\,(W_{12}\,(w_1w_2w_3)) & & & 1 & 0 & 0 & 0 & 1 & 0 \end{array}\right], \quad (1.3.16)$$

after the fourth,

$$|\psi_{68}\rangle = \left[\begin{array}{cccccccccccccc} \cdot & \cdot & \cdot & \cdot & \rightarrow & S & W & I & I & W & S & I & \cdot & \cdot \\ 1 & 0 & 0 & 0 & 1 & S_{23}\,(W_{12}\,(w_1w_2w_3)) & & & 1 & 0 & 0 & 0 & 1 & 0 \end{array}\right], \quad (1.3.17)$$

and finally after the fifth,

$$|\psi_{85}\rangle = \left[\begin{array}{cccccccccccccc} \cdot & \cdot & \cdot & \cdot & \cdot & \rightarrow & S & W & I & I & W & S & I & \cdot \\ 1 & 0 & 0 & 0 & 1 & U\,(w_1w_2w_3) & & & 1 & 0 & 0 & 0 & 1 & 0 \end{array}\right]. \quad (1.3.18)$$

Note that the first round of gates was applied to the work qubits during the first oscillation, no gates were applied during oscillations 2, 3, or 4, and the second round of gates was applied during the fifth oscillation, completing the application of the desired U. Whether or not a particular oscillation will apply gates to the data qubits in the second half of the oscillation depends on whether or not there is a 1 or 0 in the data qubit below the active symbol when it reaches the far right of the oscillation. We can see in equations 1.3.10 and 1.3.11, the active symbol $\rightarrow$ becomes an apply gate symbol $\blacktriangleleft$ because there is a 1 in the data register of the active site at the turning point. If there were a zero instead, which is true at the turning points of the next three oscillations, $\rightarrow$ would turn into $\triangleleft$ and no gates would be applied in the second half of the oscillation. This is good because the second round of gates is not yet

aligned with the work qubits (see equations 1.3.14, 1.3.15 and 1.3.16), and we do not want to be applying any gates until this alignment is achieved.

So, we see that the pattern of ones and zeros in the data qubits to the right of the work qubits is simply chosen so that transition rules 4a and 4b can properly enforce that gates are only applied for the first in every $N+1$ oscillations, which is precisely when rounds of gates will properly be aligned to the work qubits that they will be applied to. The pattern of the data qubits to the left of the work qubits mirrors that on the right so that transition rules 6a and 6b can properly convert $\blacktriangleleft$ or $\triangleleft$ back into $\rightarrow$ at the end of the oscillation. Finally, note also that whenever we begin an oscillation that applies gates, rounds of gates that are not being applied during that round are aligned so that they are always applied to $|00\rangle$ qubit states, on which the S and W gates both act trivially. So, we see that the states on the non-work qubits in the data register remain invariant under the application of our HQCA transition rules.

Finally, after one more half oscillation (8 steps) we reach a state $|\psi_{93}\rangle$ to which no forward transition rules apply:

$$|\psi_{93}\rangle = \begin{bmatrix} . & . & . & . & . & . & S & W & I & I & W & S & I & \rightarrow \\ 1 & 0 & 0 & 0 & 1 & U(w_1w_2w_3) & & 1 & 0 & 0 & 0 & 1 & 0 & \end{bmatrix}. \quad (1.3.19)$$

General Case

Now that we understand the construction via the above worked example, we can make some statements about the general case where we are applying a depth-K unitary to N work qubits. In general, if we want to apply depth-K unitary in the form of equation 1.3.1 to N qubits, we initialize the state of the length $L = (2K-1)(N+1)+2$ chain of $d = 20$ qudits as follows:

- Program register P: $P_1 = \rightarrow$, $P_2 ... P_{K(N+1)} = (U_{K,1} \dots U_{K,N-1})II(U_{K-1,1} \dots U_{K-1,N-1})II \cdots II(U_{1,1} \dots U_{1,N-1})I$, $P_{j>K(N+1)} = .$
- Data register D: $D_1 ... D_{(K-1)(N+1)+1} = (1(0)^N)^{K-1}1$, $D_{(K-1)(N+1)+2} ... D_{(K-1)(N+1)+1+N} = 0^N$ (these are the work qubits), $D_{(K-1)(N+1)+1+N+1} ... D_{L=(2K-1)(N+1)+2} = 1(0^N1)^{K-1}0$

Consider an instance of construction I with a K-depth unitary being applied to N work qubits. A single full right-moving oscillation will be achieved through the application of $2K(N+1)+1$ forward transitions: Rule 1 once and then rule 2 $K(N+1)-2$ times, and then rule 3 once completes the first half of the oscillation where the active symbol moves all the way to the right and gets to the turning point. Then applying rule $4a/b$ once prepares us for the second half of the oscillation where the active site moves all the way back to the left which is accomplished by applying rule $5a/b$ $K(N+1)-1$ times and then applying $6a/b$ once to prepare the active site

to begin the next oscillation. The first half of a right-moving oscillation is $K(N+1)$ steps and the second half is $K(N+1)+1$ steps. The final state is reached after $N(K-1)+K$ full oscillations and one half oscillation (of $K(N+1)$ steps). So, the total number of forward transitions needed to reach the end state from the start state is $T_I = 2N^2K^2 - 2N^2K + 4NK^2 - N + 2K^2 + 2K$.

Note that at each step of the sequence of forward transition moves applied on a proper start state $|\psi_0\rangle$, there is only ever exactly one forward transition that can be applied (unless you have reached the end state $|\psi_{T_I}\rangle$). This is, essentially, due to the fact that the start state has a single active symbol, and the transition rules were specifically designed to move and transform the active symbol in a prescribed way.

Nagaj and Wocjan [72] show that by time evolving the appropriate initial state $|\psi_0\rangle$ encoding the desired unitary under Hamiltonian H_I for time polynomial in K and N, a simple measurement on a subset of the chain will collapse the state of the work qubits to $U|0^N\rangle$ with high probability. We defer any run-time analysis to the end of the paper where we will present the run-time analysis for our full construction which is based on this one.

Hamiltonian and Hilbert Space Geometry

Let us choose a Hamiltonian H_I for this system as a sum of translationally invariant terms:

$$H_I = -\frac{1}{6bL}\sum_{i=1}^{L-1}\sum_{k=1}^{6b}\left(P_k + P_k^\dagger\right)_{(i,i+1)} \tag{1.3.20}$$

where the terms $P_{ki,i+1}$ correspond to the rules 1-6b and act on two neighboring qudits as

$$P_{1i,i+1} = \sum_{A\in\{W,S,I\}} |\ .\ \overrightarrow{A}\rangle\langle\rightarrow A\,|_{p_i,p_{i+1}} \otimes \mathbb{I}_{d_i,d_{i+1}}, \tag{1.3.21}$$

$$P_{2i,i+1} = \sum_{A,B\in\{W,S,I\}} |A\ \overrightarrow{B}\rangle\langle\overrightarrow{A}\ B\,|_{p_i,p_{i+1}} \otimes \mathbb{I}_{d_i,d_{i+1}}, \tag{1.3.22}$$

$$P_{3i,i+1} = \sum_{A\in\{W,S,I\}} |A\ \rightarrow\rangle\langle\overrightarrow{A}\ .\,|_{p_i,p_{i+1}} \otimes \mathbb{I}_{d_i,d_{i+1}}, \tag{1.3.23}$$

$$P_{4ai,i+1} = |\ \blacktriangleleft\ .\rangle\langle\rightarrow\ .\,|_{p_i,p_{i+1}} \otimes |1\rangle\langle 1|_{d_i} \otimes \mathbb{I}_{d_{i+1}}, \tag{1.3.24}$$

$$P_{4bi,i+1} = |\ \triangleleft\ .\rangle\langle\rightarrow\ .\,|_{p_i,p_{i+1}} \otimes |0\rangle\langle 0|_{d_i} \otimes \mathbb{I}_{d_{i+1}}, \tag{1.3.25}$$

$$P_{5ai,i+1} = \sum_{A\in\{W,S,I\}} |\ \blacktriangleleft\ A\rangle\langle A\ \blacktriangleleft\,|_{p_i,p_{i+1}} \otimes A_{d_i,d_{i+1}}, \tag{1.3.26}$$

$$P_{5bi,i+1} = \sum_{A\in\{W,S,I\}} |\ \triangleleft\ A\rangle\langle A\ \triangleleft\,|_{p_i,p_{i+1}} \otimes \mathbb{I}_{d_i,d_{i+1}}, \tag{1.3.27}$$

$$P_{6ai,i+1} = |\ .\ \rightarrow\rangle\langle\ .\ \blacktriangleleft\,|_{p_i,p_{i+1}} \otimes \mathbb{I}_{d_i} \otimes |1\rangle\langle 1|_{d_{i+1}}, \tag{1.3.28}$$

$$P_{6bi,i+1} \quad = \quad |\,.\,\rightarrow\rangle\langle\,.\,\triangleleft\,|_{p_i,p_{i+1}} \otimes \mathbb{I}_{d_i} \otimes |0\rangle\langle 0|_{d_{i+1}}. \tag{1.3.29}$$

Each local term in the sum making up H, when acting on a basis state of C, either changes the state by implementing the corresponding HQCA forward/reverse transition rule or annihilates it if the rule can not be applied to that state. The following definition will be essential for further understanding and analyzing the time evolution generated by this Hamiltonian and those based on it later in the paper:

Definition 1.3.1. Consider the Hilbert space of states C of any HQCA described in this paper. A sequence of $T+1$ states from C, $\{|\psi_t\rangle\}_{t=0}^{T}$, is called *Uniquely Orthogonally Generated* (UOG) by sequence of forward transition rules of the HQCA, $\mathcal{S}$ (which has length T), if the two following conditions are satisfied:

1. The states $\{|\psi_t\rangle\}_{t=0}^{T}$ are each a product state over all sites and registers aside from the work qubits, and are mutually orthogonal (thus the states differ in at least one site of the chain)

2. For each state $|\psi_{t<T}\rangle$, there is exactly one forward transition rule that can be applied to the state, and the application of this rule yields the state $|\psi_{t+1}\rangle$

It is clear from the illustration of a single right-moving oscillation as in the example above that the sequence of intermediate states of the chain throughout a right-moving oscillation is UOG by the sequence of transitions described above.

We will name the unique sequence of forward transitions that takes the initial state of the chain, $|\psi_0\rangle$, to the final state $|\psi_{T_I}\rangle$ $\mathcal{T}_I \in \mathcal{F}_I^{T_I}$. This sequence $\{|\psi_t\rangle\}_{t=0}^{T_I}$ is UOG by $\mathcal{T}_I$. This is easily seen from the fact that for each individual right-moving oscillation, the set of transitions that implements it is unique, and for each successive oscillation the sets of intermediate states are orthogonal to those of all other oscillations because the program sequence starts at a different position in each oscillation. Thus, every state in the sequence $\{|\psi_t\rangle\}_{t=0}^{T_I}$ is orthonormal to all others in the sequence. Further, for each state in $\{|\psi_t\rangle\}_{t=0}^{T_I}$, only one forward transition can be applied at each step because this was true of the individual right-moving oscillations. Thus $\{|\psi_t\rangle\}_{t=0}^{T_I}$ is UOG by $\mathcal{T}_I$.

Now we can see the benefit of the UOG property. The fact that the sequence of states $\{|\psi_t\rangle\}_{t=0}^{T_I}$ is UOG by $\mathcal{T}_I$ allows one to be able to think of the set of states $|\psi_t\rangle$ as the set of positions of a particle on a line, so H_I becomes

$$H_{I,line} = -\sum_{t=0}^{T-1} \left(|t\rangle\langle t+1| + |t+1\rangle\langle t|\right). \tag{1.3.30}$$

This is the Hamiltonian of a (continuous-time) quantum walk on a line of length T_I+1. Therefore, the HQCA Hamiltonian H_I induces a continuous quantum walk

on the "line" of states $\{|\psi_t\rangle\}_{t=0}^{T_I}$ of the qudit chain of length L. Note that starting from an appropriately defined start state $|\psi_0\rangle$, the outcome of any time evolution under H_I is restricted to the subspace of states spanned by $\{|\psi_t\rangle\}_{t=0}^{T_I}$.

Summary and Conclusions

- Given a depth K unitary on N qubits, we can create a simple initial state $|\psi_0\rangle$ on a qudit chain of length $L = (2K-1)(N+1)+2$ whose local site dimension is $d = 20$ (itself factorizing locally into a qubit and a $d = 10$ qudit) which encodes the desired unitary
- The forward transition rules and their reverse versions define the Hamiltonian H_I which encodes an HQCA
- Starting from $|\psi_0\rangle$ and applying only forward transition rules, one obtains a *unique* sequence of mutually orthogonal states $\{|\psi_t\rangle\}_{t=0}^{T_I-1}$. For each $|\psi_t\rangle$, there is only ever exactly one forward transition rule that can be applied, from which one obtains $|\psi_{t+1}\rangle$
- The continuous time evolution of $|\psi_0\rangle$ under H_I can be mapped to a simple 1D quantum walk of a single particle on a chain of length $T_I + 1$
- The state $|\psi_{T_I}\rangle$ has the desirable feature that it is a product state among all sites and registers except for the work qubits, and the work qubits have had the desired unitary applied to them. No forward transition rule applies to $|\psi_{T_I}\rangle$.
- One can do a simple computational basis measurement on the program register to see if the program sequence has moved far enough to the right for U to have been applied to the qubits

1.4 Construction II - Applying U More Than Once

What is needed in order to apply U more than once? In the construction that has been made so far, we see that after applying U once to the state of the work qubits, the state of the HQCA reaches a state for which no forward transition rules can be applied. In the language of the quantum walk, the particle has reached the end of the line, so all it can do is turn around and walk the other way, which would undo the application of U to the work qubits. So under arbitrarily long time evolution U will never be applied to the work qubits more than once. To apply U twice, we would desire to be in a state like the start state in Eq. (1.3.9) except with U already applied to the work qubits. So, if there were some way of taking the end state in Eq. (1.3.19) and then somehow move the program sequence back to its starting position *without undoing the application of U to the work qubits*, then we could apply forward transition rules to apply U again. Here, we modify the construction to allow this. We proceed by adding a new set of local basis states to the program register Hilbert

states and transition rules that will allow the program sequence to reset itself without undoing the application of U. We will refer to the construction described in this section as 'construction II'.

Hilbert Space

The following changes are made to the local program register Hilbert space: we add the new basis states represented by symbols $\{\overleftarrow{W}, \overleftarrow{S}, \overleftarrow{I}, \triangleright, \leftarrow, \circlearrowleft\}$. The length of the chain is increased by two sites.

Initial State

We must now define a new initial state to account for the fact that we have added two new sites. We will illustrate with the same example U being applied to three work qubits. The example initial state for our new construction will be

$$|\psi_0\rangle = \left[\begin{array}{cccccccccccccccc} \circlearrowleft & \rightarrow & S & W & I & I & W & S & I & . & . & . & . & . & . & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & w_1 & w_2 & w_3 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]. \tag{1.4.1}$$

The rules for constructing this state are identical to those for the base construction except the two ends of the chain have extra sites added with the program register in state $\circlearrowleft$ and data register in state 0.

Transition Rules

We add the following new forward transition rules:

$$7: \boxed{A}\boxed{\leftarrow} \rightarrow \boxed{\overleftarrow{A}}\boxed{.} \tag{1.4.2}$$

$$8: \boxed{B}\boxed{\overleftarrow{A}} \rightarrow \boxed{\overleftarrow{B}}\boxed{A} \tag{1.4.3}$$

$$9: \boxed{.}\boxed{\overleftarrow{A}} \rightarrow \boxed{\leftarrow}\boxed{A} \tag{1.4.4}$$

$$10: \boxed{.}\boxed{\leftarrow} \rightarrow \boxed{.}\boxed{\triangleright} \tag{1.4.5}$$

$$11: \boxed{\triangleright}\boxed{A} \rightarrow \boxed{A}\boxed{\triangleright} \tag{1.4.6}$$

$$12: \boxed{\triangleright}\boxed{.} \rightarrow \boxed{\leftarrow}\boxed{.} \tag{1.4.7}$$

$$\begin{aligned} 13a: \boxed{\rightarrow}\boxed{\circlearrowleft} &\rightarrow \boxed{\leftarrow}\boxed{\circlearrowleft} \\ 13b: \boxed{\circlearrowleft}\boxed{\leftarrow} &\rightarrow \boxed{\circlearrowleft}\boxed{\rightarrow} \end{aligned} \tag{1.4.8}$$

The forward transition rules 6-12 are designed to mimic the reverse versions of rules 1-6 using the newly introduced basis symbols. Rule 13 shows conversions between the $\rightarrow$ and $\leftarrow$ symbols via the $\circlearrowleft$ symbol, called the turn symbol.

Illustration

We now show how the additions made to our construction allow us to repeatedly apply U multiple times. Starting with our new start state

$$|\psi_0\rangle = \left[\begin{array}{cccccccccccccccc} \circlearrowleft & \rightarrow & S & W & I & I & W & S & I & . & . & . & . & . & . & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & w_1 & w_2 & w_3 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right], \tag{1.4.9}$$

the same sequence of forward transitions $\mathcal{T}_I$ that took Eq. (1.3.9) to Eq. (1.3.19) yields

$$|\psi_{T_I}\rangle = \left[\begin{array}{cccccccccccccccc} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \rightarrow & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U(w_1w_2w_3) & & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]. \tag{1.4.10}$$

Now, whereas in the previous construction, no forward transition rule could apply to $|\psi_{T_I}\rangle$, here there is one forward transition rule that applies to $|\psi_{T_I}\rangle$: rule 13a, which brings us to state $|\psi_{T_I+1}\rangle$:

$$|\psi_{T_I+1}\rangle = \left[\begin{array}{cccccccccccccccc} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \leftarrow & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U(w_1w_2w_3) & & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]. \tag{1.4.11}$$

Now from state $|\psi_{T_I+1}\rangle$ there is a unique set of T_I forward transitions through mutually orthogonal states that brings us to state $|\psi_{2T_I+1}\rangle$

$$|\psi_{2T_I+1}\rangle = \left[\begin{array}{cccccccccccccccc} \circlearrowleft & \leftarrow & S & W & I & I & W & S & I & . & . & . & . & . & . & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U(w_1w_2w_3) & & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]. \tag{1.4.12}$$

The set of transitions $\mathcal{T}_I'$ that does this is chosen from rules 7-12. Note that each rule 7-12 is similar to the reverse of one of the rules 1-6. To obtain the set of forward transitions that takes $|\psi_{T_I+1}\rangle \rightarrow |\psi_{2T_I+1}\rangle$, take the set of forward transitions that took $|\psi_0\rangle \rightarrow |\psi_{T_I}\rangle$, $\mathcal{T}_I$, reverse it, and replace each transition from 1-6 with its partner from 7-12. Basically, this is undoing everything that happened in the transitions that took $|\psi_0\rangle \rightarrow |\psi_{T_I}\rangle$ but not undoing the application of gates to the work qubits, because there are no apply gate symbols in the transitions 7-12. It is clear, then that the sequence of states $\left\{|\psi_{T_I+1+t}\rangle\right\}_{t=0}^{T_I}$ is UOG by $\mathcal{T}_I'$ for the same reason that $\{|\psi_t\rangle\}_{t=0}^{T_I}$ is UOG by $\mathcal{T}_I$. From state $|\psi_{2T_I+1}\rangle$, there is only one forward transition rule that applies: 13b, which brings us to

$$|\psi_{2T_I+2}\rangle = \left[\begin{array}{cccccccccccccccc} \circlearrowleft & \rightarrow & S & W & I & I & W & S & I & . & . & . & . & . & . & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U(w_1w_2w_3) & & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]. \tag{1.4.13}$$

Now, we see that the state is identical in structure to the initial state $|\psi_0\rangle$, but U has already been applied to the work qubits. So, applying the same set of transitions that took $|\psi_0\rangle \rightarrow |\psi_{2T_I+2}\rangle$, we should get

$$|\psi_{4T_I+4}\rangle = \left[\begin{array}{cccccccccccccccc} \circlearrowleft & \rightarrow & S & W & I & I & W & S & I & . & . & . & . & . & . & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^2(w_1w_2w_3) & & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]. \tag{1.4.14}$$

This set of forward transitions can be able to be applied over and over again, which will apply U to the work qubits again and again. That is, applying this set of transitions x times should yield the state

$$|\psi_{x(2T_I+2)}\rangle = \left[\begin{array}{cccccccccccccccc} \circlearrowleft & \rightarrow & S & W & I & I & W & S & I & . & . & . & . & . & . & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^x(w_1w_2w_3) & & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]. \tag{1.4.15}$$

Active Site

In the sequence $\mathcal{T}_I'$ which resets the gate sequence, the sequence of right moving oscillations that occurred from the sequence $\mathcal{T}_I$ play out in reverse, but with the active symbol pointing in the opposite direction this time. Seeing one of the symbols $\mathcal{A}_R = \{\rightarrow, \overrightarrow{W}, \overrightarrow{I}, \overrightarrow{S}, \blacktriangleleft, \triangleleft\}$ at the active site in a state indicates that the system is in the process of applying right moving oscillations that shift the gate sequence to the right, applying the gate sequence to the work qubits when appropriate as it goes, whereas seeing one of the symbols $\mathcal{A}_L = \{\leftarrow, \overleftarrow{W}, \overleftarrow{I}, \overleftarrow{S}, \triangleright\}$ indicates that the system is in the process of applying left moving oscillations that shift the gate sequence to the left and not applying any gates to the work qubits.

Conclusion: There is A Catch

We see that we have built a sequence of states $\{|\psi_t\rangle\}_{t=0}^{T_{II}=2T_I+2}$, for which each state has a unique forward transition described by the sequence of transitions $\mathcal{T}_{II} = \{\mathcal{T}_I, 13a, \mathcal{T}_I', 13b\}$, and for which $|\psi_{T_{II}}\rangle$ is identical to $|\psi_0\rangle$ except that U has been applied to the work qubits. We see, then, that by applying the set of transitions $\mathcal{T}_{II}$ repeatedly, the work qubits have U applied to them repeatedly.

The above conclusion would tempt one to say that by building a new Hamiltonian H_{II} analogous to H_I but with the new additional transition rules, we again have a system whose time evolution is analogous to a 1D single particle quantum walk where the farther the particle walks from the starting point of the chain $|\psi_0\rangle$, the more times U will be applied to the work qubits. This, however, is incorrect. For the states $\{|\psi_t\rangle\}_{t=0}^{xT_{II}}$ to map to adjacent position eigenstates of a particle on a chain to, the states must be UOG by $\mathcal{T}_{II}^x$. However, they are not. The only difference in state $|\psi_t\rangle$ and state $|\psi_{t+kT_{II}}\rangle$ is the number of times that U has been applied to the work qubits. For any quantum state $|\phi\rangle$, $|\phi\rangle$ and $U^k|\phi\rangle$ will not generally be orthogonal.

As a result, if we time evolve the state $|\psi_0\rangle$ with H_{II}, states that have U applied to the work qubits a different number of times can interfere with each other, and can not be distinguished by a simple measurement. There is no clear way to do a measurement to collapse the time evolved state into one where U has been applied a certain number of times. We will remedy this problem by further augmenting our construction to include a binary clock that 'counts' the number of times that U has been applied and orthogonalizes the states which have had U applied a different number of times.

1.5 Construction III: Binary Clock Construction

We would like to modify Construction II so that by applying forward transition rules, all of the uniquely obtained states of the chain at each step, $|\psi_t\rangle$, are mutually orthogonal, and as a particular consequence, states of the chain where the work qubits have had U applied a different number of times will be orthogonal, and can be distinguished by doing a computational basis state measurement on a subset of the

chain that excludes the work qubits. We will accomplish this by further modifying the construction in the previous section. First, we give an overview of how the clock itself should work, then we describe how to implement its control and integration into our existing construction, and then we will prove its correctness. Similar binary clock constructions appear in [8, 34, 51].

Hilbert Space

The clock, itself, will consist of two registers/layers: the clock register (denoted C), whose local sites C_j are four dimensional with basis $\{\ \boldsymbol{.}, X, 0, 1\}$, and the clock pointer register (denoted CP), whose local sites CP_j are five dimensional with basis states $\{\ \boldsymbol{.}, X, L, R, C\}$. So we add these two new registers to our system (C being the third layer, CP being the fourth). We will also augment the program register's local Hilbert space basis to include the symbol $\Downarrow\!\!\Downarrow$.

So, our complete local Hilbert space description is as follows: There are $L = (2K-1)(N+1)+4$ local sites on the chain, each site j consisting of four different local Hilbert spaces, one from each register:

- The program register's local sites P_j have basis $\{W, S, I, \overrightarrow{W}, \overrightarrow{S}, \overrightarrow{I}, \blacktriangleleft, \triangleleft, \boldsymbol{.}, \rightarrow, \circlearrowleft, \overleftarrow{I}, \overleftarrow{W}, \overleftarrow{S}, \triangleright, \leftarrow, \Downarrow\!\!\Downarrow\}$
- The data register's local sites D_j have basis $\{0, 1\}$
- The clock register's local sites C_j have basis $\{\ \boldsymbol{.}, 0, 1\}$
- The clock pointer register's local sites CP_j have basis $\{\ \boldsymbol{.}, X, L, R, C\}$

The total local site dimension is $16 \times 2 \times 3 \times 5 = 480$.

Initial State

The initial state for this augmented construction is as follows:

$$|\psi_0\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & w_1 & w_2 & w_3 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ . & R & . & . & . & . & . & . & . & . & . & . & . & . & . & . \end{bmatrix}, \tag{1.5.1}$$

Here, the third layer is the clock register and the fourth and final layer is the clock pointer register. Notice that the program register has a different initial configuration than the previous two constructions: this time, the gate sequence starts shifted all the way to the right, and the clock and clock pointers are specifically initialized so that no reverse transition rule applies to the initial state. This will be illustrated below.

Clock Schematic and Update Algorithm

Here we describe a simple algorithm for updating a binary counter which will be implemented in our HQCA transition rules. The idea is the following: The clock register, consisting of L sites, can store a binary number of up to $L-1$ digits long. Numbers with $n < L$ significant digits will be stored in the n rightmost sites of the clock register via the 0 and 1 states, and the remaining $L-n$ sites will all be in the state 0 except for the first register which will always be $\cdot$. Significant bits will increase from right to left, so, for example, the binary representation of the number 6, in our clock, will look like $\cdot 0...00000110$.

Given any binary number represented in this form, we can use the following schematic algorithm for incrementing it by 1:

- A) Initialize a 'pointer' which starts sitting underneath the least significant bit.
- B) If the least significant bit is 0, change it to a 1, end
- C) If the least significant bit is 1, and the next bit is 0, update both bits $10 \rightarrow 01$, end
- D) Otherwise, the first two bits are 11. In this case, move the pointer to the left, bit by bit, until it encounters a 1 bit whose next most significant bit is a 0. When it is found, update these two bits $01 \rightarrow 10$. Move the pointer back to the right, bit by bit, flipping all of the less significant 1 bits to 0 as it passes, until it is back under the least significant bit, end.
- E) If the number being incremented is $\cdot 11...1$, then the pointer halts when it reaches the most significant 1, as this number can not be incremented further.

In our construction, the CP register is where the pointer that scans through and increments the clock registers will live. Generally, when the clock is not being incremented, the CP register will be in the state $\cdot\ \cdot\ ...\ \cdot\ X$. This indicates that the 'pointer' is underneath the least significant bit of the clock, and is inactive, X. The $\cdot$ symbol underneath all of the other bits simply indicates the absence of the clock pointer. When the clock pointer is scanning to the left, looking for a 1 bit whose next most significant bit is a 0, it will be in state L and will hop right past the $\cdot$ symbols until it encounters this configuration in the clock register. Then it updates the clock bits according to the algorithm above, the pointer in state L transitions to a pointer in state R, and will now start scanning back to the right, flipping the less significant 1 bits it encounters along the way to 0 bits. Once it reaches the least significant bit, the clock pointer transitions to the C state to indicate that the clock update is complete, and then back to the X state, in which it will stay until it is once again called upon to increment the clock.

Transition Rules

Now we give the transition rules which implement the control of the clock as described above. These rules include a modification to rule 13:

$13a$: $\begin{array}{|c|c|}\hline \rightarrow & \circlearrowleft \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \circlearrowleft & \Downarrow \\ \hline \end{array}$ $13b$: $\begin{array}{|c|c|}\hline \circlearrowleft & \leftarrow \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \circlearrowleft & \rightarrow \\ \hline \end{array}$ (1.5.2)

14 : $\begin{array}{|c|c|}\hline \circlearrowleft & \Downarrow \\ \hline - & - \\ \hline - & - \\ \hline \mathbf{.} & X \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline \mathbf{.} & L \\ \hline \end{array}$ (1.5.3)

15 : $\begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & 0 \\ \hline \mathbf{.} & L \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & 1 \\ \hline \mathbf{.} & C \\ \hline \end{array}$ (1.5.4)

16 : $\begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline 0 & 1 \\ \hline \mathbf{.} & L \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline 1 & 0 \\ \hline \mathbf{.} & C \\ \hline \end{array}$ (1.5.5)

17 : $\begin{array}{|c|c|}\hline - & - \\ \hline - & - \\ \hline 1 & 1 \\ \hline \mathbf{.} & L \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline - & - \\ \hline - & - \\ \hline 1 & 1 \\ \hline L & \mathbf{.} \\ \hline \end{array}$ (1.5.6)

18 : $\begin{array}{|c|c|}\hline \mathit{not}\ \circlearrowleft & - \\ \hline - & - \\ \hline 0 & 1 \\ \hline \mathbf{.} & L \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \mathit{not}\ \circlearrowleft & - \\ \hline - & - \\ \hline 1 & 0 \\ \hline \mathbf{.} & R \\ \hline \end{array}$ (1.5.7)

19 : $\begin{array}{|c|c|}\hline \mathit{not}\ \circlearrowleft & - \\ \hline - & - \\ \hline 0 & 1 \\ \hline R & \mathbf{.} \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \mathit{not}\ \circlearrowleft & - \\ \hline - & - \\ \hline 0 & 0 \\ \hline \mathbf{.} & R \\ \hline \end{array}$ (1.5.8)

20 : $\begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline 0 & 1 \\ \hline R & \mathbf{.} \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline 0 & 0 \\ \hline \mathbf{.} & C \\ \hline \end{array}$ (1.5.9)

$$21: \quad \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline . & C \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \leftarrow & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline . & X \\ \hline \end{array} \tag{1.5.10}$$

Active Site

In analyzing the correctness of the above transition rules for achieving our goals, it will again be useful to make use of the concept of the active site and active symbol. In our augmented construction, there will again only ever be exactly one active symbol in the state, which will indicate the active site. But this time, the active symbol can either be in the program register *or* the clock pointer register, depending on whether or not the clock is being updated. The set of active symbols is now the union of the active symbols for the P and CP registers: $\mathcal{A} = \mathcal{A}_P \cup \mathcal{A}_{CP}$, with $\mathcal{A}_P = \{\leftarrow, \rightarrow, \overleftarrow{W}, \overleftarrow{S}, \overleftarrow{I}, \overrightarrow{W}, \overrightarrow{S}, \overrightarrow{I}, \triangleleft, \blacktriangleleft, \triangleright, \downdownarrows\}$ and $\mathcal{A}_{CP} = \{L, R, C\}$. Transition rules 14 and 21 describe the passing of the active symbol between the program and clock pointer registers. It is still true that every single transition rule describes the moving of an active symbol from one site to a neighbouring site and/or the conversion of one kind of active symbol into another, which is essential for our sequence of states to be UOG and for the dynamics of the HQCA to map to a quantum walk on a line.

Correctness

We claim the following: starting from the initial state

$$|\psi_0\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & w_1 & w_2 & w_3 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ . & R & . & . & . & . & . & . & . & . & . & . & . & . & . & . \end{bmatrix}, \tag{1.5.11}$$

and applying forward transition rules 1-21, one generates a unique and finite sequence of mutually orthogonal states $\{|\psi_t\rangle\}_{t\geqslant 0}$, whose states on all registers across all layers aside from the work qubits are orthogonal product states. Moreover,

A) The final state is

$$|\psi_F\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^k & (w_1 w_2 w_3) & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ . & L & . & . & . & . & . & . & . & . & . & . & . & . & . & . \end{bmatrix}, \tag{1.5.12}$$

where U has been applied to the work qubits $k = 2^{L-1}$ times, where $L = (2K-1)(N+1)+4$ is the number of sites in the chain.

B) Suppose you define a Hamiltonian, H_{III}, out of terms that implement transition rules 1-21 and their Hermitian conjugates, and use this Hamiltonian to

continuously time evolve the state $|\psi_0\rangle$. If, at any point in the time evolution, one measures the clock layer to hold the binary number k, and the first site of the CP layer to hold the state C, then the post measurement state of the work qubits is guaranteed to be in the state $U^k|w_1w_2w_3\rangle$.

I will separate the proof into several small pieces.

Lemma 1 1. *Starting from the state*

$$|S_0^{U,x}\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \leftarrow & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^x & (w_1w_2w_3) & 1 & 0 & 0 & 0 & 1 & 0 & 0 & \\ . & & & & & & & \multicolumn{8}{c}{binary\ x} & & & \\ . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & X \end{bmatrix}. \quad (1.5.13)$$

there is a sequence of states $S^{U,x} = \left\{|S_t^{U,x}\rangle\right\}_{t=0}^{n_U}$ *UOG by sequence of* n_U *forward transitions* $\mathcal{U}$ *and whose final state is*

$$|S_{n_U}^{U,x}\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \rightarrow & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^{x+1} & (w_1w_2w_3) & 1 & 0 & 0 & 0 & 1 & 0 & 0 & \\ . & & & & & & & \multicolumn{8}{c}{binary\ x} & & & \\ . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & X \end{bmatrix}. \quad (1.5.14)$$

for any $x \in \{0, 1, \cdots, 2^{L-1} - 1\}$.

Proof. By the correctness of Construction II, the sequence of forward transitions $\mathcal{U} = \{\mathcal{T}_I', 13b, \mathcal{T}_I\}$ fulfills the requirements. □

The sequence of forward transitions $\mathcal{U}$ described above will referred to as an *application transition* since it generates a single application of U to the work qubits, and $n_U = T_{II} - 1$ is the number of transitions in $\mathcal{U}$. The sequence of mutually orthogonal states $S^{U,x} = \left\{|S_t^{U,x}\rangle\right\}_{t=0}^{n_U}$ is called the xth application sequence.

Lemma 2 1. *Starting with a state of the form*

$$|S_0^{C,x}\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^{x+1} & (w_1w_2w_3) & 1 & 0 & 0 & 0 & 1 & 0 & 0 & \\ . & & & & & & & \multicolumn{8}{c}{binary\ x} & & & \\ . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & L \end{bmatrix}, \quad (1.5.15)$$

the sequence of states $S^{C,x} = \left\{|S_t^{C,x}\rangle\right\}_{t=0}^{n_{C,x}}$ *is UOG by sequence of forward transitions* $C\mathcal{U}_x$*, with final state*

$$|S_{n_{C,x}}^{C,x}\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^{x+1} & (w_1w_2w_3) & 1 & 0 & 0 & 0 & 1 & 0 & 0 & \\ . & & & & & & & \multicolumn{8}{c}{binary\ x+1} & & & \\ . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & C \end{bmatrix}. \quad (1.5.16)$$

for all $x \in \{0, 1, \cdots, 2^{L-1}\}$.

Proof. Once a state of the form $|S_0^{C,x}\rangle$ is reached, there is no longer any active symbol in the program register, and the active symbol is now in the CP register.

First, observe that no matter what number the clock layer holds, there will only ever be one of four possibly valid forward transitions from a state of the form $|S_0^{C,k}\rangle$, which will be either rule 15, 16, or 17 depending on the values of the clock's least two significant bits. Referring to the clock update algorithm described earlier, rule 15 corresponds to situation B), rule 16 corresponds to situation C), and rule 17 corresponds to situation D).

If the least significant bit of the clock in state $|S_0^{C,x}\rangle$ is zero (least two significant bits are 10 or 00), then only rule 15 can be applied, resulting in state

$$|S_{n_{C,x}}^{C,x}\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^{x+1} & (w_1 w_2 w_3) & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & & & & & & & binary\, x+1 & & & & & & & & \\ . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & C \end{bmatrix}. \quad (1.5.17)$$

so we're done. (By examining transition rule 15, it's obvious that in any of these situations, applying the single transition rule correctly increments the clock's stored number, as per case *B* of the clock update algorithm described earlier.)

If the two least significant bits of the clock in state $|S_0^{C,x}\rangle$ are 01, then only rule 16 can be applied, resulting in state

$$|S_{n_{C,x}}^{C,x}\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^{x+1} & (w_1 w_2 w_3) & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & & & & & & & binary\, x+1 & & & & & & & & \\ . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & C \end{bmatrix}. \quad (1.5.18)$$

so we're done. (Again, by examining transition rule 16, it's obvious that in any of these situations, applying the single transition rule correctly increments the clock's stored number, as per case *C* of the clock update algorithm described earlier.)

If rules 15 and 16 don't apply to $|S_0^{C,x}\rangle$, then let $2 \leqslant p \leqslant \lceil \log_2(x) \rceil$ be the number of least significant bits of x that form an uninterrupted string of 1s. Then it is clear that there are $p-1$ unique forward transitions through $p-1$ applications of rule 17. Because $x < 2^{L-1}$, we know that $p < L-1$. Then, after the $p-1$ applications of rule 17, registers p and $p+1$ of the chain must fit the description of the initial state of rule 18, in which case rules 15-17 can not apply (after the first application of 17, the program registers won't have the $\circlearrowleft$'s needed in the program register to allow 15 or 16 to apply), therefore only rule 18 can be applied because it is the only remaining transition rule whose initial state has L in the clock pointer.

After rule 18 is applied, the clock pointer's active site is in R mode, so only rules 19 or 20 could potentially be applied. If $p = 2$, then only rule 20 can be applied because the existence of the $\circlearrowleft$ symbol in the program register above the R symbol will forbid application of rule 19. If $p > 2$, then after application of rule 18, we see

that bits p and $p-1$ of the number stored in the clock (counting from the right) are exactly the requisite initial state for applying rule 19, so 19 is the only way forward. This will obviously be true for $p-2$ applications of rule 19, at which point the R symbol will have the $\circlearrowleft$ symbol in the program register above it, so only rule 20 will be able to be applied. In either case, at this point we will have implemented case E) of the clock increase algorithm, and will be in state $|S^{C,x}_{n_{C,x}}\rangle$ as above. Note that the state of the program and data layers never changed at any point in this sequence because we only used rules 15 through 20, and they leave those layers invariant.

By having described all possibilities, we see that the sequence $S^{C,x}$ is UOG by the appropriate sequence of forward transitions $C\mathcal{U}_x$. □

The sequence of forward transitions, $C\mathcal{U}_x$, that takes $|S^{C,x}_0\rangle \rightarrow |S^{C,x}_{n_{C,x}}\rangle$ is called a *clock transition*, and the sequence of mutually orthogonal states $S^{C,x} = S^{C,x} = \left\{|S^{C,x}_t\rangle\right\}_{t=0}^{n_{C,x}}$ is called the xth clock sequence.

Lemma 3 1. *The states in sequences $S^{C,k}$ and $S^{C,j}$ are mutually orthogonal for $j \neq k$.*

Proof. Suppose that there were states $|\phi_j\rangle \in S^{C,j}$ and $|\phi_k\rangle \in S^{C,k}$ which are not orthogonal. This necessarily means that the configurations of the clock and clock pointer registers in both states are identical. This, then, implies that it is possible to find a sequence of forward transitions to apply to $|\phi_j\rangle$ that brings it to $|S^{C,k}_{n_{C,k}}\rangle$. But, then, this contradicts Lemma 2 which states that the clock update sequence $S^{C,k}$ with is UOG, as this would imply that there are at least two distinct forward transition rules that could be applied to $|\phi_j\rangle$. □

Lemma 4 1. *The states in sequences $S^{P,k}$ and $S^{P,j}$ are mutually orthogonal for $j \neq k$.*

Proof. This is obvious because the clocks will hold different numbers in each sequence (j and k respectively). □

Lemma 5 1. *Starting from the state*

$$|S^{C,2^{L-1}-1}_0\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^{2^{L-1}} & (w_1w_2w_3) & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & L \end{bmatrix} \tag{1.5.19}$$

there is a unique sequence of $L-2$ forward transitions which maps us to the final state

$$|S^{C,2^{L-1}-1}_0\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^{2^{L-1}} & (w_1w_2w_3) & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ . & L & . & . & . & . & . & . & . & . & . & . & . & . & . & . \end{bmatrix} \tag{1.5.20}$$

from which there are no possible forward transitions.

Proof. Clearly, applying rule 17 $L-2$ times does the job. □

Lemma 6 1. *Starting from the initial state*

$$|\psi_0\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & w_1 & w_2 & w_3 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ . & R & . & . & . & . & . & . & . & . & . & . & . & . & . & . \end{bmatrix}, \tag{1.5.21}$$

and applying forward transition rules 1-21, one generates the sequence $\{|\psi_t\rangle\}_{t=0}^{F}$ *which is UOG by some sequence of F transitions,* $\mathcal{S}$*, and whose final state is reaching the final state*

$$|\psi_F\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \circlearrowleft & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & U^{2^{L-1}} & (w_1 w_2 w_3) & & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ . & L & . & . & . & . & . & . & . & . & . & . & . & . & . & . \end{bmatrix}. \tag{1.5.22}$$

Proof. By lemmas 1 through 5, we see that the sequence of forward transitions of length F, $\mathcal{S} = \{\mathcal{L}_{-1}, \{\mathcal{L}_x\}_{x=0}^{2^{L-1}-1}, \mathcal{L}_F\}$, with subsequences defined

$$\mathcal{L}_{-1} = \{19^{L-3}, 20, 21\} \tag{1.5.23}$$

$$\mathcal{L}_x = \{\mathcal{U}, 13a, 14, \mathcal{UC}_x, 21\} \tag{1.5.24}$$

$$\mathcal{L}_F = \{\mathcal{U}, 13a, 14, 17^{L-2}\} \tag{1.5.25}$$

does the job. □

1.6 Construction IV: Deterministic Selection of x in time poly(x, N)

Now we augment construction III so that a particular number x of applications of U can be selected to be applied to the work qubits, and we can obtain the state where this has happened more or less deterministically by time evolving for a time polynomial in x and N. To do this, we build in a new register that holds the number of desired applications x in binary form, and once the number in the clock register reaches x, forward transitions are modified so that although the quantum walk continues forward, gates are no longer applied to data qubits, so there is some $T < F$ such that for all $|\psi_{t>T}\rangle$, the work qubits will always have U^x applied to them.

Hilbert Space

We add two new registers in this construction: the target register T, and the a second clock register C_2. The local sites of both registers are three dimensional $\{0, 1, \ .\ \}$. We also make the following additions of new states to the local sites of the other registers:

- The program register adds symbols $\{\overrightarrow{I}^{\times}, \overrightarrow{S}^{\times}, \overrightarrow{W}^{\times}, \triangleleft^{\times}, \rightarrow^{\times}, \overleftarrow{I}^{\times}, \overleftarrow{S}^{\times}, \overleftarrow{W}^{\times}, \triangleright^{\times}, \leftarrow^{\times}, \Downarrow^{\times}\}$

- The clock pointer register adds symbols $\{\overleftarrow{C}, CX, R^{\times}, C^{\times}, L^{\times}\}$

With these additions, the local dimensions of the program, data, clock, clock pointer, target and second clock pointer registers are now 27, 2, 3, 10, 3, and 3 respectively. The total local dimension of a site of the qudit chain C is thus $d = 27 \times 2 \times 3 \times 10 \times 3 \times 3 = 14580$.

Initial State

In the initial state we must now have initialization of the fifth and sixth layers, the target and second clock registers respectively. The target register is initialized to hold the binary representation of the target number of applications, x, with significant digits increasing from right to left, and all places greater than the most significant 1 of x being padded by . symbols. The second clock register is initialized the same way the clock register was initialized in the previous construction.

$$|\psi_0\rangle = \begin{bmatrix} \circlearrowleft & . & . & . & . & . & . & S & W & I & I & W & S & I & \leftarrow & \circlearrowleft \\ 0 & 1 & 0 & 0 & 0 & 1 & w_1 & w_2 & w_3 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ . & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & X \\ . & & & & & & & & \textit{binary } x & & & & & & & \\ . & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{1.6.1}$$

Transition Rules

Here are the added transition rules, including a modification of transition rule 21:

$$21: \quad \begin{array}{|c|c|} \hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline . & CX \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|} \hline \leftarrow & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline . & X \\ \hline \end{array} \tag{1.6.2}$$

$$22: \quad \begin{array}{|c|c|} \hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline . & C^{\times} \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|} \hline \leftarrow^{\times} & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline . & X \\ \hline \end{array} \tag{1.6.3}$$

$23a$:

$\circlearrowleft$	$\circlearrowleft$
–	–
not .	a
.	C
not .	a

$\rightarrow$

$\circlearrowleft$	$\circlearrowleft$
–	–
not .	a
$\overleftarrow{C}$	.
not .	a

(1.6.4)

$23b$:

$\circlearrowleft$	$\circlearrowleft$
–	–
not .	a
.	C
.	a

$\rightarrow$

$\circlearrowleft$	$\circlearrowleft$
–	–
not .	a
$\overleftarrow{C}$	.
.	a

$23c$:

$\circlearrowleft$	$\circlearrowleft$
–	–
.	a
.	C
not .	a

$\rightarrow$

$\circlearrowleft$	$\circlearrowleft$
–	–
.	a
$\overleftarrow{C}$	.
not .	a

$24a$:

not $\circlearrowleft$	–
–	–
not .	a
.	$\overleftarrow{C}$
not .	a

$\rightarrow$

not $\circlearrowleft$	–
–	–
not .	a
$\overleftarrow{C}$	.
not .	a

(1.6.5)

$24b$:

not $\circlearrowleft$	–
–	–
not .	a
.	$\overleftarrow{C}$
.	a

$\rightarrow$

not $\circlearrowleft$	–
–	–
not .	a
$\overleftarrow{C}$	.
.	a

$24c$:

not $\circlearrowleft$	–
–	–
.	a
.	$\overleftarrow{C}$
not .	a

$\rightarrow$

not $\circlearrowleft$	–
–	–
.	a
$\overleftarrow{C}$	.
not .	a

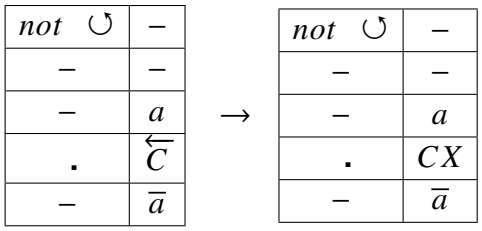
25 :

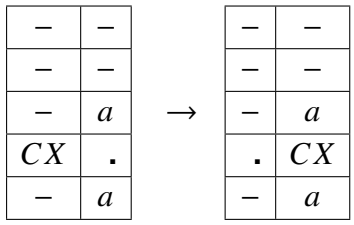
$\circlearrowleft$	–
–	–
–	a
.	C
–	$\overline{a}$

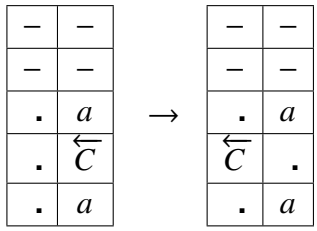
$\rightarrow$

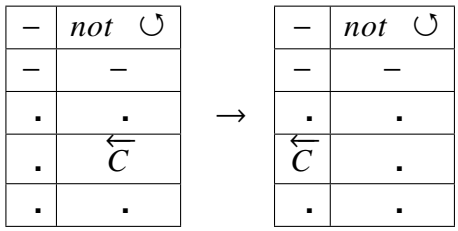
$\circlearrowleft$	–
–	–
–	a
.	CX
–	$\overline{a}$

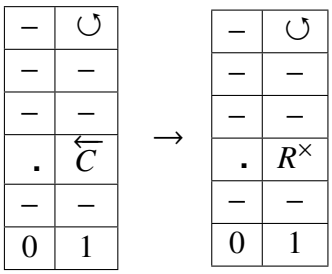
(1.6.6)

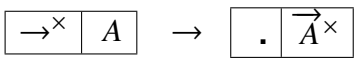
26 :

(1.6.7)

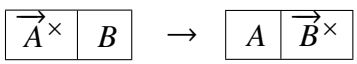
27 :

(1.6.8)

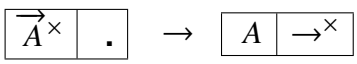
28 :

(1.6.9)

29 :

(1.6.10)

30 :

(1.6.11)

31 :

(1.6.12)

32 :

(1.6.13)

33 :

(1.6.14)

34 : $\boxed{\rightarrow^{\times}}\boxed{\cdot} \rightarrow \boxed{\triangleleft^{\times}}\boxed{\cdot}$ (1.6.15)

35 : $\boxed{A}\boxed{\triangleleft^{\times}} \rightarrow \boxed{\triangleleft^{\times}}\boxed{A}$ (1.6.16)

36 : $\boxed{\cdot}\boxed{\triangleleft^{\times}} \rightarrow \boxed{\cdot}\boxed{\rightarrow^{\times}}$ (1.6.17)

37 : $\boxed{A}\boxed{\leftarrow^{\times}} \rightarrow \boxed{\overleftarrow{A}^{\times}}\boxed{\cdot}$ (1.6.18)

38 : $\boxed{B}\boxed{\overleftarrow{A}^{\times}} \rightarrow \boxed{\overleftarrow{B}^{\times}}\boxed{A}$ (1.6.19)

39 : $\boxed{\cdot}\boxed{\overleftarrow{A}^{\times}} \rightarrow \boxed{\leftarrow^{\times}}\boxed{A}$ (1.6.20)

40 : $\boxed{\cdot}\boxed{\leftarrow^{\times}} \rightarrow \boxed{\cdot}\boxed{\triangleright^{\times}}$ (1.6.21)

41 : $\boxed{\triangleright^{\times}}\boxed{A} \rightarrow \boxed{A}\boxed{\triangleright^{\times}}$ (1.6.22)

42 : $\boxed{\triangleright^{\times}}\boxed{\cdot} \rightarrow \boxed{\leftarrow^{\times}}\boxed{\cdot}$ (1.6.23)

43a : $\boxed{\rightarrow^{\times}}\boxed{\circlearrowleft} \rightarrow \boxed{\circlearrowleft}\boxed{\Downarrow^{\times}}$

43b : $\boxed{\circlearrowleft}\boxed{\leftarrow^{\times}} \rightarrow \boxed{\circlearrowleft}\boxed{\rightarrow^{\times}}$ (1.6.24)

44 : $\begin{array}{|c|c|}\hline \circlearrowleft & \Downarrow^{\times} \\ \hline - & - \\ \hline - & - \\ \hline \cdot & X \\ \hline - & - \\ \hline - & - \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline \cdot & L^{\times} \\ \hline - & - \\ \hline - & - \\ \hline \end{array}$ (1.6.25)

45 : $\begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline \cdot & L^{\times} \\ \hline - & - \\ \hline - & 0 \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline \cdot & C^{\times} \\ \hline - & - \\ \hline - & 1 \\ \hline \end{array}$ (1.6.26)

$$46: \quad \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline \boldsymbol{.} & L^{\times} \\ \hline - & - \\ \hline 0 & 1 \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline \boldsymbol{.} & C^{\times} \\ \hline - & - \\ \hline 1 & 0 \\ \hline \end{array} \tag{1.6.27}$$

$$47: \quad \begin{array}{|c|c|}\hline - & - \\ \hline - & - \\ \hline - & - \\ \hline \boldsymbol{.} & L^{\times} \\ \hline - & - \\ \hline 1 & 1 \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline - & - \\ \hline - & - \\ \hline - & - \\ \hline L^{\times} & \boldsymbol{.} \\ \hline - & - \\ \hline 1 & 1 \\ \hline \end{array} \tag{1.6.28}$$

$$48: \quad \begin{array}{|c|c|}\hline \mathit{not} \ \circlearrowleft & - \\ \hline - & - \\ \hline - & - \\ \hline \boldsymbol{.} & L^{\times} \\ \hline - & - \\ \hline 0 & 1 \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \mathit{not} \ \circlearrowleft & - \\ \hline - & - \\ \hline - & - \\ \hline \boldsymbol{.} & R^{\times} \\ \hline - & - \\ \hline 1 & 0 \\ \hline \end{array} \tag{1.6.29}$$

$$49: \quad \begin{array}{|c|c|}\hline \mathit{not} \ \circlearrowleft & - \\ \hline - & - \\ \hline - & - \\ \hline R^{\times} & \boldsymbol{.} \\ \hline - & - \\ \hline 0 & 1 \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \mathit{not} \ \circlearrowleft & - \\ \hline - & - \\ \hline - & - \\ \hline \boldsymbol{.} & R^{\times} \\ \hline - & - \\ \hline 0 & 0 \\ \hline \end{array} \tag{1.6.30}$$

$$50: \quad \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline R^{\times} & \boldsymbol{.} \\ \hline - & - \\ \hline 0 & 1 \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|}\hline \circlearrowleft & \circlearrowleft \\ \hline - & - \\ \hline - & - \\ \hline \boldsymbol{.} & C^{\times} \\ \hline - & - \\ \hline 0 & 0 \\ \hline \end{array} \tag{1.6.31}$$

Whenever just a appears in a transition rule it must be one of $\{0, 1\}$. Whenever a and $\overline{a}$ appear, one of a or $\overline{a}$ must be in $\{0, 1\}$ and the remaining symbol is either the logical negation, or the $\boldsymbol{.}$ symbol.

Discussion

Here's a brief outline of how this modified construction with these new transition rules works:

- Transition rules 22-30 implement a new part of the clock update phase by giving it a new function at the end: when the clock pointer has finished updating the clock and is in C mode, it then begins a second sweep to the left in $\overleftarrow{C}$ mode, comparing the number stored in the clock to the number stored in the target register bit by bit
- If the clock pointer discovers bits that do not match, it turns into CX mode, moves back to the right, and the active symbol moves back up to the program layer to begin the next unitary application as normal in construction III
- If the clock pointer in $\overleftarrow{C}$ mode determines that the clock and target registers match, it continues moving left until it reaches the left end of the chain, and this causes the clock pointer to transition to $R^{\times}$ mode
- Once the the clock pointer has transitioned to $R^{\times}$ mode, all future active symbols will have the $\times$ symbol attached and obey transition rules 31-50, which identically mimic transition rules 1-20 of construction III, but are modified slightly so that a unitary is never applied to the work qubits ever again (the presence of the $\times$ symbol on all active symbols indicates this), and now the clock pointer (operating in an identical fashion to how it did in construction III, without a phase where it compares to a target register) treats the the second clock register $C2$ as the clock register.

At this point, based on our in depth analysis of construction III, it is easy to check that the sequence of states generated by forward transition rules $1 - 50$ is UOG by the appropriate sequence of unique forward transitions. The outcome, then, is that the Hamiltonian H_{IV} based on transition rules 1-50 implements a quantum walk on a line whose length is $l = \exp(N, K)$ (the length of the line for constructions H_{III} and H_{IV} is exponential because the clock can count up to a time $\exp(N, K)$). Once the quantum walk has proceeded far enough for U to have been applied to the work qubits x times (this distance is $\text{poly}(x, N, K)$), the quantum walk can continue forward for $\exp(N, K)$ more steps without ever changing the state of the data and clock registers ever again. We complete the analysis of the construction below by proving that we only need to run the simulation for a time $\text{poly}(x, N, K)$ in order to be able to do a simple measurement on the chain that will collapse the state of the work qubits to one where U^x has been applied with high probability.

Run-Time Analysis

Here we prove that after running construction IV with an appropriate input state for a time $\tau = \text{poly}(x, N, K)$, a computational basis measurement on the clock register will return the result that x is stored in the clock register (and therefore U^x has been applied to the work qubits) with high probability.

This is accomplished by slightly modifying a lemma from [72]:

Lemma 7 1. *Consider a continuous-time quantum walk on a line of length l, where the Hamiltonian is the negative of the adjacency matrix for the line. Let the system evolve for a time $\tau \leqslant \tau^*$ chosen uniformly at random, starting from position basis state $|0\rangle$ (one end of the line). The probability to measure a state $|m\rangle$ with $m > (1-F)l$, $F \in [0,1]$, is then bounded from below as $p^* \geqslant F - O\left(\frac{l}{\tau^*}\right) - O\left(\frac{1}{l}\right)$.*

Proof. Let $p_\tau(m)$ be the probability of measuring state $|m\rangle$ at time τ in the above scenario. Then the time average of $p_\tau(m)$ for $0 \leqslant \tau \leqslant \tau^*$ is

$$\overline{p}_{\tau^*}(m) = \frac{1}{\tau^*} \int_0^{\tau^*} p_\tau(m) d\tau \tag{1.6.32}$$

which, in the limit $\tau^* \to \infty$, converges to the limiting distribution [72]

$$\pi(m) = \frac{2 + \delta_{m,0} + \delta_{m,l-1}}{2(l+1)}, \tag{1.6.33}$$

in the sense that

$$\sum_{m=0}^{l-1} |\overline{p}_{\tau^*}(m) - \pi(m)| \leqslant O\left(\frac{l}{\tau^*}\right). \tag{1.6.34}$$

Then, the probability of measuring state $|m\rangle$ for $m > (1-F)l$ for time $\tau < \tau^*$ chosen uniformly at random is

$$p^* = \sum_{m > (1-F)l} \overline{p}_{\tau^*}(m). \tag{1.6.35}$$

We have that

$$\sum_{m=0}^{l-1} |\overline{p}_{\tau^*}(m) - \pi(m)| \tag{1.6.36}$$

$$\geqslant \sum_{m > (1-F)l} |\overline{p}_{\tau^*}(m) - \pi(m)| \tag{1.6.37}$$

$$\geqslant \left| \sum_{m > (1-F)l} \left(\overline{p}_{\tau^*}(m) - \pi(m)\right) \right| \tag{1.6.38}$$

$$= \left| p^* - \sum_{m>(1-F)l} \frac{2 + \delta_{m,0} + \delta_{m,l-1}}{2(l+1)} \right| \tag{1.6.39}$$

$$= \left| p^* - F + \frac{F}{l+1} + \frac{1}{2(l+1)} \right| \tag{1.6.40}$$

$$\geqslant -p^* + F - \frac{F}{l+1} - \frac{1}{2(l+1)} \tag{1.6.41}$$

But, $O\left(\frac{l}{\tau^*}\right) \geqslant \sum_{m=0}^{l-1} |\overline{p}_{\tau^*}(m) - \pi(m)|$, thus, we have that

$$p^* \geqslant F - O\left(\frac{l}{\tau^*}\right) - O\left(\frac{1}{l}\right), \tag{1.6.42}$$

as desired. □

We now consider the consequences of this lemma for the run-time of our simulation. The length of the chain that the states $|\psi_t\rangle$ of the simulator map onto position eigenstates $|t\rangle$ of is $l = \exp(N, K)$. For $x = \exp(N)$, parameters can easily be chosen such that the fraction F of l such that $|\psi_{t>(1-F)l}\rangle$ has had U^x applied to the work qubits is $1 - \exp(-N)$. Then by setting $\tau^* = \text{poly}(l)$ appropriately, and picking a uniformly random time between 0 and τ^*, we will succeed in applying U^x to the work qubits with probability $p^* \geqslant 1 - 1/\exp(\text{poly}(N, K))$. For all states $|\psi_{t>(1-F)l}\rangle$, which have had U^x applied to the work qubits, the clock register will hold the number x which was originally set in the target register. So we only need to do a computational basis measurement on the clock register and find it in the state holding the binary number x to know that we've succeeded.

Note that if we would like to simulate the application of unitary U^x for $x = \text{poly}(N)$, the above strategy does not actually seem very helpful: because $l = \exp(\text{poly}(N, K))$ is exponential, we need to wait for, on average, exponential time to guarantee high success probability even though we're simulating something that should only take polynomial time. This can easily be fixed. By modifying the start state of the simulator, $|\psi_0\rangle$, so that the . symbol in the target register is far enough to the right, one can exponentially reduce l. This is because once the target register starts functioning as a clock (after U^x has been applied), the length of the remaining distance the particle can travel is exponential in the number of sites to the right of the . symbol in the target register. The end state is not reached until the target register has counted up to the largest number that it can. So, one can place the . at the appropriate site (say, 3 sites to the left of the most significant digit of the number x stored in the target register) so that l will be polynomial in N, but will also be large enough relative to x so that one only needs τ^* to be polynomial to achieve a high rate of success. Note, however, that in this scenario, the probability of success will only be inverse polynomially close to unity rather than exponentially.

Finally, we mention that from the start state defined in equation (76), it is actually possible to apply reverse transitions to this start state which allows you to start running a 'clock check and update' phase in reverse using the reverse of transition rules 21, 25 and 20 (if the target register's least significant bit is 1), or rules 21, 27, 23, 24 and 20 (if the target register's least significant bit is 0). However, it is easy to check that in either case, only one unique sequence of no more than $3L$ reverse transitions can be applied to this state before reaching a state from which no further reverse transitions can be made. Thus the effect on run-time analysis is insignificant.

1.7 Acknowledgements

We thank Dorit Aharonov for interesting discussions, Elizabeth Crosson for helpful discussions and suggestions, and Toby Cubitt for helpful comments on our first draft that lead us to correct several mistakes in our construction. Author T.C.B. acknowledges financial support from the National Science and Engineering Research Council of Canada (NSERC) in the form of a Postgraduate Scholarship (PGS-D) award during the time in which this work was completed.

Chapter 2

GOOD APPROXIMATE QUANTUM LDPC CODES FROM SPACETIME CIRCUIT HAMILTONIANS

2.1 Introduction

A central result in the theory of classical error correcting codes is that there exist families of *good* linear $[N, k, d]$ codes, which have linear dimension $k = \Omega(N)$, linear distance $d = \Omega(N)$, constant sparsity parity checks, and linear time encoding and decoding algorithms. These *low-density parity check* (LDPC) codes [46] have many theoretical as well as practical applications.

A grand challenge in quantum information theory is to construct a *quantum* counterpart to classical LDPC codes with similarly optimal parameters. Traditionally this effort has focused on CSS stabilizer codes[1], where the notion of sparse parity checks corresponds to stabilizer generators that each act on $O(1)$ physical qubits, with each qubit participating in only $O(1)$ of such checks. The existence of QLDPC codes with good parameters and fast encoding/decoding algorithms would have significant practical impact; for example, Gottesman has shown these would imply schemes for fault tolerant quantum computation with constant overhead [50].

Despite many years of investigation, we do not yet know of QLDPC codes that simultaneously achieve constant rate and relative distance while maintaining constant locality and sparsity. The QLDPC codes of [67, 76] have a constant rate, but the minimum distance does not exceed $O(\sqrt{N})$ where N is the number of physical qubits. So far the QLDPC code with the best distance scaling is the construction of Freedman, Meyers and Luo [45] which achieves minimum distance distance $O(\sqrt{N \log N})$, but only encodes a single qubit. Bravyi and Hastings gave a probabilistic construction of a code with constant rate and linear distance, but the stabilizer generators each act on $\sqrt{N}$ physical qubits [19]. Hastings proved that, assuming a conjecture about high dimensional geometry, there exist QLDPC codes encoding a constant number of qubits (i.e. have vanishing rate) with distance scaling as $\Omega(N^{1-\xi})$ for any $\xi > 0$ [52, 54].

The question of whether good QLDPC codes exist also has importance for Hamiltonian complexity and the construction of exotic models in physics. This connection arises because any QECC code space that can be enforced by a set of constant-weight check operators can also be identified as the ground space of a local Hamiltonian.

[1] The CSS construction [26, 74] combines two classical codes, $C_1 = [N, k_1, d_1]$ and $C_2 = [N, k_2, d_2]$ to form an $[[N, k_1 + k_2 - N, \min(d_1, d_2)]]$ QECC with commuting check terms that generate a stabilizer subgroup of the Pauli group.

A central goal in these areas is to identify classes of local Hamiltonians with robust entanglement properties, and QLDPC codes provide a fruitful source of candidates. However, if the local terms are stabilizers then H is always a commuting Hamiltonian, and despite the richness of these systems they only capture a subset of local Hamiltonians and the properties they can exhibit.

Here we explore the QLDPC Conjecture (which posits that there exist asymptotically good QLDPC codes) through the correspondence between QLDPC codes and local Hamiltonians. This leads us to relax the requirement of being a CSS stabilizer code in two ways:

1. The code satisfies an *approximate* error-correction property: after an error channel is applied the decoding procedure recovers encoded states up to some $1-\varepsilon$ fidelity, where $\varepsilon = o(1)$.

2. The codespace is specified as the groundspace of a frustration-free local Hamiltonian $H = \Pi_1 + \cdots + \Pi_m$, where the local projectors Π_i don't necessarily commute.

Codes satisfying the approximate reovery condition are known as *approximate quantum error correcting codes* (AQECC), and codes with noncommuting frustration-free local check terms have been considered as a generalization of QLDPC in Hamiltonian complexity, therefore we call codes satisfying satisfying these conditions *approximate QLDPC codes.*

Our results

Our main result is a construction of approximate QLDPC codes with nearly-optimal parameters.

Theorem 2.1.1. *For infinitely many N there exists N-qubit subspaces $\{C_N\}$ with the following properties:*

1. *C_N is an AQECC that encodes $k = \tilde{\Omega}(N)$ logical qubits in N physical qubits, has distance $d = \tilde{\Omega}(N)$, approximation error $\varepsilon = O(1/\mathrm{polylog}\, N)$, and a $\mathrm{poly}(N)$ time encoding algorithm.*

2. *C_N is the ground space of a frustration-free local Hamiltonian $H^{(N)} = \sum H_i^{(N)}$ such that each term $H_i^{(N)}$ acts on $O(1)$ qubits, and each physical qubit participates in at most $\mathrm{polylog}\, N$ terms.*

3. *The Hamiltonian $H^{(N)}$ has spectral gap $\tilde{\Omega}(N^{-3.09})$ and it is spatially local in $\mathrm{polylog}(N)$ dimensions (i.e. it can be embedded in $\mathbb{R}^{\mathrm{polylog}\, N}$ with finite qubit density and geometrically local interactions).*

Here, the notation $\tilde{\Omega}(\cdot)$ *suppresses factors of* $\operatorname{polylog} N$.

The fact that the local check terms do not commute means that it is impossible to measure them all simultaneously. However, in Section 2.5 we show that any Pauli error will increase the energy of at least one local check term by at least $1/\operatorname{polylog}(N)$, and we use this to show that this family of codes is capable of *locally* detecting arbitrary Pauli errors with $\operatorname{polylog}(N)$ depth circuits.

Theorem 2.1.2. *For the family of codes described above, there exists with high probabilty a collection* $\mathcal{D}$ *of* $\operatorname{polylog}(N)$*-local projectors satisfying the following properties:*

1. *Each projector* $\Pi \in \mathcal{D}$ *acts on* 10 *physical qubits in the code and* $s = \operatorname{polylog}(N)$ *ancilla qubits initialized in the* $|0\rangle$ *state, and* $\Pi|\psi\rangle|0^s\rangle = 0$ *for all* $\Pi \in \mathcal{D}$ *if and only if* $|\psi\rangle \in C^N$.

2. *For all Pauli channels* $\mathcal{E}$*, for all codewords* $|\psi\rangle \in C$*, there exists a projector* $\Pi \in \mathcal{D}$ *such that*

$$\operatorname{Tr}\left(\Pi \ (\mathcal{E}(\psi) \otimes |0^s\rangle\langle 0^s|)\right) \geqslant (1-\alpha)(1-2^{-\operatorname{polylog}(N)}) \tag{2.1.1}$$

   where $\psi = |\psi\rangle\langle\psi|$ *and* α *is the total weight of the channel* $\mathcal{E}$ *on the (nonlocal) Pauli stabilizers in* $\mathcal{S}$.

Furthermore, there exists a measurement M*, implementable by a circuit of* $\operatorname{polylog}(N)$ *depth acting on* $O(N \operatorname{polylog}(N))$ *qubits, such that for all Pauli channels* $\mathcal{E}$ *and for all codewords* $|\psi\rangle \in C$

$$\operatorname{Tr}\left(M \ \left(\mathcal{E}(\psi) \otimes |0^{Ns}\rangle\langle 0^{Ns}|\right)\right) \geqslant (1-\alpha)(1-2^{-\operatorname{polylog}(N)}). \tag{2.1.2}$$

Our construction of this family of codes is based on a recently discovered connection between AQECC and Feynman-Kitaev (FK) Hamiltonians [73]. FK Hamiltonians have ground states of the form $\frac{1}{\sqrt{T+1}} \sum_{t=0}^{T} |t\rangle|\psi_t\rangle$, where $|\psi_t\rangle = U_t \ldots U_1 |0^n\rangle$[2] is the state of a quantum circuit at time t, and are used to prove the quantum version of the Cook-Levin theorem. The connection to AQECC is based on mapping the encoding circuit of a QECC to the ground space of a local Hamiltonian. To construct the family of codes in Theorem 2.1.1 we apply the connection formed in [73] to a randomized construction of good quantum codes with polylogarithmic depth encoding circuits [24]. The polylogarithmic factors in our construction arises from the additional "clock" qubits that are used in this mapping from circuits to ground

[2] We use n for the number of input qubits in a circuit Hamiltonian, and N for the number of physical qubits in our code construction. $N = n \operatorname{polylog}(n)$ in our construction because of the overhead used to represent the clock.

states. However, the standard FK construction uses a single global clock variable and does not allow for gates to be applied in parallel; to take full advantage of these parallel encoding circuits we present a substantial new technical analysis of the many-clock "spacetime" [23, 71] version of the FK construction that assigns an independent clock variable t_i to each qubit i in the circuit[3].

The spacetime circuit Hamiltonian enforces a ground state that is a uniform superposition over all valid configurations of these clocks (where validity is determined by the pattern of gates in the circuit), and it is unitarily equivalent to the normalized Laplacian of a random walk on the high-dimensional space of partially completed circuit configurations. Spacetime circuit Hamiltonians have been used previously for universal adiabatic computation and QMA-completeness constructions that are spatially local on a square lattice and do not require perturbative gadgets [23, 49, 66]. The analysis of the spectral gap in these previous works has always relied on the exact solutions to certain 1 + 1 dimensional quantum spin chains [61]. Here we develop a nearly tight lower bound on the spectral gap of the spacetime circuit Hamiltonian for a particular uniform class of circuits based on bitonic sorting networks. These sorting networks are used to transform a D depth circuit with arbitrary connectivity and n qubits into a depth $D\log(n)^2$ circuit[4] with spatially local connectivity in $\log(n)$ dimensions. By analyzing these sorting networks we prove the following general theorem in Section 2.4.

Theorem 2.1.3. *For any depth D quantum circuit of 2-local gates on n qubits, where n is a power of 2, there is an associated spacetime circuit-to-Hamiltonian construction which is spatially local in polylog(n) dimensions and has a spectral gap that is $\Omega(n^{-3.09}D^{-2}\log^{-6}(n))$.*

The spectral gap of a code Hamiltonian lower bounds the soundness of the code, since it determines the minimum energy of states outside of the code space. In our code construction we take $D = \text{polylog}(n)$, and since the circuit Hamiltonian acts on a total of $N = n\,\text{polylog}(n)$ qubits this accounts for the bound on the spectral gap in Theorem 2.1.1. Since our proof holds for any circuit with arbitrary connectivity we state the general result here for future potential applications to QMA and universal adiabatic computation.

Discussion

We believe that our approximate QLDPC codes, beyond being an attempt to address the QLDPC Conjecture via a different perspective, also illustrate a compelling synthesis of various intriguing concepts of quantum information theory, and furthermore, highlight several connections that deserve closer investigation.

[3] The term "spacetime" comes from relativistic physics, in which time is necessarily measured by local clocks.

[4] All logarithms in this work are base 2.

Approximate quantum error correction. AQECCs generalize QECCs by only requiring that the quantum information stored in the code, after the action of an error channel, be recoverable with fidelity at least $1-\varepsilon$. AQECCs have long been known to be capable of achieving better parameters than standard QECCs [32, 64], though the necessary and sufficient conditions for approximate recovery were only established within the last decade [15]. AQECC have found applications to fault-tolerant quantum computation [21, 63] through the analysis of realistic perturbations to exact QECC, and have recently experienced a resurgence in popularity in physics due to connections made with the holographic correspondence in quantum gravity [10]. Recently [44] have considered a version of local AQECC which also includes the possibility of locally approximate correction of errors in order to investigate the ultimate limits of the storage of quantum information in space. One can interpret our approximate QLDPC codes as providing another demonstration that the AQECC condition is a useful relaxation that facilitates the construction of codes with superior parameters than what is (known to be) achievable in the standard QECC framework.

Codes from local Hamiltonians. As previously mentioned, QLDPC codes have been a fruitful source of local Hamiltonians with robust entanglement properties, which are central objects of study in quantum Hamiltonian complexity and condensed matter theory. The first example of a QLDPC code was Kitaev's toric code, which is also a canonical example of a topologically ordered phase of matter [58]. Most research on QECC has been focused on stabilizer codes, like the toric code, for which the associated code Hamiltonians are commuting and frustration-free. In this paper we proceed in the opposite direction by asking: what kinds of quantum codes can we construct from local Hamiltonians whose terms don't necessarily commute? With this perspective, the extensive toolbox of techniques for constructing and analyzing Hamiltonians in quantum computing and quantum physics becomes immediately useful. This approach is inspired by several recent papers:

1. In [40], Eldar, et al. defined general QLDPC codes to be subspaces S that are stabilized by a collection of local projectors $\{\Pi_i\}$; in other words, $\Pi_i|\psi\rangle = 0$ for all i if and only if $|\psi\rangle \in S$. They call the Π_i projectors "parity checks" in analogy to the parity check terms of CSS codes; however, the projectors $\{\Pi_i\}$ need not be parity checks in the traditional sense.

2. In [44], Flammia, et al. formalized a notion of local AQECCs that includes an additional condition of approximate local correctability. This notion was applied to derive bounds on the ultimate limits of the storage of quantum information in spatially local codes.

3. In [18], Brandao et al. show that qutrit systems on a line with nearest-neighbor interactions can form approximate QLDPC that encode $\log(N)$ qubits with

distance $\log(N)$, and also show that AQECC can appear generically in energy subspaces of local Hamiltonians.

4. In [73], Nirkhe, et al. shows that by using the Feynman-Kitaev circuit-to-Hamiltonian construction and a *non-local* CSS code, one can obtain a *local* approximate QECC where the corresponding Hamiltonian's ground space is approximately the original CSS code.

Although there are still many hurdles to climb before codes with noncommuting checks can be realistically applied to fault-tolerance protocols, these recent developments form an exciting frontier in the study of local Hamiltonians. Another example of this connection is that the approximate codes developed in [73] and extended here can be seen as an instance of the recently formalized notion of Hamiltonian sparsification [6].

Comparison with the sparse subsystem codes of [12]. In [12] Bacon et al. construct subsystem codes with distance $\Omega(N^{1-\xi})$ for $\xi = O(1/\sqrt{\log N})$ and constant weight gauge generators, and these were termed "sparse subsystem codes." These are the best parameters achieved to date for any exact QECC in the ground space of a local Hamiltonian. Even more remarkable, in relation to the present work, is the fact that the codes of Bacon et al. have local checks that arise in a completely different way from quantum circuits. The difference is that [12] considers fault-tolerant circuit gadgets (instead of encoding circuits as in [73] and this work) and enforces the correct operation of these Clifford circuits according to the Gottesman-Knill theorem (rather than FK circuit Hamiltonians).

Another difference between these code constructions is that the code Hamiltonians of Bacon et al. are necessarily frustrated due to the fact that the noncommuting gauge generators are all Pauli operators, which therefore anticommute and share no simultaneous eigenstates. Although frustration does not always preclude the possibility of local error correction [44], there is no lower bound established on the spectral gap of the codes in [12] (and so there may be states outside the codespace with exponentially small energy), and detecting an error on a single qubit requires measuring $\text{poly}(N)$ gauge generators in order to ascertain the syndromes of nonlocal stabilizers. With this understanding we summarize past results on QECC with strong parameters:

Reference	# of logical qubits	Distance	Locality	Notes
[76]	$\Theta(N)$	$\Theta(\sqrt{N})$	$O(1)$	CSS Stabilizer code
[45]	$O(1)$	$O(\sqrt{N \log N})$	$O(1)$	CSS Stabilizer code
[19]	$\Theta(N)$	$\Theta(N)$	$\Omega(\sqrt{N})$	CSS Stabilizer code
[52, 54]	$O(1)$	$\Omega(N^{1-\xi})$ for all $\xi > 0$	$O(1)$	CSS code, assumes conjecture in high dimensional geometry
[12]	$O(N)$	$\Omega(N^{1-\xi})$ for all $\xi > 0$	$O(1)$	Subsystem Stabilizer code, frustrated Hamiltonian
This paper	$\Omega(N/\text{polylog}\, N)$	$\Omega(N/\text{polylog}\, N)$	$O(1)$	approximate QLDPC code

Connections with QPCP. On of the most significant open problems in Hamiltonian complexity is to resolve the quantum PCP conjecture [9], which posits that quantum proofs can be made probabilistically checkable. Since local Hamiltonians and the complexity class QMA are the respective quantum generalizations of constraint satisfaction problems and NP, the QPCP conjecture is equivalent to the statement that it is QMA-complete to decide whether the ground state energy of a Hamiltonian $H = \sum_{i=1}^{m} H_i$ is less than a or greater than b (under the promise that one of these is the case), where $b - a > \frac{c}{(m \cdot \max_i \|H_i\|)}$ for some $c = \Omega(1)$ corresponds to constant relative precision. One reason this question is difficult is any trivial state which is output by a constant-depth quantum circuit acting on a product state can be given as an NP witness, and many of the commonly studied classes of local Hamiltonians necessarily have low-energy trivial states. Therefore in order for QPCP to hold there must be some Hamiltonian with no low-energy trivial states, and even this weaker NLTS conjecture [53] remains an open problem.

One approach to resolving the NLTS and QPCP conjectures is to develop the quantum analogue of locally testable codes, which are defined in [5] as codes with frustration-free but not necessarily commuting local checks, good parameters, and a *soundness* property which states that the energy of a state with respect to the constraints grows linearly with its distance from the code space. Therefore constructing good QLDPC is necessary for constructing QLTC, but it is not sufficient since in general QLDPC may have low energy states outside the code space. This collection of open challenges that are stimulating innovations in Hamiltonian complexity is known as the robust entanglement zoo [39], since they all involve generalizing known properties of quantum ground states to states with constant relative distance above the code space.

Just as the classical PCP Theorem indirectly transforms a Cook-Levin computational tableau into a probabilistically checkable CSP, a QPCP construction could be seen as transforming the FK circuit-to-Hamiltonian construction into a local Hamiltonian with robust entanglement. While known limitations on generalized FK constructions make such a direct approach unlikey [14, 47, 48], our Theorem 2.1.2 on local error detection in polylog(N) depth is the first result to quantitatively substantiate the belief that the spacetime Hamiltonian construction is more robust than the standard

global-clock FK Hamiltonian. Specifically, we show that the energy of a state after the application of a Pauli error channel is inversely proportional to the *depth* of the circuit in the spacetime construction, whereas it is proportional to the *size* of the circuit in the standard FK construction. In fact in Section 2.6 we describe an alternate version of our approximate QLDPC construction that is based on global-clock FK and a modified distribution over time steps of the quantum circuit, and this version can achieve any scaling of the approximation error $\varepsilon(N) > 0$ at the expense of decreasing the spectral gap to $\tilde{\Omega}(\varepsilon N^{-3})$, but this substantially weakens the corresponding version of Theorem 2.1.2 and forces the local error detection circuits to have superlinear depth. This results suggest that continued investigation into alternative circuit-to-Hamiltonian constructions might be a fruitful direction of research, and might possibly make headway towards the mystery of the QPCP conjecture.

Overview of the remaining sections. Section 2.1 overviews the spacetime circuit Hamiltonian used in our construction, and Section 2.1 sketches the proof techniques we use to lower bound the spectral gap of the code Hamiltonian, which is the main technical contribution of this work. Section 2.2 formally defines approximate QLDPC codes and develops the machinery needed to describe our construction, including the good codes with polylogarithmic depth encoding circuits due to Brown and Fawzi [24] in Section 2.2, spacetime circuit Hamiltonians in Section 2.2, and bitonic sorting networks in Section 2.2. Our code construction and the efficient encoding circuit are given in Section 2.3, and the analysis of the spectral gap result in Theorems 2.1.1 and 2.1.3 is given in Section 2.4. The local error detection analysis underlying Theorem 2.1.2 is given in Section 2.5, and finally we discuss alternate versions of the construction in Section 2.6 and a spatially local embedding in Section 2.6. Appendix 2.7 contains many detailed results on combinatorial properties of partially completed circuit configurations of bitonic sorting networks, as well as the connection between these circuit configurations and dyadic tilings.

Description of the code Hamiltonian

In [73] it was recognized that the FK Hamiltonian which maps circuits to ground states could be used to develop a set of local checks for AQECC for which only an efficient encoding circuit was previously been found. For a circuit with local gates $U_1, ..., U_T$ the FK ground states are

$$|\Psi\rangle = \frac{1}{\sqrt{T+1}} \sum_{t=0}^{T} |t\rangle_{\mathsf{C}} \otimes (U_t U_{t-1} \cdots U_1)|\psi, 0 \dots 0\rangle_{\mathsf{S}}. \tag{2.1.3}$$

Such states are called *history states*. The register C, called the *clock register*, indicates how many gates have been applied to the all zeroes state, which is stored in register S (called the *state register*) containing an initial state $|\psi\rangle$ and ancillas.

Although this state has only a $1/(T+1)$ fidelity with the output of the circuit, the standard technique for increasing the overlap to be inverse polynomially close to 1 is to pad the end of the circuit with identity gates (for recent work on more efficient methods for biasing the history state towards its endpoints, see [14, 25]). This technique allows history states to capture approximate versions of QECC that have efficient encoding circuits. The approximation error of the code is directly related to history state overlap with the output of the encoding circuit.

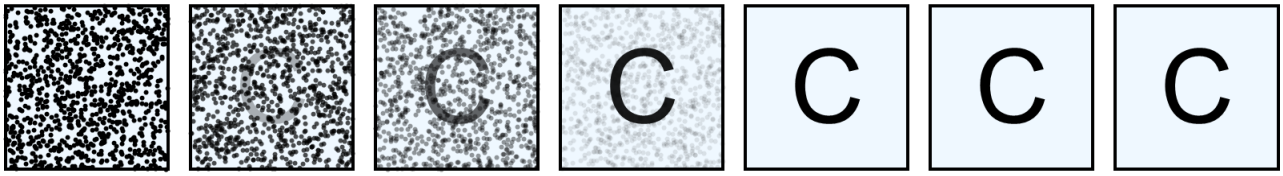


Figure 2.1: The approximate nature of the codes introduced in [73] arises from the fact that part of the history state superposition corresponding to early time steps, which do not match the output of the encoding circuit and are treated as noise in our analysis. Once a sufficient depth to form a codeword is reached, the computation can be padded with identity gates in order to increase the overlap of this approximate codeword with the original codeword it is approximating.

The Hamiltonian which enforces the ground space spanned by states of the form (2.1.3) is formed by projectors that check the input state of the computation, as well as *propagation terms* that check that the branch of the superposition corresponding to time t and the branch corresponding to time $t+1$ differ by the application of the gate U_{t+1} to the state register. The linear ordering of the computation $U_1, \ldots, U_T$ is enforced via the sum of these propagation terms. The propagation Hamiltonian is unitarily equivalent to a normalized Laplacian on the path graph with vertices $\{0, ..., T\}$ and therefore has a spectral gap that is $\Theta(T^{-2})$. For the purpose of lower bounding the energy of excitations that leave the code space, it is important to check the spectral gap of the full Hamiltonian including the input check terms, see Section 2.1 for further discussion.

In this work we use the spacetime version of the FK circuit Hamiltonian [23], which assigns a clock register to each computational qubit, and has a ground space spanned by uniform superposition over all valid time configurations $\boldsymbol{\tau} = (t_1, ..., t_n)$ of the state of the computation after the gates prior to $\boldsymbol{\tau}$ have been performed,

$$|\psi\rangle = \frac{1}{|\mathcal{T}|^{1/2}} \sum_{\boldsymbol{\tau} \in C} |\boldsymbol{\tau}\rangle_{\mathsf{C}} \otimes U(\boldsymbol{\tau} \leftarrow 0)|0 \cdots 0\rangle_{\mathsf{S}}. \tag{2.1.4}$$

Here $\mathcal{T}$ is the set of all valid time configurations $\boldsymbol{\tau}$, which is any vector $(t_1, \ldots, t_n)$ that the clock registers could hold if a subset of gates that respected causal dependence (see Definition 2.3.4) were applied. To avoid boundary effects at the beginning and end of the computation we use circular (periodic) time, which involves reversing the

gates in the second half of the circuit so that the computation returns to its initial state. In Section 2.2 implement these periodic clocks using qubits.

The necessity of including these causal constraints is one of the complications introduced by the use of spacetime circuit Hamiltonians, but a far more significant challenge is lower bounding the spectral gap of the spacetime propagation Hamiltonian. In contrast with single-clock circuit Hamiltonians, the geometric arrangement of the gates in the circuit now has a significant effect on the spectrum of the spacetime circuit Hamiltonian due to the causal constraints. All lower bounds in previous works apply to spacetime Hamiltonians in 2 spatial dimensions, which represent 1 (space) + 1 (time) dimensional quantum circuits. This is not only due to the importance of planar connectivity for practical applications, but it is also a symptom of the general fact that exactly solvable models in mathematical physics are hardly known beyond 1 + 1 dimensions. The 1 + 1 dimensional circuit propagation Hamiltonian is unitarily equivalent to a stochastic model describing the evolution of a string in the plane. For higher dimensional circuits it corresponds to the dynamics of membranes or crystal surface growth, where no known solutions are available. To overcome this in the present work we use sorting networks to turn arbitrary random circuits into circuits with uniform connectivity, and then we apply powerful techniques and past results from the theory of Markov chains to analyze the resulting high-dimensional spacetime circuit Hamiltonians.

Proof sketch for the spectral gap analysis

Our analysis of the spectral gap Δ_{prop} of the spacetime circuit propagation Hamiltonian begins with the standard mapping from H_{prop} to a a Markov chain transition matrix P. [5] To analyze the latter, we apply a Markov chain decomposition method due to Madras and Randall [68], which is used to split the Markov chain and its state space into pieces that are easier to analyze individually. For our decomposition of choice these pieces come in several closely related variants, which all essentially correspond to the set of time configurations contained within the final phase of a bitonic sorting circuit (as shown in Figure 2.3 for 8 lanes) which we call a bitonic block. As described in Appendix 2.7, an arbitrary circuit consisting of 2-local gates can be transformed into a sequence of consecutive bitonic blocks, with at most a polylogarithmic factor of blow up in the depth.

After dividing the set of valid time configurations Ω (the state space of the Markov chain) into subsets Ω_i of configurations confined to bitonic blocks of the form illustrated in Figure 2.3, the subsets will form a quasi-linear chain in the sense

[5] The re-scaled Hamiltonian $H_{\text{prop}}/\|H_{\text{prop}}\|$ is unitarily equivalent to a normalized graph Laplacian $\mathcal{L}$ for the graph with vertices corresponding to valid time configurations and edges corresponding to local gate updates on those time configurations. P is the transition matrix for the random walk on this graph, which is obtained from $I - \mathcal{L}$ by a similarity transformation. The point is that these mappings provide an algebraic relation between Δ_{prop} and Δ_P.

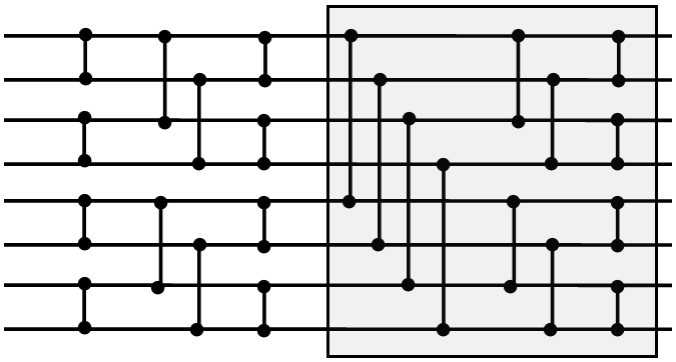


Figure 2.2: A bitonic sorting architecture on $n = 8$ bits. We refer to the final phase of the architecture, corresponding to the last $\log(n) = 3$ layers enclosed in a gray box, as a bitonic block. Note that the gates in each layer are executed simultaneously, but are drawn as non-overlapping for visual clarity. An arbitrary circuit consisting of 2-local gates can be transformed to have the architecture of consecutive repetitions of bitonic blocks at the cost of increasing the depth by a factor of $\log(N)^2$.

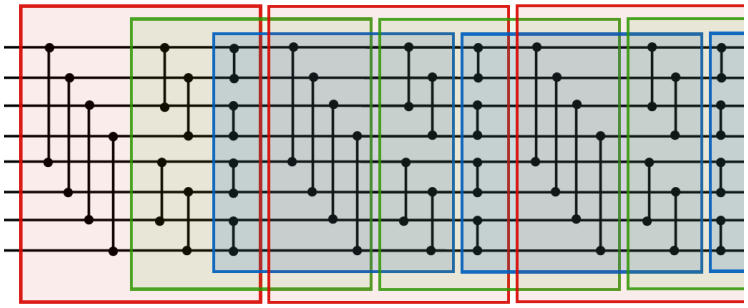


Figure 2.3: The Markov chain block decomposition for a sequence of padded bitonic sorting architecture on 8 bits. The set of valid time configurations contained entirely within the i-th colored rectangle constitutes the block Ω_i. The set of time configurations in two rectangles of different colors are related by a permutation of the qubit wires. The aggregate chain $\overline{P}$ has a nonzero transition probability $\overline{P}(i, j)$ iff the rectangles corresponding to the blocks Ω_i and Ω_j are overlapping. Each block Ω_i has a nonzero transition probability to $\log N$ other blocks Ω_j. Every valid time configuration is contained in at least one of the blocks, and no time configuration is contained in more than $\log N$ blocks.

that Ω_i and Ω_j have nonempty intersections when $|i - j| \leqslant \log n$. To apply the decomposition method we need to analyze (1) the spectral gap of the restricted Markov chains P_i that are confined to stay within each of the subsets Ω_i, and (2) the spectral gap of an aggregate Markov chain $\overline{P}$ that moves between the blocks based on transition probabilities related to the size of the intersections of the blocks.

As suggested by its quasi-linear connectivity, the spectral gap of the aggregate chain can be lower bounded using Cheeger's inequality in similar manner as is done for the path graph Laplacian. The main technical challenge is to accurately compute the transition probabilities $\overline{P}(i, j) = \pi(\Omega_i \cap \Omega_j)/(\Theta\pi(\Omega_i))$, which involve the ratio of the number of configurations within each of the blocks to the number within the

pairwise intersections, $|\Omega_i \cap \Omega_j|/|\Omega_i|$, as well as the maximum number of blocks Θ that can contain any particular time configuration. In Appendix 2.7, we develop a recurrence relation to exactly count these configurations and show that the former is constant for consecutive blocks (and decays doubly exponentially with $|i-j|$ for longer distance transitions), and the latter is logarithmic in n. Using asymptotic properties of the recurrence relation we show that the transition probabilities between $i, i+1$ are equal to $(\phi \log n)^{-1}$, where $\phi = (1+\sqrt{5})/2$ is the golden ratio. If there are m blocks in total so that the length of the path is m, we use Cheeger's inequality to show that the spectral gap $\Delta_{\overline{P}}$ of the aggregate chain satisfies

$$\Delta_{\overline{P}} \geqslant (\phi m \log n)^{-2} . \tag{2.1.5}$$

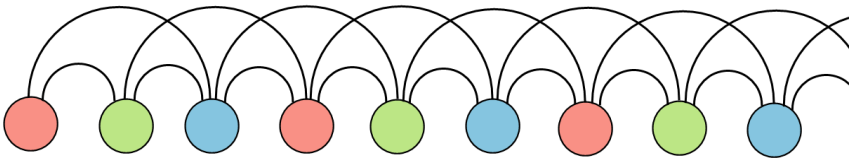


Figure 2.4: An illustration of the states and transitions in the aggregate chain corresponding to the subsets of time configurations contained with the blocks in fig 2.3.

Turning to the analysis of the restricted chains P_i, we present the discovery of a surprising and beautiful connection between valid time configurations of architectures of the form shown in Figure 2.2 with combinatorial structures known as dyadic tilings [56]. Dyadic tilings are tilings of the unit square by equal-area dyadic rectangles, which are rectangles of the form $[a2^{-s}, (a+1)2^{-s}] \times [b2^{-t}, (b+1)2^{-t}]$, where a, b, s, t are nonnegative integers. These tilings have a natural recursive characterization: beginning from the unit square, draw a line that is either a horizontal or vertical bisector. This divides the square into two rectangles, and in each of these one chooses a horizontal or vertical bisector, and so on. After $\ell = \log(n)$ such recursive steps one obtains a dyadic tiling of rank ℓ with a total of n dyadic rectangles, each with area $1/n$. Some examples are given in Figure 2.5.

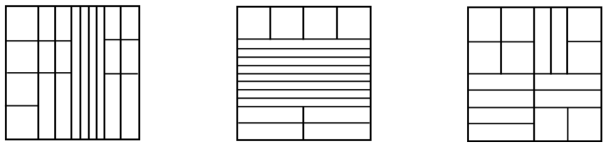


Figure 2.5: Examples of dyadic tilings of rank 4.

For a spacetime circuit with n qubits, we choose the blocks Ω_i in the decomposition so that for each block there is an exact bijection between the time configurations within the block and the set of equal-area dyadic tilings of rank $\ell = \log n$. Moreover,

it turns out that the natural Markov chain on time configurations can also be mapped onto a previously defined Markov chain for dyadic tilings called the edge-flip chain. This Markov chain selects a rectangle of area $1/n$ in the current dyadic tiling and one of its four edges at random, and flips this edge if the result would be another dyadic tiling. The correspondence is described in Figure 2.6.

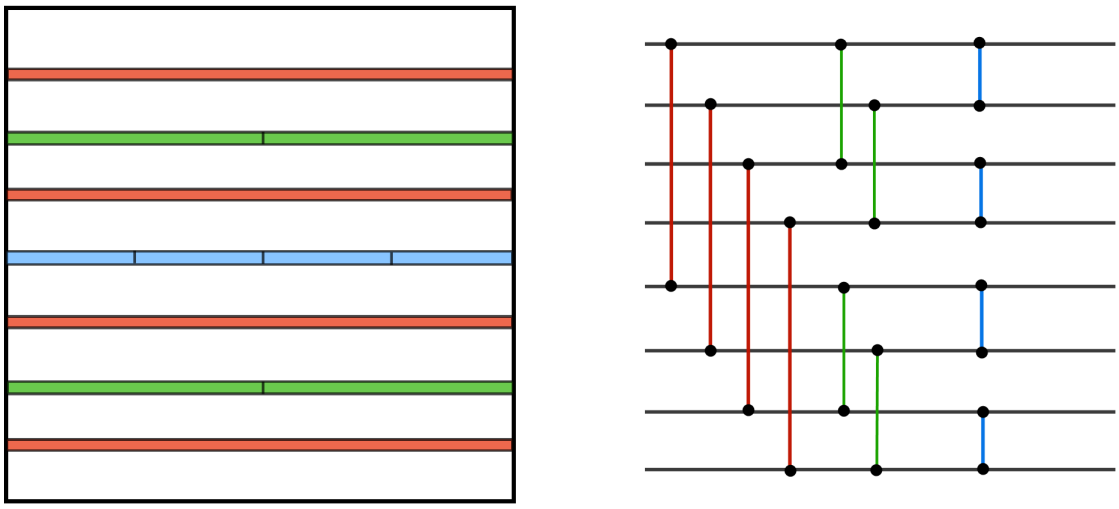


Figure 2.6: A color-coding of the correspondence between dyadic tilings and valid time configurations of a bitonic sorting circuit. The colored line segments in (a) correspond to sub-edges which when rotated by $\pi/2$ about their midpoint will be sub-edges of a vertical edge in some dyadic tiling. These edges are placed in correspondence with the gates of the bitonic sorting circuit in (b), with the convention that colored line segments in (a) are ordered from left to right and from top to bottom, and the gates in a given commuting layer in (b) are enumerated from top to bottom. Given an arbitrary dyadic tiling, one checks which of the colored line segments in (a) correspond to vertical sub-edges in the tiling, and these correspond to gates that are in the past causal cone of the bitonic time configuration associated with that tiling.

The mixing time of this edge flip chain was an open problem for over a decade, but has recently been the subject of a tour de force analysis that establishes an upper bound on the mixing time that is polynomial in n. Adapting these results using our bijection between these Markov chains yields

$$\Delta_{P_i} = \Omega\left(n^{-4.09}\right) \quad , \quad \text{for all} \quad i = 1, ..., m, \tag{2.1.6}$$

where the value of the exponent can be taken to be $\log(17) = 4.087\ldots$. Once (2.1.5) and (2.1.6) are established, we combine them according to the decomposition result,

$$\Delta_P \geqslant \frac{1}{2}\Delta_{\overline{P}} \min_{i=1,\ldots,m} \Delta_{P_i} = \Omega\left(n^{-4.09} m^{-2} \text{polylog}(n)^{-1}\right),$$

which is an inverse polynomial lower bound on the gap. The circuit propagation Hamiltonian is equivalent to the Markov chain P scaled by a factor of n, and so we

obtain $\Delta_{\text{prop}} = \widetilde{\Omega}(n^{-3.09})$. Finally, using the version of the spacetime Hamiltonian with circular time we show that every state in the code space has overlap $1/\text{polylog}(n)$ with the input terms and so the geometrical lemma yields a gap of $\widetilde{\Omega}(n^{-3.09})$ for the full code Hamiltonian.

2.2 Preliminaries

In what follows, we present the definitions of the main ingredients of our code construction and analysis.

Approximate QLDPC codes

Here we present the formal definition of an approximate QLDPC code.

Definition 2.2.1 (Approximate QLDPC code)**.** A 2^k-dimensional subspace C of $(\mathbb{C}^2)^{\otimes N}$ is a $[[N, k, d, \varepsilon, \ell, s]]$ *approximate QLDPC code* iff there exists a (not necessarily commuting) set of projectors $\{H_1, \ldots, H_m\}$ acting on N qubits such that

1. Each term H_i acts on at most ℓ qubits (i.e. *locality*) and each qubit participates in at most s terms (i.e. *sparsity*).

2. For all $|\psi\rangle$, we have that $|\psi\rangle \in C$ if and only if $\langle\psi|H|\psi\rangle = 0$, where $H = H_1 + \cdots + H_m$.

3. There exist encoding and recovery maps Enc, Rec such that for all $|\phi\rangle \in (\mathbb{C}^2)^{\otimes k} \otimes \mathcal{R}$ where $\mathcal{R}$ is some purifying register, for all completely positive trace preserving maps $\mathcal{E}$ acting on at most $(d-1)/2$ qubits, we have that the image of Enc is exactly the code C and

$$F\left(\text{Rec} \circ \mathcal{E} \circ \text{Enc}(|\phi\rangle\langle\phi|), |\phi\rangle\langle\phi|\right) \geqslant 1 - \varepsilon \tag{2.2.1}$$

   where $F(\cdot, \cdot)$ denotes the fidelity function. Here, the maps Enc, $\mathcal{E}$, and Rec do not act on register $\mathcal{R}$.

The first condition of the above definition enforces the locality and sparsity conditions of the approximate QLDPC code. The second condition enforces that the code is the ground space of a frustration-free local Hamiltonian. The third condition corresponds to the approximate error-correcting condition, where we only require that the decoded state is *close* to the original state (i.e., we no longer insist that $\text{Rec} \circ \mathcal{E} \circ \text{Enc}$ is exactly the identity channel). Although there are few results on approximate quantum error-correcting codes, we do know that relaxing the exact decoding condition yields codes with properties that cannot be achieved using exact codes [15, 64].

Parallel quantum circuits

We establish some notational conventions for parallel quantum circuits.

Consider the following model of depth D circuits on n qubits. The circuit C consists of D layers $L_1, \ldots, L_D$. In each layer L_t for $1 \leqslant t \leqslant D$, the n qubits are partitioned into $n/2$ disjoint pairs $\{(p, q)\}$, and a two-qubit gate $U_t(p, q)$ acts on the qubit pair (p, q). Layer L_1 is applied first, then layer L_2, and so on. The unitary corresponding to circuit C is

$$\prod_{t=1}^{D} \bigotimes_{(p,q)\in L_t} U_t[p, q] \tag{2.2.2}$$

where the product is written from right to left. In other words, the unitary $\bigotimes_{(p_1,q_1)\in L_1} U_1[p_1, q_1]$ is the rightmost factor, followed by $\bigotimes_{(p_2,q_2)\in L_2} U_2[p_2, q_2]$, and so on.

Model for random low-depth Clifford circuits. Our model for random depth D Clifford circuits is to choose, for each layer L_t, a random partition $\{(p, q)\}$ of the n qubits, and then for each pair (p, q), and let $U_t[p, q]$ be a uniformly chosen from the two-qubit Clifford group (i.e., the set of all unitaries that preserve the Pauli group under conjugation).

Brown and Fawzi showed that for $D = O(\log^3 n)$, the circuit C is an encoding circuit for a good error-correcting code with high probability [24]:

Theorem 2.2.2 ([24])**.** *For all $\delta > 0$, for all integers $n, k, d > 0$ satisfying*

$$\frac{k}{m} \leqslant 1 - h(d/n) - \log(3)d/n - 4\delta, \tag{2.2.3}$$

with $h(\cdot)$ as the binary entropy function, the circuit C described in the paragraph above is an encoding circuit for a $[[m, k, d]]$ stabilizer code with probability at least $1 - \Omega(n^{-8})$. In other words, with high probability the subspace $C = \{C|\psi\rangle|0\rangle^{\otimes(n-k)} : |\psi\rangle$ is a k-qubit state$\}$ is a $[[n, k, d]]$ stabilizer code.

Notation 2.2.3. To avoid confusion with the blocklength of our approximate QLDPC code that we construct in our paper (which is denoted by N), we will use n to denote the blocklength of the Brown-Fawzi random circuit code.

Since the circuits are Clifford circuits, the resulting code is a stabilizer code.

The spacetime circuit Hamiltonian construction

As mentioned in the introduction, we use a small variant of the spacetime circuit Hamiltonian of Brueckmann and Terhal [23] to create our code Hamiltonian. In this section, we present the spacetime construction for general depth D circuits.

In Section 2.3, we will describe the specific circuit that we will use for our code Hamiltonian.

Let D be an even integer and let C be an n-qubit circuit of depth D where $L_1, \ldots, L_D$ be the D layers of C, where each L_t is a set of $n/2$ two-qubit gates[6] $U_t[p, q]$ acting on disjoint pairs of qubits $\{(p, q)\}$. We assume that C is a "circular" circuit; in other words, that it is equivalent to the identity circuit.

We let $H_{circuit}[C]$ denote the circular spacetime circuit Hamiltonian corresponding to the circular circuit C. Let $X \stackrel{\text{def}}{=} \frac{D-2}{2}$. The Hamiltonian is defined on $n(1+X+1) = n(X+2)$ qubits, which is divided into three classes of registers: (1) data registers $\mathsf{S}_1, \ldots, \mathsf{S}_n$, (2) clock registers $\mathsf{C}_1, \ldots, \mathsf{C}_n$, and (3) flag registers $\mathsf{F}_1, \ldots, \mathsf{F}_n$.

The data register S_i is a qubit register that corresponds to the i-th qubit that the circuit C acts on. The flag register F_i is a qubit register that indicates whether the i-th qubit's local clock is in the "forward phase" or the "backward phase"; this denotes which half (first or second) of clock states the clock is in. The clock register C_i consists of X qubits and indicates the local time of the i-th data qubit (within the forward phase or the backward phase). The valid clock states for register C_i are $\{|1^j 0^{X-j}\rangle\}$ for $0 \leqslant j \leqslant X$ (i.e. a domain wall clock).

Following Brueckmann and Terhal [23], the flag register combined with the clock register allows us to put our qubit clocks "on a circle": we index time from 0 to $2X + 1 = D - 1$, and we identify time $t = D$ with $t = 0$. We encode time steps t according to the following convention. For notational convenience, we let the register T_i (for "time register") denote the union of F_i and C_i.

$$|t\rangle_{\mathsf{T}_i} \stackrel{\text{def}}{=} \begin{cases} |0\rangle_{\mathsf{F}_i} \otimes |1^t\, 0^{X-t}\rangle_{\mathsf{C}_i} & \text{if } t \in \{0, 1, \ldots, X\} \\ |1\rangle_{\mathsf{F}_i} \otimes |1^X\rangle_{\mathsf{C}_i} & \text{if } t = X+1 \\ |1\rangle_{\mathsf{F}_i} \otimes |1^{2X+1-t}\, 0^{t-X-1}\rangle_{\mathsf{C}_i} & \text{if } t \in \{X+2, \ldots, 2X+1\}. \end{cases} \tag{2.2.4}$$

In other words, the time register evolves in the following way:

$$|0\rangle_{\mathsf{F}_i} \otimes |00\cdots0\rangle_{\mathsf{C}_i} \to \cdots \to |0\rangle_{\mathsf{F}_i} \otimes |11\cdots1\rangle_{\mathsf{C}_i} \qquad (0 \leqslant t \leqslant X) \tag{2.2.5}$$

$$\to \quad |1\rangle_{\mathsf{F}_i} \otimes |11\cdots1\rangle_{\mathsf{C}_i} \qquad (t = X+1) \tag{2.2.6}$$

$$\to \quad |1\rangle_{\mathsf{F}_i} \otimes |1\cdots10\rangle_{\mathsf{C}_i} \to \cdots \to |1\rangle_{\mathsf{F}_i} \otimes |00\cdots0\rangle_{\mathsf{C}_i} \qquad (X+2 \leqslant t \leqslant 2X+1). \tag{2.2.7}$$

[6] By padding with identity gates, we can assume without loss of generality that every layer has exactly $n/2$ two-qubit gates.

Notice that in any transition from $|t\rangle_{\mathsf{T}_i}$ to $|t+1\rangle_{\mathsf{T}_i}$, there is at most one qubit being flipped.

For the remainder of this section we fix a circuit C and assume it fixed. The spacetime Hamiltonian $H_{circuit}[C]$ is defined as

$$H_{circuit} = H_{clock} + H_{init} + H_{prop} + H_{causal}. \tag{2.2.8}$$

Notation 2.2.4. In what follows, subscripts of operators such as "F_i" in "$|0\rangle\langle 0|_{\mathsf{F}_i}$" indicates which registers the operators act on. Let $\Pi^{(\alpha)}_{\mathsf{R}}$ be the projector $|\alpha\rangle\langle\alpha|_{\mathsf{R}}$ for any register R.

The terms H_{clock}, H_{init}, H_{prop}, and H_{causal} are defined as follows:

(1) H_{clock}**:** The term H_{clock} enforces that all the clock registers are encoded as described above. We write $H_{clock} = \sum_{i=1}^{n} H_{clock}[i]$ where

$$H_{clock}[i] = \sum_{j=1}^{D-1} \Pi^{(01)}_{\mathsf{C}_{i,j}\mathsf{C}_{i,j+1}}. \tag{2.2.9}$$

This enforces that the register C_i encodes a domain wall.

(2) H_{init}**:** The initialization term is defined as $H_{init} = \sum_{i=k+1}^{n} H_{init}[i]$ for some integer $1 \leqslant k \leqslant n$, [7] where

$$H_{init}[i] = \Pi^{(1)}_{\mathsf{C}_{i,0}} \otimes \Pi^{(1)}_{\mathsf{S}_i}. \tag{2.2.10}$$

This term checks that the last $n-k$ qubits are in the state $|0\rangle$ when their corresponding time registers are in state $|0\rangle_{\mathsf{T}_i}$ or $|2X+1\rangle_{\mathsf{T}_i}$. We only need to check one bit of the time register T_i because of the previous set of terms enforcing that the clock is a domain wall.

(3) H_{prop}**:** The propagation term H_{prop} is defined to be $H_{prop} = \sum_{t=0}^{D-1} \sum_{(p,q)\in L_t} H_t[p,q]$, where

$$H_t[p,q] = \frac{1}{2}\Big[\big(A_{t,t}[p,q] + A_{t+1,t+1}[p,q]\big) \otimes \mathbb{1} - A_{t+1,t}[p,q] \otimes U_t[p,q] - A_{t,t+1}[p,q] \otimes (U_t[p,q])^\dagger \Big] \tag{2.2.11}$$

$$\text{and } A_{t,t'}[p,q] = |u_t[p]\rangle\langle u_{t'}[p]| \otimes |u_t[q]\rangle\langle u_{t'}[q]|, \tag{2.2.12}$$

[7] In our case, k will eventually be the number of logical qubits.

$$|u_t[p]\rangle = \begin{cases} \mathbb{1} \otimes |0\rangle_{\mathsf{F}_p} |1\rangle_{\mathsf{C}_{p,t}} |0\rangle_{\mathsf{C}_{p,t+1}} & \text{if } 0 \leqslant t < X \\ \mathbb{1} \otimes |0\rangle_{\mathsf{F}_p} |1\rangle_{\mathsf{C}_{p,X}} & \text{if } t = X \\ \mathbb{1} \otimes |1\rangle_{\mathsf{F}_p} |1\rangle_{\mathsf{C}_{p,X}} & \text{if } t = X+1 \\ \mathbb{1} \otimes |1\rangle_{\mathsf{F}_p} |1\rangle_{\mathsf{C}_{p,2X+1-t}} |0\rangle_{\mathsf{C}_{p,2X+2-t}} & \text{if } X+1 \leqslant t \leqslant 2X+1. \end{cases} \tag{2.2.13}$$

Here, $|u_t[p]\rangle$ is the tensor product of a state on the specified qubits and the identity operator on all unspecified qubits. This term enforces the agreement of slices of the superposition corresponding to two time configurations differing by a gate with respect to the unitary $U_t[p,q]$. Because of the H_{clock} terms, the checks only require looking at a few qubits of the time registers[8].

(4) H_{causal}**:** The term H_{causal} is used to *enforce causality* meaning that the superposition is only over *valid* time configurations (see Definition 2.3.4). At a high level, a time configuration $\boldsymbol{\tau} = (t_1, \ldots, t_n)$ is valid if and only if for all pairs of qubits (p,q) sharing a gate in layer L_t, both clocks t_p and t_q are $\leqslant t$ or $> t$. This is, however, complicated by the circularity of time imposed in this particular construction as "all clocks are both ahead and behind any particular t". In reality, we require the more complicated definition: for all pairs of qubits (p,q) sharing gates in layers L_{t_a} and L_{t_b} for $t_a < t_b$, either t_p, t_q are both $\in [t_a, t_b)$ or are both $\notin [t_a, t_b)$.

Let C_p be the set of qubits q which interact with qubit p.

$$H_{causal} = \sum_{p=1}^{n} \sum_{q \in C_p} H_{causal}[p,q] \tag{2.2.14}$$

where $H_{causal}[p,q]$ is defined as follows. Let $t^{(1)} < t^{(2)} < \ldots < t^{(f)}$ be the times at which p and q share a gate. Then,

$$H_{causal}[p,q] = \sum_{j=1}^{f} \sum_{t_p = t^{(j)}}^{t^{(j+1)}-1} A_{t_p,t_p}[p] \otimes B_{t^{(j)},t^{(j+1)}}[q] \tag{2.2.15}$$

where $B_{t,t'}[q]$ is a projector ensuring that qubit t_q is between t and t' (respecting circularity)[9]. Therefore, we verify that qubit q is valid with respect to qubit p. The definition of $B_{t,t'}[q]$ is case dependent.

Case 1 If $0 \leqslant t, t' \leqslant X$. In this case, the flag qubit must be $|0\rangle_{\mathsf{F}_q}$. Furthermore, $\mathsf{C}_{q,t}$ must be $|1\rangle$ and $\mathsf{C}_{q,t'}$ must be $|0\rangle$. Therefore,

$$B_{t,t'}[q] = \Pi^{(0)}_{\mathsf{F}_q} \Pi^{(1)}_{\mathsf{C}_{q,t}} \Pi^{(0)}_{\mathsf{C}_{q,t'}}. \tag{2.2.16}$$

[8] In effect, $|u_t[p]\rangle$ is the minimal description of $|t\rangle_{\mathsf{T}_p}$ given that the state is a ground-state of H_{clock}.

[9] By this we mean that if $t < t'$, the projector is onto the set $\{t, \ldots, t'-1\}$. If $t' < t$, then the projector is onto the set $\{t, \ldots, X\} \cup \{0, \ldots, t'-1\}$.

Case 2 If $X+1 \leqslant t, t' \leqslant 2X+1$. This is the similar except the flag is flipped. Hence,

$$B_{t,t'}[q] = \Pi^{(1)}_{\mathsf{F}_q} \Pi^{(1)}_{\mathsf{C}_{q,2X+2-t'}} \Pi^{(0)}_{\mathsf{C}_{q,2X+2-t}}. \tag{2.2.17}$$

Case 3 If $0 \leqslant t \leqslant X$ and $X+1 \leqslant t' \leqslant 2X+1$. In this case, the flag qubit may be different. However, we can write the projector as the sum of the two projectors for the different flags.

$$B_{t,t'}[q] = \Pi^{(0)}_{\mathsf{F}_q} \Pi^{(1)}_{\mathsf{C}_{q,t}} + \Pi^{(1)}_{\mathsf{F}_q} \Pi^{(1)}_{\mathsf{C}_{q,2X+2-t'}}. \tag{2.2.18}$$

Case 4 If $0 \leqslant t' \leqslant X$ and $X+1 \leqslant t \leqslant 2X+1$. This is similar except again the flag is flipped. Hence,

$$B_{t,t'}[q] = \Pi^{(0)}_{\mathsf{F}_q} \Pi^{(0)}_{\mathsf{C}_{q,t'}} + \Pi^{(1)}_{\mathsf{F}_q} \Pi^{(0)}_{\mathsf{C}_{q,2X+2-t}}. \tag{2.2.19}$$

Bitonic sorting networks

In this section, we describe a class of circuits called *bitonic sorting networks*. These are parallel circuits, devised by Batcher [13], that are used to efficiently sort data arrays. Specifically, these are circuits acting on n elements, with depth $O(\log^2 n)$. In each layer of the circuit, pairs of elements are compared and swapped. Equivalently, for every permutation π on n elements, there is a bitonic sorting network consisting of SWAP and identity gates that implements π.

Bitonic sorting networks will be a crucial component of our code construction, as we use them to "uniformize" the random Brown-Fawzi encoding circuits before applying the spacetime circuit Hamiltonian construction. The uniformity of the resulting circuits will be the key ingredient that allows us to analyze the spectral gap of the Hamiltonian.

Notation 2.2.5. We will assume that the number of qubits n, is a power of 2, with $n = 2^\ell$ for some integer ℓ.

For this paper, we will be interested in the architecture (i.e. the wiring and gate structure) of the bitonic sorting circuit. A bitonic sorting architecture consists of smaller sub-architectures, called *bitonic blocks*.

Definition 2.2.6. An *architecture* is a directed acyclic graph where each vertex v has $\deg_{in}(v) = \deg_{out}(v) \in \{1, 2\}$ except for specific vertices s and t which have $\deg_{out}(s) = \deg_{in}(t) = n$ and $\deg_{out}(t) = \deg_{in}(s) = 0$. A circuit C (acting on n qubits) over an architecture is instantiated by specifying a gate for each vertex $v \notin \{s, t\}$ in the graph that acts on the qubits labelled by the edges adjacent to the vertex. The vertices s and t represent the state prior to and after the application of the circuit. That is, we can think of an architecture as an outline of a quantum circuit and one needs to fill in the blanks (specify each gate) to instantiate a circuit.

Definition 2.2.7 (Bitonic block [13])**.** For a positive integer ℓ, the bitonic block of *rank* ℓ, $\mathcal{B}_\ell$, is a circuit architecture acting on 2^ℓ qubits. $\mathcal{B}_\ell$ is recurisvely defined with the architecture $\mathcal{B}_1$ being an architecture consisting of a single layer, $\mathcal{L}_1$, with a gate between qubits 1 and 2 (see part (a) of Figure 2.7).

For $\ell > 1$, the bitonic block $\mathcal{B}_\ell$ is a ℓ-depth architecture with the first layer, $\mathcal{L}_1$ being $2^{\ell-1}$ gates connecting qubit i to $i+2^{\ell-1}$ for $i = 1, 2, \ldots, 2^{\ell-1}$. The following $\ell-1$ layers, $\mathcal{L}_\ell, \mathcal{L}_{ell-1}, \ldots, \mathcal{L}_2$ are defined recursively as $\mathcal{B}_{\ell-1}^{\otimes 2}$ where one of the two blocks acts on the qubits $\{1, 2, \ldots, 2^{\ell-1}\}$ and the other on the qubits $\{2^{\ell-1}+1, 2^{\ell-1}+2, \ldots, 2^\ell\}$.

See Figure 2.7 for illustrations of blocks $\mathcal{B}_2$, and $\mathcal{B}_3$.

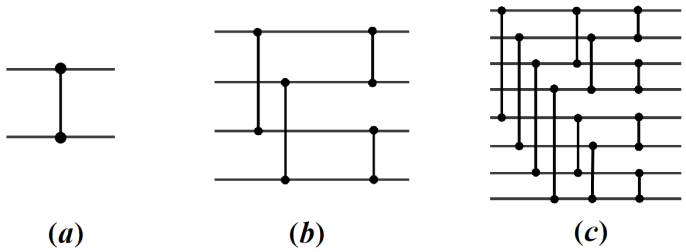


Figure 2.7: (a) Bitonic block $\mathcal{B}_1$. (b) Bitonic block $\mathcal{B}_2$. (c) Bitonic block $\mathcal{B}_3$.

Theorem 2.2.8 ([13])**.** *Let B_ℓ be an instantiation of a bitonic block architecture $\mathcal{B}_\ell$ with generalized comparator gates for some well-ordering – i.e. given two input wires, it either swaps them or performs the identity such that the larger element is on the lower wire. Then, given two monotonically decreasing sequences of length $2^{\ell-1}$ as inputs, the output of the circuit B is the merged monotonically decreasing sequence.*

Corollary 2.2.9 ([13])**.** *The following $\frac{\ell(\ell+1)}{2}$ depth circuit is a sorting circuit:*

$$B_\ell B_{\ell-1}^{\otimes 2} B_{\ell-2}^{\otimes 4} \ldots B_1^{\otimes 2^{\ell-1}}. \tag{2.2.20}$$

We will use the notation $\mathcal{B}_\ell^{\times r}$ for the product (i.e. concatenation) of r bitonic blocks $\mathcal{B}_\ell$. To simplify the analysis, we can insert additional layers of identity gates so that the circuit architecture is $\mathcal{B}_\ell^{\times \ell}$; this at most doubles the size of the circuit. Therefore, we can make the following statement:

Lemma 2.2.10. *For any permutation $\pi \in S_n$, there exists a circuit B_π of the architecture $\mathcal{B}_\ell^{\times \ell}$ applying π on the input wires.*

Proof. Note which comparator gates of the bitonic sorting circuit would be SWAP gates if sorting according to the permutation π. Pad with identity gates as previously stated till the circuit conforms to the architecture. □

Uniformizing circuits for spacetime Hamiltonians

We now present a general method for encoding depth D circuits C into a spacetime circuit Hamiltonian, in a way that allows us to give a good lower bound on the spectral gap. Let C denote a circuit of depth D consisting of layers $L_1, \ldots, L_D$, where each L_t is a set of $n/2$ two-qubit gates.

We preprocess the circuit C in multiple steps to obtain a slightly larger-depth circuit C'. We "uniformize" the circuit using bitonic sorting networks described in the previous section. The circuit C will not, in general, correspond to nearest-neighbor interactions in small dimension. We add bitonic sorting networks in between each layer L_t of C to ensure that all the Clifford gates act on adjacent qubits. Because of the regular structure of the sorting networks, the resulting circuit will consist of nearest-neighbor interactions on a hypercube of dimension $\ell = \log n$.

More formally, we do the following: label the qubits using $\{1, \ldots, n\}$. In a layer L_t of C for $1 \leqslant t \leqslant D$, a qubit q is generally not paired with a neighboring qubit $q-1$ or $q+1$. Instead, there is some permutation π_t on n qubits that maps the pairs $\{(1,2),(2,3),\ldots,(n-1,n)\}$ to the pairs $L_t = \{(p,q)\}$. Let $\pi_t(L_t)$ denote the layer where all the qubits are permuted by π_t, and all the gates in L_t now act on consecutive qubits.

By Lemma 2.2.10, there exists a circuit B_{π_t} with the architecture $\mathcal{B}_\ell^{\times \ell}$ for $\ell = \log n$ that implements the permutation π_t. Replace each layer L_t in C by the following subcircuit K_t: first apply $B_{\pi_t \pi_t^{-1}}$ and then apply the layer $\pi_t(L_t)$. Since the last layer of $B_{\pi_t \pi_t^{-1}}$ and $\pi_t(L_t)$ have the same architecture, we can merge the gates into a single layer. Here we assume $\pi_0 = \mathbb{1}$.

The final C' is the composition of the subcircuits $K_1, K_2, \ldots, K_{D_1}$, yielding a depth $O(D \log^2 n)$ circuit. Note that by induction, circuit C' is exactly equivalent to the original circuit C. Notice that each subcircuit K_t can be implemented as nearest-neighbor gates on a hypercube of dimension $\log n$, and thus the same holds for C' as well.

Let D' denote the depth of circuit C'. We consider spacetime Hamiltonians $H_{circuit}[C']$ of the circuit C', as described in Section 2.2. We first note that it has the following properties: it is a 9-local Hamiltonian, the terms act locally on a $O(D' + \log n)$-dimensional lattice, and each qubit participates in at most $O(D')$ terms.

2.3 Construction of the code Hamiltonian

Here we describe our code construction in detail. Let $\varepsilon > 0$ be the desired target approximation error. Let n, k, d be integers satisfying Theorem 2.2.2 where $k, d = \Omega(n)$. Let C_0 denote a Clifford circuit of depth $D_0 = O(\log^3 n)$ that is an encoding circuit of an $[[n,k,d]]$ code C_{BF}, as promised by Theorem 2.2.2. Let

$L_1, \ldots, L_{D_0}$ be the D_0 layers of C_0, where each L_t is a set of at $n/2$ two-qubit Clifford gates.

The first preprocessing step is to replace all the Clifford gates by gates from the set

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \qquad S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \qquad CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2.3.1}$$

This is possible because the gate set $\{I, H, S, CNOT\}$ generates the Clifford group; thus every two-qubit Clifford gate can be written as a $O(1)$-length product of I, H, S, and $CNOT$ gates. The depth of this circuit is $D_1 = O(D_0)$. Let C_1 denote this circuit.

Next, we *pad* the circuit to have depth $3D_1/\varepsilon$ where the last $1 - (\varepsilon/3)$ fraction of the layers are simply applications of the identity gate on consecutive pairs of qubits. Call this padded circuit C_1'; its depth is $D_1' = 3D_1/\varepsilon$.

Now, let C_2 be the circuit obtained by preprocessing C_1' as described in Section 2.2. This has depth $D = O(\log^5 n)$. Let $H_{circuit}[C_2]$ denote the corresponding spacetime circuit Hamiltonian, acting on $N = O(nD)$ qubits. For what follows, we will abbreviate $H_{circuit}[C_2]$ as H.

Let C denote the ground space of H. This will be our code. We now show that C is an approximate QLDPC code, and we establish its parameters.

Theorem 2.3.1. *For all $\varepsilon > 0$, the subspace C is a $[[N, k, d, \varepsilon, \ell, s]]$ approximate QLDPC code, for $k = \Omega(N/\log^5 N)$, $d = \Omega(N/\log^5 N)$, $\ell = 9$, and $s = \text{polylog}(N)$.*

Proof. First we have to show that C is the image of an encoding map, Enc. We present methods for efficiently generating a codeword of the code C in Section 2.3.

Next, we present a recovery map for the code (i.e. a map that approximately corrects errors and decodes). An important point is that the Brown-Fawzi stabilizer code underlying our construction was probabilistically chosen and there is no known efficient correction algorithm for their code. However, since the stabilizer code encoded by the circuit C satisfies the Knill-Laflamme error correction conditions [60], there exists an ideal recovery map, Rec_{BF}, that can correct any error on $(d-1)/2$ qubits or less. In other words, for all errors $\mathcal{E}$ acting on at most $(d-1)/2$ qubits, the following is equivalent to the identity channel on k qubits:

$$\text{Rec}_{BF} \circ \mathcal{E} \circ \text{Enc}_{BF} \tag{2.3.2}$$

where Enc_{BF} is the encoding map for the Brown-Fawzi code. This is all that we will need.

Our recovery map Rec for our code works as follows: given an input state on registers $\mathsf{S}_1 \cdots \mathsf{S}_n$ and $\mathsf{T}_1 \cdots \mathsf{T}_n$ (i.e. the data and time registers), it

1. Traces out the registers $\mathsf{T}_1 \cdots \mathsf{T}_n$.
2. Applies the Brown-Fawzi ideal recovery map Rec_{BF} to $\mathsf{S}_1 \cdots \mathsf{S}_n$.

We now prove the approximate error correction condition. We rely on the following Lemma, which we prove in Appendix 2.7.

Lemma 2.3.2. *Let $\mathcal{T}$ denote the set of all time configurations of a spacetime history state. There exists a subset $\mathcal{T}_{comp} \subset \mathcal{T}$ such that for all spacetime history states*

$$|\psi\rangle = \frac{1}{\sqrt{|\mathcal{T}|}} \sum_{\boldsymbol{\tau} \in \mathcal{T}} |\boldsymbol{\tau}\rangle \otimes |\psi_{\boldsymbol{\tau}}\rangle \tag{2.3.3}$$

there exists a codeword $|\Gamma\rangle \in C_{BF}$ such that if $\boldsymbol{\tau} \in \mathcal{T}_{comp}$, then $|\psi_{\boldsymbol{\tau}}\rangle = |\Gamma\rangle$. Furthermore, we have that

$$\frac{|\mathcal{T}_{comp}|}{|\mathcal{T}|} \geqslant 1 - \varepsilon. \tag{2.3.4}$$

Recall that C_{BF} is the $[[n, k, d]]$ stabilizer code[10] whose encoding map is the circuit C_0 described above.

Let $|\phi\rangle \in (\mathbb{C}^2)^{\otimes k} \otimes \mathcal{R}$ be a k-qubit message ρ that has been purified (i.e., $\rho = \mathrm{Tr}_{\mathcal{R}}(|\phi\rangle\langle\phi|)$). Let a Schmidt decomposition of $|\phi\rangle$ be $\sum_i \sqrt{p_i}|\xi_i\rangle|i\rangle$, where the $\{|\xi_i\rangle\}$ correspond to the Hilbert space $(\mathbb{C}^2)^{\otimes k}$ and the $\{|i\rangle\}$ are orthonormal vectors in $\mathcal{R}$. Let $|\Psi\rangle\langle\Psi| = \mathrm{Enc}(|\phi\rangle\langle\phi|)$, so that $|\Psi\rangle = \sum_i \sqrt{p_i}|\Psi_i\rangle|i\rangle$ where $|\Psi_i\rangle = \frac{1}{\sqrt{T}} \sum_{\boldsymbol{\tau}} |\boldsymbol{\tau}\rangle \otimes |\psi_{i,\boldsymbol{\tau}}\rangle$ is the spacetime history state for circuit C on input state $|\xi_i\rangle \otimes |0^{(n-k)}\rangle$. By Lemma 2.3.2 we can write

$$|\Psi_i\rangle = \frac{1}{\sqrt{|\mathcal{T}|}} \sum_{\boldsymbol{\tau} \notin \mathcal{T}_{comp}} |\boldsymbol{\tau}\rangle \otimes |\psi_{i,\boldsymbol{\tau}}\rangle + \frac{1}{\sqrt{|\mathcal{T}|}} \sum_{\boldsymbol{\tau} \in \mathcal{T}_{comp}} |\boldsymbol{\tau}\rangle \otimes |\Gamma_i\rangle. \tag{2.3.5}$$

Define the following (subnormalized) states:

$$|\lambda\rangle = \frac{1}{\sqrt{|\mathcal{T}|}} \sum_{\boldsymbol{\tau} \in \mathcal{T}_{comp}} |\boldsymbol{\tau}\rangle, \qquad |\widetilde{\Psi_i}\rangle = |\lambda\rangle \otimes |\Gamma_i\rangle. \tag{2.3.6}$$

Note that $|\widetilde{\Psi_i}\rangle$ has norm equal to $\||\lambda\rangle\|^2 \geqslant 1-\varepsilon$ because of Lemma 2.3.2. Furthermore, $|\langle\widetilde{\Psi_i}|\Psi_i\rangle|^2 = \||\lambda\rangle\|^2$. If we define $|\widetilde{\Psi}\rangle = \sum_i \sqrt{p_i}|\widetilde{\Psi_i}\rangle|i\rangle$, then we have that

$$F(|\Psi\rangle\langle\Psi|, |\widetilde{\Psi}\rangle\langle\widetilde{\Psi}|) \geqslant (1-\varepsilon)^2 \geqslant 1 - 2\varepsilon. \tag{2.3.7}$$

[10] The BF subscript stands for "Brown-Fawzi".

Let $\mathcal{E}$ be a completely positive, trace preserving map acting on at most $(d-1)/2$ qubits. Since C_{BF} is a code that can correct up to $(d-1)/2$ errors, and $|\widetilde{\Psi}\rangle$ is a (sub-normalized) superposition of codewords of C_{BF} (along with a state $|\lambda\rangle$ that gets traced out by Rec), we have that $\text{Rec} \circ \mathcal{E}(|\widetilde{\Psi}\rangle\langle\widetilde{\Psi}|) = |\phi\rangle\langle\phi| \cdot \langle\lambda|\lambda\rangle$. Since the fidelity metric is non-decreasing under quantum operations, we have that

$$F\left(\text{Rec} \circ \mathcal{E} \circ \text{Enc}(|\phi\rangle\langle\phi|), |\phi\rangle\langle\phi|\right) \geqslant F\left(\text{Rec} \circ \mathcal{E}(|\Psi\rangle\langle\Psi|), \text{Rec} \circ \mathcal{E}(|\widetilde{\Psi}\rangle\langle\widetilde{\Psi}|)\right) \tag{2.3.8}$$

$$\geqslant F\left(|\Psi\rangle\langle\Psi|, |\widetilde{\Psi}\rangle\langle\widetilde{\Psi}|\right) \tag{2.3.9}$$

$$\geqslant 1 - 2\varepsilon. \tag{2.3.10}$$

As discussed in Section 2.2, the geometry underlying the Hamiltonian H is a lattice with dimension $O(D \text{ polylog } n)$; each 9-local term acts in a spatially-local manner on this lattice, and each qubit participates in polylog n terms. This establishes the Theorem. □

Encoding circuit

We demonstrate that there is an efficient circuit generating a ground-state of the Hamiltonian.

Theorem 2.3.3. *There exists an encoding circuit of polynomial size in n which on input* $|\psi\rangle$ *generates the state* $\text{Enc}(\psi)$. *In particular, the polynomial size circuit generating the state is* $\log(n) + 2$ *spatially local.*

Proving the generability of the ground-state is done in two parts. We first show that once one can generate a particular superposition over the time registers, one can generate the ground-state. Next, we provide an efficient algorithm for generating the particular superposition. This is encapsulated formally in Lemmas 2.3.6 and 2.3.7.

The superposition over the time registers (the union of all clock and flag registers) of interest is the uniform superpositions over *valid partially applied configurations* – or *valid configurations*, for brevity - of an architecture[11]. Imagine progressively applying a circuit from an architecture, gate by gate. Non-commutativity of gates in different layers demands that the gates of the circuit cannot be applied in any order, but must be applied in a way that respects causality.

Definition 2.3.4 (Partial configuration of an architecture)**.** A partial configuration of an architecture on n qubits of depth D, is a vector of integers describing how many layers of gates have been applied per qubit: $\tau \in Z_{D+1}^n$. A partial configuration is *valid* if it respects the causal dependence of the gates in the circuit.

[11] A formal definition of an architecture is given as Definition 2.2.6.

Formally, consider a gate g at depth d_g acting on qubits i and j as *applied* if $d_g \leqslant \tau_i$ and $d_g \leqslant \tau_j$. Then an architecture is valid if for every marked gate g, any gate g' such that $g' \to g$ in the DAG represented the architecture (see Definition 2.2.6) is also marked.

Notationally, we refer to a qubit i being at time $t = \tau_i$.

Specifically, we are interested in generating the uniform superposition over valid configurations of the architecture behind the circuit C_2 from Section. We note that the architecture is similar to the product of bitonic block architectures (see Definition 2.7.12), $\mathcal{B}_\ell^{\times m}$ for $m = D_0', \ell = O(\log^4 n/\varepsilon)$, and $\ell = \log n$. However, on closer inspection, since the code Hamiltonian includes terms that check the consistency between clocks at the final time state and the initial time state, this does not exactly correspond to the spacetime Hamiltonian construction from a linear product of bitonic blocks. Rather, it corresponds to the spacetime Hamiltonian construction from a "circular" product of bitonic blocks. The set of valid configurations for this Hamiltonian includes configurations which "wrap around" the final time state and back to the initial time state. Formally,

Definition 2.3.5 (Valid configurations of a circular architecture)**.** Let $\mathcal{A}$ be an architecture on n qubits of depth D. Let $\mathcal{A}^{\times\infty}$ be the infinite circuit defined by taking infinite consecutive copies of $\mathcal{A}$:

$$\mathcal{A}^{\times\infty} = \prod_{j=-\infty}^{\infty} \mathcal{A}. \tag{2.3.11}$$

Let $\mathcal{V}_\infty \subset \mathbb{Z}^n$ be the set of valid configurations for $\mathcal{A}^{\times\infty}$. Define $\mathcal{V} \in \mathbb{Z}_D^n$ by $\mathcal{V} = \mathcal{V}_\infty / D\mathbb{Z}^n$, i.e. identity identical time configurations in the infinite copies. The set of valid configurations for the circular architecture is $\mathcal{V}$.

We call this a *circular architecture*, $\mathcal{B}_\ell^{\leftrightarrow m}$ and describe it with more formality in the Appendix (Definition 2.7.14).

Lemma 2.3.6. *Let $|\Xi\rangle$ be the uniform superposition over valid configurations of the architecture $\mathcal{B}_\ell^{\leftrightarrow m}$. Formally, let $\mathcal{V} \subseteq Z_{m\ell}^n$ be the set of valid configurations. Then,*

$$|\Xi\rangle = \frac{1}{\sqrt{|\mathcal{V}|}} \sum_{(t_1,\ldots,t_n)\in\mathcal{V}} |t_1\rangle_{\mathsf{T}_1} |t_2\rangle_{\mathsf{T}_2} \ldots |t_n\rangle_{\mathsf{T}_n}. \tag{2.3.12}$$

The state $|\Xi\rangle$ can be generated efficiently.

Proof. By Theorem 2.7.15, the number of valid configurations, $a_\ell^{\leftrightarrow m} \stackrel{\text{def}}{=} |\mathcal{V}|$ has a recursive definition and is at most doubly exponential in ℓ. Therefore, it can be

calculated in time poly(n). There exists an enumeration bijection $f : [a_\ell^{\leftrightarrow m}] \to \mathcal{V}$ which is (classically) efficient such that f^{-1} is also (classically) efficient. We extend this to reversible operations

$$f(|j\rangle|c\rangle) = |j\rangle|c \oplus f(j)\rangle \quad \text{and} \quad f^{-1}(|j\rangle|c\rangle) = |j \oplus f^{-1}(c)\rangle|c\rangle. \tag{2.3.13}$$

As a consequence, we can create $|\Xi\rangle$ by starting[12] with

$$\frac{1}{\sqrt{a_\ell^{\leftrightarrow m}}} \sum_{j=1}^{a_\ell^{\leftrightarrow m}} |j\rangle|0\rangle \tag{2.3.14}$$

and applying f followed by f^{-1}. A construction of f and f^{-1} is given as Theorem 2.7.26 in the Appendix.

□

Lemma 2.3.7. *Given the state $|\Xi\rangle$ (2.3.12) and an initial state $|\psi\rangle \in (\mathbb{C}^2)^{\otimes k}$, one can efficiently generate the state*

$$|\Psi\rangle = \frac{1}{\sqrt{\mathcal{V}}} \sum_{(t_1,\ldots,t_n)\in\mathcal{V}} |t_1\rangle_{\mathsf{T}_1}|t_2\rangle_{\mathsf{T}_2}\ldots|t_n\rangle_{\mathsf{T}_n} \otimes U_{t_1,\ldots,t_n}|\psi\rangle_{\mathsf{S}_1\mathsf{S}_2\ldots\mathsf{S}_k}|0^{n-k}\rangle_{\mathsf{S}_{k+1}\mathsf{S}_{k+2}\ldots\mathsf{S}_n} \tag{2.3.15}$$

where $U_{t_1,\ldots,t_n}$ is the unitary acting on $(\mathbb{C}^2)^{\otimes n}$ defined by the action of the valid configuration $(t_1,\ldots,t_n)$. The efficient generating circuit is spatially local in $2+\log_2(n)$ dimensions.

Proof. We describe three methods for efficiently constructing the state $|\Xi\rangle$ (2.3.12). The first describes a quantum circuit, the second is approximate and relies on phase estimation, and the third is based on adiabatic computation. We describe the first in detail and provide sketches for the other two.

Notationally, we will let T be the union of registers $\{\mathsf{T}_i\}$ and similarly define the registers C, F and S.

Method 1 (quantum circuit). Let C_2 be the circuit from which the spacetime circuit Hamiltonian is built. We modify the circuit C_2 into a new circuit C_2' which acts in one additional dimension such that

$$C_2'\left(|\Xi\rangle_{\widetilde{\mathsf{T}}} \otimes |\psi\rangle_{\mathsf{S}_1\mathsf{S}_2\ldots\mathsf{S}_k}|0^{n-k}\rangle_{\mathsf{S}_{k+1}\mathsf{S}_{k+2}\ldots\mathsf{S}_n} \otimes |0\rangle_{\mathsf{T}}\right) = |0\rangle_{\widetilde{\mathsf{T}}} \otimes |\Psi\rangle_{\mathsf{ST}}. \tag{2.3.16}$$

[12] The state $\frac{1}{\sqrt{b_k}}\sum_{j=1}^{a_\ell^{\leftrightarrow m}}|j\rangle$ can be generated efficiently by the following. Let W be the largest power of 2 greater than $a_\ell^{\leftrightarrow m}$. Using Hadamard gates, we can generate the superposition $\frac{1}{\sqrt{W}}\sum_{j=1}^{W}|j\rangle|0\rangle$. Then, we apply the reversible operation $|j\rangle|z\rangle \mapsto |j\rangle|z \oplus 1_{j\leqslant b_k}\rangle$ and measure the ancilla in the standard basis. If the measurement is 1, we achieve the desired state. The measurement 0 will occur with probability $\leqslant 1/2$.

Here the register $\widetilde{\mathsf{T}}$ is a copy of the register T. The additional dimension of C_2' over C_2 is the additional interaction with the register $\widetilde{\mathsf{T}}$. Each sub-register $\widetilde{\mathsf{T}}_i$ will interact with register T_i as well as any register $\widetilde{\mathsf{T}}_j$ for register T_j that T_i interacts with.

For every gate g in C_2 at depth t acting on qubit registers S_i and S_j, we replace g with a constant depth circuit acting on $\mathsf{S}_i, \mathsf{S}_j, \mathsf{T}_i, \mathsf{T}_j, \widetilde{\mathsf{T}}_i, \widetilde{\mathsf{T}}_j$. The constant depth circuit applies the following map: On input $|\psi\rangle_{\mathsf{S}_i\mathsf{S}_j}|t_i\rangle_{\mathsf{T}_i}|t_j\rangle_{\mathsf{T}_j}|T_i\rangle_{\widetilde{\mathsf{T}}_j}|T_j\rangle_{\widetilde{\mathsf{T}}_j}$, if $t_i < T_i$ and $t_j < T_j$, then the gate g is applied to $|\psi\rangle_{\mathsf{S}_i\mathsf{S}_j}$ and the registers T_i and T_j are incremented. Otherwise, the identity map is applied. Equivalently,

$$|\psi\rangle_{\mathsf{S}_i\mathsf{S}_j}|t_i\rangle_{\mathsf{T}_i}|t_j\rangle_{\mathsf{T}_j}|T_i\rangle_{\widetilde{\mathsf{T}}_j}|T_j\rangle_{\widetilde{\mathsf{T}}_j} \mapsto \begin{cases} g|\psi\rangle_{\mathsf{S}_i\mathsf{S}_j}|t_i+1\rangle_{\mathsf{T}_i}|t_j+1\rangle_{\mathsf{T}_j}|T_i\rangle_{\widetilde{\mathsf{T}}_j}|T_j\rangle_{\widetilde{\mathsf{T}}_j} & \text{if } t_i < T_i \wedge t_j < T_j \\ |\psi\rangle_{\mathsf{S}_i\mathsf{S}_j}|t_i\rangle_{\mathsf{T}_i}|t_j\rangle_{\mathsf{T}_j}|T_i\rangle_{\widetilde{\mathsf{T}}_j}|T_j\rangle_{\widetilde{\mathsf{T}}_j} & \text{otherwise.} \end{cases} \tag{2.3.17}$$

Notice that making this adjustment to every gate g in C_2 will on input $|T_1\rangle_{\widetilde{\mathsf{T}}_1}|T_n\rangle_{\widetilde{\mathsf{T}}_n}\ldots|T_n\rangle_{\widetilde{\mathsf{T}}_n}$, generate the partial computation of C_2 up to $(T_1,\ldots,T_n)$. Furthermore, in the end, the registers T and $\widetilde{\mathsf{T}}$ will both contain $(T_1,\ldots,T_n)$. Then, we can apply the map $|a\rangle_{\widetilde{\mathsf{T}}}|b\rangle_{\mathsf{T}} \mapsto |a\oplus b\rangle_{\widetilde{\mathsf{T}}}|b\rangle_{\mathsf{T}}$ to erase the $\widetilde{\mathsf{T}}$ register.

By linearity, when ran on the input $|\Xi\rangle_{\widetilde{\mathsf{T}}}$, this will yield the state $|\Psi\rangle$.

There is one complication to consider. We must consider valid configurations which cross time $t = 0$ (i.e. some clocks are near the end while others are just starting). To fix this, we first preprocess register $\widetilde{\mathsf{C}}$ such that any register with $|T_i\rangle_{\widetilde{\mathsf{C}}_i}$ for $T_i < D/2$ is replaced with $T_i + D$. This ensures that all clock registers are in the range[13] $[D/2, 3D/2]$. We now perform the same adjustment to each gate g except we do it for gates of the circuit $C'C'$ (circuit repeated twice). We follow it with postprocessing to return all clock registers to between 0 and D.

Method 2 (phase estimation). Apply the original random Clifford circuit C_0 of depth D_0 to the computational qubits, to the form the a state with no entanglement between the clocks and the data qubits,

$$|\Psi'\rangle = |\Xi\rangle \otimes C_0|\psi\rangle_{\mathsf{S}_1\mathsf{S}_2\ldots\mathsf{S}_k}|0^{n-k}\rangle_{\mathsf{S}_{k+1}\mathsf{S}_{k+2}\ldots\mathsf{S}_n}. \tag{2.3.18}$$

Since the spacetime history state $|\Psi\rangle$ is padded to length $T = \mathrm{poly}(D_0)$ with identity gates, the overlap between these states is

$$|\langle\Psi|\Psi'\rangle|^2 \geqslant 1 - \left(\frac{D_0}{T}\right)^2 = 1 - O\left(\frac{1}{\mathrm{polylog}(n)}\right) \tag{2.3.19}$$

Since the spectral gap of the code Hamiltonian scales as $1/\mathrm{poly}(n)$, we can apply a phase estimation circuit U_{PE} to $|\Psi'\rangle$ that estimates the first $O(\log n)$ digits of the

[13] This ensure that all valid configurations are "consecutive" because the width of a configuration (Lemma 2.7.2) is much smaller than D.

energy of this state with respect to the code Hamiltonian. By the overlap calculation above this phase estimation yields an eigenvalue of 0 with probability $1 - o(1)$ and projects $|\Psi'\rangle$ into the ground space of the code Hamiltonian. Using the fact that distinct code words are orthogonal, this produces a state of the form

$$\sqrt{1-\varepsilon^2}|\Psi\rangle + \varepsilon|\Psi''\rangle \tag{2.3.20}$$

where $|\Psi''\rangle$ is contained in the code space and $\varepsilon \leqslant D_0/T \leqslant 1/\text{polylog}(n)$.

Method 3 (adiabatic computation). As described in Section 3.4 of [23], a standard way to turn a circuit Hamiltonian $H(U_1, ..., U_L)$ into a procedure for adiabatically preparing the ground state is to use a continuous family of circuit Hamiltonians $H(s) = H(U_1(s), ..., U_L(s))$ to define the adiabatic path. Here $s \in [0, 1]$ is a parameter such that $U_i(0) = I$ and $U_i(1) = U_i$ for each i. Since we use 2-qubit gates and $SU(4)$ is simply connected we may define $U_i(s) = U_i^s$ for each i. Every Hamiltonian in this continuous family has the same spectral gap, which we have shown is $1/\text{poly}(n)$ in Section 2.4, and so any rigorous version of the adiabatic theorem suffices to turn the initial ground state of $H(I, ..., I)$ into the ground state of $H(U_1, ..., U_L)$ in polynomial time. Alternatively, instead of the adiabatic theorem, one can discretize the adiabatic path into polynomially many steps and use phase estimation to move between consecutive steps, which suffices to prepare the ground state of $H(U_1, ..., U_L)$ with exponentially small error. □

2.4 Spectral gap analysis

Our analysis of the spectral gap of the spacetime circuit Hamiltonian begins with several standard steps that are applied to Feynman-Kitaev Hamiltonians [7] and their spacetime variants [23]. First one defines a global unitary rotation,

$$W = \sum_{\boldsymbol{\tau} \in \mathcal{T}} U(\boldsymbol{\tau} \leftarrow 0)|\boldsymbol{\tau}\rangle\langle\boldsymbol{\tau}| \tag{2.4.1}$$

which when applied to full Hamiltonian yields,

$$W^\dagger H W = H_{\text{init}} + \mathcal{L}_{\text{prop}} \otimes \mathbb{1} + H_{\text{causal}} \tag{2.4.2}$$

$$\mathcal{L}_{\text{prop}} = \sum_{\boldsymbol{\tau}, \boldsymbol{\tau}' \in \mathcal{T}} \left(|\boldsymbol{\tau}\rangle\langle\boldsymbol{\tau}| + |\boldsymbol{\tau}'\rangle\langle\boldsymbol{\tau}'| - |\boldsymbol{\tau}\rangle\langle\boldsymbol{\tau}'| - |\boldsymbol{\tau}'\rangle\langle\boldsymbol{\tau}| \right) \tag{2.4.3}$$

where $\mathcal{L}_{\text{prop}}$ is the combinatorial Laplacian of a graph with vertices corresponding to valid time configurations and edges connecting time configurations $\boldsymbol{\tau}, \boldsymbol{\tau}'$ that differ by the application of a 2-local gate. Since $[H_{\text{causal}}, H_{\text{prop}}] = 0$, any state with energy less than 1 will be in the ground space of H_{causal}.

Applying the argument from Section 3.1.4 of [7], the Hamiltonian (2.4.2) in the rotated frame only acts on the computational qubits through H_{in}; so the Hamiltonian

can be written in block diagonal form with each block B_i corresponding to a different input string $i = 0, ..., 2^n - 1$,

$$W^\dagger H W = \begin{pmatrix} B_0 & & & \\ & B_1 & & \\ & & \ddots & \\ & & & B_{2^n-1} \end{pmatrix} \tag{2.4.4}$$

The ground space of $W^\dagger H W$ is contained in the block B_0; therefore, the spectral gap of H will either be the spectral gap of Δ_{prop} within B_0, or it will be the minimum among the ground state energies in the other blocks B_i. To lower bound the ground state energies in the blocks $B_i \neq 0$ we can apply the geometrical lemma.

Kitaev's Geometrical lemma. Let $H_1, H_2 \geq 0$ be positive semi-definite operators with 0 as an eigenvalue, and let the least nonzero eigenvalue of H_1 and H_2 be lower bounded by Λ, then

$$H_1 + H_2 \geq \Lambda \sin^2\left(\frac{\theta}{2}\right) \quad , \quad \cos^2\theta = \max_{\substack{|\xi\rangle \in \text{Ker}(H_1) \\ |\eta\rangle \in \text{Ker}(H_2)}} \tag{2.4.5}$$

The lemma is applied in each of the blocks B_i with $i \neq 0$, taking $H_1 = H_{\text{in}}$ and $H_2 = \mathcal{L}_{\text{prop}} \otimes \mathbb{1}$. Since the spectral gap of H_{in} is 1 we take $\Lambda = \Delta_{\text{prop}}$, where Δ_{prop} is the spectral gap of $\mathcal{L}_{\text{prop}}$. The sine of the angle between the kernels of H_{in} and H_{prop} is lower bounded by the overlap of the ground state of H_{prop} with any one of the local terms in H_{in}, which is $1/T$. Therefore the spectral gap of the full Hamiltonian satisfies

$$\Delta_H = \frac{\Delta}{T} = \widetilde{\Omega}(\Delta_{\text{prop}}). \tag{2.4.6}$$

It remains to lower bound the spectral gap Δ of the graph Laplacian $\mathcal{L}_{\text{prop}}$. This graph Laplacian is a stoquastic frustration-free Hamiltonian with a uniform ground state in the time configuration basis, and so it can be mapped to a Markov chain transition matrix by shift and rescaling,

$$P = I - \mathcal{L}_{\text{prop}} / \|\mathcal{L}_{\text{prop}}\| \tag{2.4.7}$$

The transformation of a spacetime Hamiltonian H_{prop} into a Markov chain in [23] first maps the 1 + 1 dimensional spacetime Hamiltonian to the ferromagnetic Heisenberg chain and then relates the Heisenberg model to a Markov chain transition matrix by rescaling its operator norm. This multistep mapping reveals additional insights about the physics of that model, and is the basis for the gap analysis in [23], but the mapping from stoquastic Hamiltonians to Markov chains is entirely general as described in [33].

The operator norm satisfies $\|L_{\text{prop}}\| = \|H_{\text{prop}}\| = \Omega(n)$ and so in terms of the spectral gap Δ_P of the Markov chain (2.4.7) we have

$$\Delta_H = \Omega(n\Delta_P). \tag{2.4.8}$$

Preliminaries on Markov chains

Throughout this section let (Ω, π, P) be an irreducible, ergodic, reversible Markov chain on the state space Ω, with stationary distribution π and transition matrix P (see [65] for background on these terms).

Block decomposition method [68]. The state space is decomposed into subsets $\Omega = \cup_{i=1}^{R} \Omega_i$ ("blocks"), which in general will have nonempty pairwise intersection, $\Omega_i \cap \Omega_j \neq \emptyset$. Let $\Theta \stackrel{\text{def}}{=} \max_{x\in\Omega} |\{i : x \in \Omega_i\}|$ be the maximum numbers of sets that can contain any single element $x \in \Omega$. For any $S \subseteq \Omega$, define $\pi(S) \stackrel{\text{def}}{=} \sum_{x\in S} \pi(x)$. Define the aggregate ("block") Markov chain $\overline{P}$ on the state space $\{1, ..., R\}$,

$$\overline{P}_{ij} \stackrel{\text{def}}{=} \frac{\pi\left(\Omega_i \cap \Omega_j\right)}{\Theta\pi\left(\Omega_i\right)} \quad , \quad i, j \in \{1, ..., R\}. \tag{2.4.9}$$

One can easily check that these transition probabilities are reversible with respect to the distribution $\bar{\pi}_i \stackrel{\text{def}}{=} \pi(\Omega_i)$. Next define a restricted ("within-block") chain P_i for each subset Ω_i as follows: if $x \in \Omega_i$ and $y \neq x$ then

$$P_i(x, y) \stackrel{\text{def}}{=} \begin{cases} P(x, y) & y \in \Omega_i \\ 0 & y \notin \Omega_i \end{cases}, \tag{2.4.10}$$

and $P_i(x, x) \stackrel{\text{def}}{=} 1 - \sum_{y\neq x} P_i(x, y)$. The spectral gap of Δ_P satisfies the lower bound

$$\Delta_P \geqslant \frac{1}{2}\Delta_{\overline{P}} \min_{i=1,...,R} \Delta_{P_i} \tag{2.4.11}$$

Cheeger's inequality. For any nonempty subset $S \subseteq \Omega$ define the conductance $\Phi(S)$ by

$$\Phi(S) \stackrel{\text{def}}{=} \frac{1}{\pi(S)} \sum_{x\in S, y\in S^c} \pi_x P_{xy} \tag{2.4.12}$$

and define $\Phi_P \stackrel{\text{def}}{=} \min_{S:0<\pi(S)\leqslant 1/2} \Phi(S)$. Cheeger's inequality states that

$$\frac{\Phi_P^2}{2} \leqslant \Delta_P. \tag{2.4.13}$$

Decomposition of the circuit Propagation Markov chain

The subsets in our decomposition are defined by

$$\Omega_r \stackrel{\text{def}}{=} \{(t_1, ..., t_n) \in \Omega : r \leqslant t_i \leqslant r + \ell \text{ for all } i = 1, ..., n\} \quad , \quad r = 0, ..., D - \ell \tag{2.4.14}$$

(recall $\ell \stackrel{\text{def}}{=} \log(n)$). Every valid time configuration is contained in at least one Ω_r, so $\Omega = \cup_{r=0}^{D-r} \Omega_r$ as required. The maximum number of blocks that can contain any particular time configuration is $\Theta = k$; this maximum is attained by any configuration $(t_1, ..., t_n)$ for which $t_1 = ... = t_n = r$, with such configurations being contained in $\Omega_{r-\ell}, ..., \Omega_r$.

Next we compute the aggregate transition probabilities $\overline{P}_{ij}$. In particular, we will need the transition probability between consecutive blocks $P_{r,r+1}$. Since the distribution over time configurations is uniform, we have

$$\frac{\pi(\Omega_r \cap \Omega_{r'})}{\pi(\Omega_r)} = \frac{|\Omega_r \cap \Omega_{r'}|}{|\Omega_r|}$$

for all $r, r' = 0, \ldots, D - \ell$. To determine $|\Omega_r|$, we use the recursion relation $a_\ell = 2a_{\ell-1}^2 - a_{\ell-2}^4$ for the number of partially completed circuit configurations a_ℓ of a bitonic block of rank ℓ which is defined in Definition 2.2.7 and the recursive relation is proved in Theorem 2.7.10). This recursion relation has been studied previously [62] and has the asymptotic solution $a_\ell = \phi^{-1}\omega^{2^\ell}$, where $\phi = (1 + \sqrt{5})/2$ is the golden ratio and $\omega = 1.8445...$ does not have a known closed form. Next in Appendix 2.7 we show that $|\Omega_r| = |\Omega_{r'}|$ for every pair of blocks r, r', which follows from the fact that the valid configurations of any circuit architecture are invariant under permutation of the qubit labels, together with an explicit set of permutations we define that relates the architecture in each block. Therefore we have $|\Omega_r| = a_\ell = \phi^{-1}\omega^{2^\ell}$ for all $r = 0, ..., D - \ell$.

To evaluate (2.4.9) we also need to count the number of configurations contained in the intersection of two such consecutive blocks, see Figure 2.8. The key insight is that removing either the first or last layer of any block will split it into two independent bitonic blocks on half the number of qubits, which implies that $|\Omega_r \cap \Omega_{r+1}| = a_{\ell-1}^2$ and so

$$\frac{\pi(\Omega_r \cap \Omega_{r+1})}{\pi(\Omega_r)} = \frac{a_{\ell-1}^2}{a_\ell} = \phi^{-2} \quad , \quad r = 0, ..., T - \ell. \tag{2.4.15}$$

and similarly,

$$\frac{\pi(\Omega_r \cap \Omega_{r+j})}{\pi(\Omega_r)} = \frac{a_{\ell-j}^{2^j}}{a_\ell} = \phi^{-2^j} \quad , \quad r = 0, ..., T - \ell. \tag{2.4.16}$$

and so the aggregate transition probabilities decay doubly exponentially with distance,

$$\overline{P}(r, r + j) = \ell^{-1}\phi^{-2^j} \quad , \quad r = 0, ..., D - \ell. \tag{2.4.17}$$

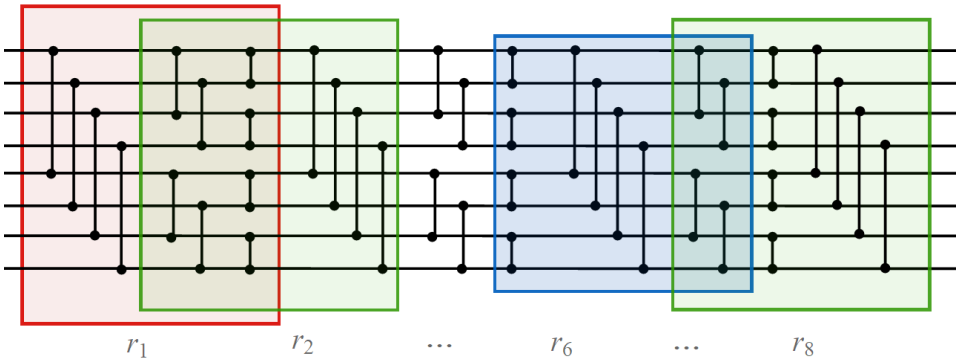


Figure 2.8: The region in the intersection of the red and green bitonic blocks $\mathcal{B}_\ni$ contains the time configurations that belong to $\Omega_{r_1} \cap \Omega_{r_2}$. The key insight is that the valid time configurations in the intersection of these blocks can be counted by observing that the intersection corresponds to two independent copies of $\mathcal{B}_\in$. Similarly, the configurations contained in the intersection $\Omega_6 \cap \Omega_8$ correspond to 4 independent copies of $\mathcal{B}_1$.

Next we lower bound the minimum conductance $\Phi_{\overline{P}}$. Let S be a nonempty subset of blocks, $S \subset \{0, ..., D-\ell\}$. Define $\pi(S) \stackrel{\text{def}}{=} \sum_{r\in S} \pi(\Omega_r)$, and assume $\pi(S) \leqslant 1/2$. There must be some $r'' < D-\ell$ such that $r'' \notin S$, and so

$$\frac{1}{\pi(S)} \sum_{r\in S, r'\in S^c} \pi(\Omega_r)\overline{P}_{r,r'} \geqslant 2\ell^{-1}\pi(\Omega_{r''-1}\cap\Omega_{r''}) = 2\ell^{-1}\frac{a_{\ell-1}^2}{|C|} \geqslant 2\ell^{-1}(m\ell+1)^{-1}\phi^{-2}. \tag{2.4.18}$$

Therefore, Cheeger's inequality yields

$$\Delta_{\overline{P}} \geqslant 2\ell^{-2}(m\ell+1)^{-2}\phi^{-4} = \frac{1}{\text{polylog}(n)}. \tag{2.4.19}$$

It remains to lower bound the spectral gaps Δ_{P_i} corresponding to the restricted ("within-block") chains defined in (2.4.10). By our careful choice of the block decomposition, we demonstrate in Section 2.7 a one-to-one correspondence between the time configurations in any block Ω_i and the equal area dyadic tilings of a unit square, and crucially this correspondence also exactly maps the edge-flip Markov chain moves considered in [27] to the updates which describe the application of a local gate to a valid time configuration. Since the relaxation time of the edge-flip Markov chain is $O(n^{4.09})$ we have $\Delta_{P_i} = \Omega(n^{-4.09})$ and so

$$\Delta_P = \widetilde{\Omega}(n^{-4.09}), \tag{2.4.20}$$

and by (2.4.8) this implies

$$\Delta_H = \widetilde{\Omega}(n^{-3.09}). \tag{2.4.21}$$

2.5 Local detection of Pauli errors

In this section we describe the local detection of errors on spacetime codewords with probability $1 - 2^{-\text{polylog}(N)}$ with polylog(N)-depth circuits. The class of errors

that we handle is the set of tensor products of Pauli operators on the physical qubits (which includes data and time qubits). Interestingly, we can detect Pauli errors even if the weight of the error (the number of qubits affected) exceeds the distance of the spacetime code! Here we only describe a single round of error detection while assuming the ability to perform measurements implemented by low-depth circuits perfectly.

Definition 2.5.1 (Pauli group)**.** The *Pauli group on N qubits*, denoted by $\mathcal{P}_N$, is the group generated by the N-fold tensor product of the Pauli matrices

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad \sigma_X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \sigma_Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad \sigma_Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{2.5.1}$$

along with multiplication by $\{\pm 1, \pm i\}$.

Definition 2.5.2 (Pauli channels)**.** A quantum operator $\mathcal{E}$ acting on N qubits is a *Pauli channel* if it has a Kraus decomposition

$$\mathcal{E}(\rho) = \sum_{P \in \mathcal{P}_N} c_P \, P \rho P^\dagger \tag{2.5.2}$$

where $\{c_P\}$ is a probability distribution over $\mathcal{P}_N$.

Pauli stabilizers of the spacetime code

There are nonidentity elements $P \neq I$ of the Pauli group $\mathcal{P}_N$ that stabilize the spacetime code, i.e., for all $|\psi\rangle \in C$, we have $P|\psi\rangle = |\psi\rangle$. In this section, we identify three stabilizers; in the next section, we will argue that these are the *only* nonidentity stabilizers, and all other nonidentity Pauli operators can be locally detected with high probability.

Let C denote the circuit such that the code Hamiltonian is $H_{circuit}[C]$, as described in Section 2.3. Recall that $X = \frac{D-2}{2}$ where D is the depth of C.

For any $p \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, z\}$ consider the set of 4 qubits $\mathsf{C}_{p,j}, \mathsf{C}_{q_1,j}, \mathsf{C}_{q_2,j}$, and $\mathsf{C}_{p',j}$ where q_1 is the qubit in layer L_j interacting with p, q_2 is the qubit in layer L_{2X+1-j} interacting with p, and p' is the qubit in layer L_{2X+1-j} interacting with q_1. Because the layers L_j and L_{2X+1-j} are different layers of the bitonic architecture, we know that these layers together form a product of bitonic blocks of rank 2, $\mathcal{B}_2$ (see Corollary 2.7.9). Then, it is easy to also see that p' is the qubit in layer L_j interacting with q_2. Define $rect(p, j)$ as the elements $\{p, q_1, q_2, p'\}$. It is not difficult to see that $rect(\cdot, j)$ yields the same set on inputs p, q_1, q_2, p'. Let the stabilizer $S_{clock}[rect(p, j)]$ be

$$S_{clock}[rect(p,j)] \stackrel{\text{def}}{=} \sigma_Z(\mathsf{C}_{p,j}) \otimes \sigma_Z(\mathsf{C}_{q_1,j}) \otimes \sigma_Z(\mathsf{C}_{q_2,j}) \otimes \sigma_Z(\mathsf{C}_{p',j}) \tag{2.5.3}$$

where $\sigma_Z(\mathsf{C}_{p,j})$ denotes the σ_Z operator acting on the clock qubit $\mathsf{C}_{p,j}$). Furthermore, let

$$S_{flag} \stackrel{\text{def}}{=} \bigotimes_{p=1}^{n} \sigma_Z(\mathsf{F}_p) \tag{2.5.4}$$

where $\sigma_Z(\mathsf{F}_p)$ denotes the σ_Z operator acting on the flag qubit corresponding to data qubit p. This is the product of σ_Z's acting on all the flag qubits.

Claim 2.5.3. *$S_{clock}[rect(p, j)]$ for any qubit p and $1 \leqslant j \leqslant X$ and S_{flag} are Pauli stabilizers of the spacetime code.*

Proof. Let $\boldsymbol{\tau} = (t_1, \ldots, t_n) \in \mathcal{T}$ be a valid time configuration of the spacetime history state.

Recall that every time configuration can be seen as the result of incrementing the clocks by applying gates. Therefore there is a sequence of time configurations $(\boldsymbol{\tau}_0, \boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_f = \boldsymbol{\tau})$ such that $\boldsymbol{\tau}_0$ has all clock and flag registers set to 0, and each $\boldsymbol{\tau}_{i+1}$ differs from $\boldsymbol{\tau}_i$ by the application of a gate. To each time configuration $\boldsymbol{\tau}_i$ we can associate a $b_{\boldsymbol{\tau}_i} \in \{\pm 1\}$ such that

$$S_{clock}[rect(p, j)]|\boldsymbol{\tau}_i\rangle = b_{\boldsymbol{\tau}_i}|\boldsymbol{\tau}_i\rangle. \tag{2.5.5}$$

Clearly $b_{\boldsymbol{\tau}_0} = 1$. We argue that $b_{\boldsymbol{\tau}_{i+1}} = b_{\boldsymbol{\tau}_i}$. Consider the gate differentiating these two configurations. Applying it must change the time registers by flipping the values of $\mathsf{C}_{r,j'}$ and $\mathsf{C}_{s,j'}$ (and perhaps the corresponding flag registers). This either flips the sign of $b_{\boldsymbol{\tau}_i}$ twice (if this gate is one of $rect(p, j)$) or not at all (if it is not). Therefore, $b_{\boldsymbol{\tau}_{i+1}} = b_{\boldsymbol{\tau}_i}$. This proves that $b_{\boldsymbol{\tau}} = 1$ and that $S_{clock}[rect(p, j)]$ is a stabilizer as

$$S_{clock}[rect(p, j)]|\psi\rangle = \frac{1}{\sqrt{|\mathcal{T}|}} \sum_{\boldsymbol{\tau} \in \mathcal{T}} S_{clock}[rect(p, j)]|\boldsymbol{\tau}\rangle \otimes |\psi_{\boldsymbol{\tau}}\rangle = |\psi\rangle. \tag{2.5.6}$$

A similar argument can be made showing that S_{flag} is also a stabilizer by arguing that either pair of flag qubits must be flipped or none are flip when transitioning from a valid time configuration to the next. Therefore, S_{flag} is also a stabilizer.

□

Let $\mathcal{S}$ be the closure of the following set under product,

$$\{I, S_{flag}\} \cup \bigcup_{p=1}^{n} \bigcup_{j=1}^{X} S_{clock}[rect(p, j)]. \tag{2.5.7}$$

Every element of $\mathcal{S}$ is a stabilizer.

Locally detecting errors

In this section, we argue that there is a set of local operators that can detect, with high probability, any Pauli error in $\mathcal{P}_N \setminus \mathcal{S}$.

Our argument will rely on a structural property of the Brown-Fawzi circuit C that holds with high probability (when the circuit is sampled according to the random Clifford model described in Section 2.2).

Definition 2.5.4. A depth D circuit on n qubit is *nice* if for every qubit $p \in \{1, \ldots, n\}$, there exists layers $1 \leqslant t, t' \leqslant D$ such that:

1. The two-qubit gate acting on p in layer L_t is $H \otimes I$, where the Hadamard gate H acts on qubit p, and
2. The two-qubit gate acting on p in layer $L_{t'}$ is $S \otimes I$, where the phase gate $S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$ acts on qubit p.

Fact 2.5.5 ([24])**.** *The Brown-Fawzi encoding circuit C sampled according to the random Clifford model described in Section 2.2 is nice with probability at least $1 - 2^{-\Omega(\log^2 n)}$.*

The following fact can be verified via a simple computation.

Fact 2.5.6. *Each of the following unitary operators has eigenvalues i and $-i$:*

1. $H\sigma_X H\sigma_X$
2. $H\sigma_Z H\sigma_Z$
3. $S^\dagger \sigma_Y^\dagger S \sigma_Y$

Here, H is the Hadamard gate, and S is the phase gate.

We now proceed to prove the local error detection property of the spacetime code.

Theorem 2.5.7. *Suppose the Brown-Fawzi encoding circuit defining the spacetime code C is nice. Then there exists a collection $\mathcal{D}$ of* polylog(N)*-local projectors satisfying the following properties:*

1. *Each projector $\Pi \in \mathcal{D}$ acts on* 10 *qubits of the code space, and acts on $s =$* polylog(N) *ancilla qubits initialized in the $|0\rangle$ state.*
2. *For all n-qubit states $|\psi\rangle$, we have that $\Pi|\psi\rangle|0^s\rangle = 0$ for all $\Pi \in \mathcal{D}$ if and only if $|\psi\rangle$ is a codeword in the spacetime code C.*

3. *For all Pauli channels $\mathcal{E}$, for all codewords $|\psi\rangle \in C$, there exists a projector $\Pi \in \mathcal{D}$ such that*

$$\mathrm{Tr}\left(\Pi\ (\mathcal{E}(\psi) \otimes |0^s\rangle\langle 0^s|)\right) \geqslant (1-\alpha)(1-2^{-\mathrm{polylog}(N)}) \tag{2.5.8}$$

where $\psi = |\psi\rangle\langle\psi|$ and $\alpha = \sum_{P\in\mathcal{S}} c_P$ is the weights of the channel $\mathcal{E}$ on the Pauli stabilizers in $\mathcal{S}$.

Furthermore, there exists a measurement M, implementable by a circuit of $\mathrm{polylog}(N)$ *depth acting on $O(N\,\mathrm{polylog}(N))$ qubits, such that for all Pauli channels $\mathcal{E}$ and for all codewords $|\psi\rangle \in C$*

$$\mathrm{Tr}\left(M\ \left(\mathcal{E}(\psi) \otimes |0^{Ns}\rangle\langle 0^{Ns}|\right)\right) \geqslant (1-\alpha)(1-2^{-\mathrm{polylog}(N)}). \tag{2.5.9}$$

Proof. We first define a set of projectors $\mathcal{D}_0$ that *weakly* detect errors, in the sense that for every Pauli channel $\mathcal{E}$, for every spacetime codeword $|\psi\rangle$, there is a projector $\Pi \in \mathcal{D}_0$ that has expectation value at least $(1-\alpha)/\mathrm{polylog}(N)$ on $\mathcal{E}(\psi)$. We will then boost the set $\mathcal{D}_0$ into the desired set $\mathcal{D}$ that detects errors with high probability, using QMA-amplification techniques.

A weak set of detector projections. The weak detection set $\mathcal{D}_0$ will simply be the set of local terms of the spacetime circuit Hamiltonian defining the spacetime code. We first show that for each member P of the Pauli group $\mathcal{P}_N$ that is not a Pauli stabilizer in $\mathcal{S}$, there exists a projector $\Pi \in \mathcal{D}_0$ such that

$$\mathrm{Tr}\left(\Pi P\psi P^\dagger\right) \geqslant \frac{1}{\mathrm{polylog}\, N}. \tag{2.5.10}$$

Fix a $P \in \mathcal{P}_N \setminus \mathcal{S}$, and fix a spacetime codeword $|\psi\rangle \in C$, which we can write as

$$|\psi\rangle = \frac{1}{\sqrt{|\mathcal{T}|}} \sum_{\boldsymbol{\tau}\in\mathcal{T}} |\boldsymbol{\tau}\rangle \otimes |\psi_{\boldsymbol{\tau}}\rangle. \tag{2.5.11}$$

We divide our analysis into several cases. Write $P = P_{flag} \otimes P_{clock} \otimes P_{data}$, where P_{flag} acts on the $\{\mathsf{F}_p\}$ registers, P_{clock} acts on the $\{\mathsf{C}_{p,j}\}$ registers and P_{data} acts on the $\{\mathsf{S}_p\}$ registers.

Case 1. Suppose that P_{clock} has a tensor factor that is either σ_X or $\sigma_Y = -i\sigma_Z\sigma_X$. In other words, there exists a data qubit $p \in \{1,\ldots,n\}$ and an associated clock qubit $j \in \{1,\ldots,X\}$ such that the tensor factor of P corresponding to the register $\mathsf{C}_{p,j}$ (i.e. the part of P acting on the j'th clock qubit of p's clock) is one of $\{\sigma_X, \sigma_Y\}$. We can write $P = P_Z P_X$ where P_Z consists of only σ_Z and identity factors and P_X consists only σ_X and identity factors. By assumption, P_X has at least one σ_X acting on a clock qubit.

Let $\rho = \mathrm{Tr}_{\overline{\mathsf{C}_p}}(P\psi P^\dagger)$ denote[14] the reduced density matrix of $P|\psi\rangle$ on the clock

[14] Here and throughout the paper, we use the notation $\overline{\mathsf{R}}$ to refer to all registers excluding R.

register C_p. Notice that $\rho = P_p \operatorname{Tr}_{\overline{\mathsf{C}_p}}(\psi) P_p^\dagger$ where P_p is the restriction of P to the qubits of the C_p register. We now appeal to the following Lemma, which we prove in the Appendix as Lemma 2.7.17:

Lemma 2.5.8. *The marginal distribution of the clock register C_p of any data qubit p in a spacetime codeword is uniform over the $X + 1 = D/2$ states*

$$|0^X\rangle, |10^{X-1}\rangle, \ldots, |1^X\rangle. \tag{2.5.12}$$

Lemma 2.5.8 implies that $\operatorname{Tr}_{\overline{\mathsf{C}_p}}(\psi) = \frac{1}{X+1} \sum_{t=0}^{X} |1^t 0^{X-t}\rangle\langle 1^t 0^{X-t}|$. Since this is a convex combination over standard basis states, we have that

$$\rho = P_X \left(\frac{1}{X+1} \sum_{t=0}^{X} |1^t 0^{X-t}\rangle\langle 1^t 0^{X-t}| \right) P_X^\dagger. \tag{2.5.13}$$

It is easy to see that for all $P_X \neq I$, we have that there is at least one $0 \leqslant t \leqslant X$ such that $P_X|1^t 0^{X-t}\rangle$ is not a valid clock state – that is, there is a location $j \in \{1, \ldots, X\}$ such that

$$\operatorname{Tr}\left(|01\rangle\langle 01|_{\mathsf{C}_{p,j}\mathsf{C}_{p,j+1}} P_X |1^t 0^{X-t}\rangle\langle 1^t 0^{X-t}| P_X^\dagger \right) = 1. \tag{2.5.14}$$

Notice that the projector $\Pi_j = |01\rangle\langle 01|_{\mathsf{C}_{p,j}\mathsf{C}_{p,j+1}}$ is precisely one of the terms in the spacetime Hamiltonian (see (2.2.9)). Thus we have that

$$\operatorname{Tr}(\Pi_j \, P\psi P^\dagger) = \operatorname{Tr}(\Pi_j \, \rho) \geqslant \frac{2}{D} = \Omega\left(\frac{1}{\log^5 N} \right). \tag{2.5.15}$$

Case 2. Suppose that P_{clock} only has σ_Z or identity factors and P_{flag} has a tensor factor that is either σ_X or $\sigma_Y = -i\sigma_Z\sigma_X$. In other words, there exists a data qubit $p \in \{1, \ldots, n\}$ such that the associated tensor factor of P corresponding to the register F_p is one of $\{\sigma_X, \sigma_Y\}$. We can write $P_{flag} \otimes P_{clock} = P_Z P_X$ where P_Z consists of (up to multiplication by $\{1, -1, i, -i\}$) only σ_Z and identity factors and P_X consists only σ_X and identity factors. By assumption, P_X has at least one σ_X acting on a flag qubit.

Case 2.1. We first consider a subcase that P_X includes as a factor the operator

$$T_{flag} = \bigotimes_p \sigma_X(\mathsf{F}_p). \tag{2.5.16}$$

In other words, P_X flips every flag qubit. This maps every valid time configuration $\boldsymbol{\tau} = (t_1, \ldots, t_n)$ to a "mirror" time configuration $\overline{\boldsymbol{\tau}} = (\bar{t}_1, \ldots, \bar{t}_n)$ where $\bar{t}_j = D - 1 - t_j$ according to the mapping described in (2.2.4). Mirror time configurations are also valid time configurations (i.e. they satisfy the causality constraints of the spacetime Hamiltonian).

We argue that these mirror time configurations, combined with the state of the data qubits, cannot satisfy the propagation constraints of the spacetime Hamiltonian. To see this, suppose that the circuit C had the following subcircuit J appended to both the beginning and end of the circuit C. The subcircuit J consists of two bitonic block architectures $\mathcal{B}_\ell$, wherein each block all the gates are identity gates except for the last layer, which is populated with $\sigma_X \otimes \sigma_X$ gates acting on each neighboring pair of qubits. Thus the subcircuit J is equivalent to the identity circuit because the two layers of σ_X gates cancel each other out. Appending J to the beginning and end of the circuit C yields a circuit with a small increase in depth, and it can be checked that this does not qualitatively affect the analysis of the spacetime Hamiltonian. Thus we will assume that our circuit C has this structure.

The circuit C acts on n qubits, $n-k$ of which are ancilla qubits that are initialized in all the all zeroes state. Let p denote an ancilla qubit. Let $\boldsymbol{\tau} = (t_1, \ldots, t_n)$ be any valid time configuration such that $t_p = \ell - 1$. Since this time is before the first row of σ_X gates in the circuit C, qubit p in $|\psi_{\boldsymbol{\tau}}\rangle$ is in the state $|0\rangle$. The σ_X gates get applied in the transition from time $\ell - 1$ to ℓ, so qubit p in $|\psi'_{\boldsymbol{\tau}}\rangle$ is in the state $|1\rangle$, where $\boldsymbol{\tau}'$ is the time configuration obtained from $\boldsymbol{\tau}$ by applying the two-qubit $\sigma_X \otimes \sigma_X$ gate to qubit p and its neighbor.

Now consider the mirror time configurations $\overline{\boldsymbol{\tau}} = (\bar{t}_1, \ldots, \bar{t}_n)$, and $\overline{\boldsymbol{\tau}}' = (\bar{t}'_1, \ldots, \bar{t}'_n)$. We have that $\bar{t}_p = D - 1 - t_p = D - \ell - 1$ and $\bar{t}'_p = D - 1 - t'_p = D - \ell - 2$. The gate on qubit p corresponding between times $D - \ell - 2$ and $D - \ell - 1$ is an identity gate (because we're assuming that the circuit C has the subcircuit J at the end).

Let $\Pi = H_t[p, q]$ for $t = D - \ell - 1$. This projector acts as the identity on the S register. Observe that

$$P|\psi\rangle = \frac{1}{\sqrt{|\mathcal{T}|}} \sum_{\boldsymbol{\tau} \in \mathcal{T}} P_Z|\overline{\boldsymbol{\tau}}\rangle \otimes P_{data}|\psi_{\boldsymbol{\tau}}\rangle \tag{2.5.17}$$

$$= \frac{1}{\sqrt{|\mathcal{T}|}} \sum_{\boldsymbol{\tau} \in \mathcal{T}} |\overline{\boldsymbol{\tau}}\rangle \otimes b_{\boldsymbol{\tau}} P_{data}|\psi_{\boldsymbol{\tau}}\rangle \tag{2.5.18}$$

for some $b_{\boldsymbol{\tau}} \in \{\pm 1\}$.

In what follows, we use the notation $\boldsymbol{\tau}[p, q]$ to denote the pair (t_p, t_q), and use $\boldsymbol{\tau} \to_{p,q} \boldsymbol{\tau}'$ to indicate that the time configuration $\boldsymbol{\tau}'$ is $\boldsymbol{\tau}$ updated by the gate $U_t[p, q]$. We now calculate the expectation

$$\langle\psi|P^\dagger \Pi P|\psi\rangle \tag{2.5.19}$$

$$= \frac{1}{|\mathcal{T}|} \sum_{\substack{\tau\in\mathcal{T}:\\ \overline{\tau}[p,q]=(t,t)\\ \overline{\tau}\to_{p,q}\overline{\tau}'}} \left(b_\tau \langle\psi_\tau| + b_{\tau'}\langle\psi_{\tau'}|\right) P_{data}^\dagger \Pi P_{data} \left(b_\tau|\psi_\tau\rangle + b_{\tau'}|\psi_{\tau'}\rangle\right) \tag{2.5.20}$$

$$= \frac{1}{|\mathcal{T}|} \sum_{\substack{\tau\in\mathcal{T}:\\ \overline{\tau}[p,q]=(t,t)\\ \overline{\tau}\to_{p,q}\overline{\tau}'}} \left(1 - b_\tau b'_\tau \mathrm{Re}\,\langle\psi_\tau|\psi_{\tau'}\rangle\right). \tag{2.5.21}$$

Observe that when $\overline{\tau}[p,q] = (t,t)$, we have that $\tau[p,q] = (D-1-t, D-1-t) = (\ell,\ell)$, and $\tau'[p,q] = (\ell-1,\ell-1)$. But from the reasoning above, we have that $|\psi_\tau\rangle$ and $|\psi_{\tau'}\rangle$ are orthogonal, because the state of qubit p of the two vectors are orthogonal. Therefore

$$\langle\psi|P^\dagger\Pi P|\psi\rangle = \sum_{\substack{\tau\in\mathcal{T}:\\ \overline{\tau}[p,q]=(t,t)\\ \overline{\tau}\to_{p,q}\overline{\tau}'}} \frac{1}{|\mathcal{T}|}. \tag{2.5.22}$$

This is equal to the probability that a uniformly random time configuration τ is such that $\overline{\tau}[p,q] = (t,t)$. By Lemma 2.5.8, this is at least $1/D^2 = 1/\mathrm{polylog}(N)$.

Case 2.2. The second subcase is that there is at least one flag qubit F_p such that P_X is identity on it. We follow a similar line of reasoning as in Case 1.

Let (p,q) be a pair of data qubits such that P_X acts as the identity on F_p but has a σ_X acting on F_q. Let $t^* = \lceil X/2\rceil$. Let $\tau = (t_1,\ldots,t_n)$ be any time configuration where $t_p = t_q = t^*$.

Let $|\tau'\rangle = P_X|\tau\rangle$. Then it must be that $\tau' \notin \mathcal{T}$. In other words, it is not a valid time configuration. This is because if $\tau' = (t'_1,\ldots,t'_n)$, then $t'_p = t_p = t^*$, yet $t'_q = 2X+1-t_q > X+1$. In particular, $f_q(\tau') = 1$ and $f_p(\tau') = 0$, which violates the causality constraints on the set of time configurations. In other words, the "membrane" described by τ' is broken between qubits p and q. Let Π be the component of $H_{causal}[p,q]$ verifying $t_p = t^*$ from the spacetime Hamiltonian (see (2.2.14)). Then we have that $\langle\tau'|\Pi|\tau'\rangle = 1$.

Let $\rho = \mathrm{Tr}_{\overline{\mathsf{T}}}(P\psi P^\dagger)$ denote the reduced density matrix of $P|\psi\rangle$ on the time configuration register T. Notice that $\rho = P^{time}\,\mathrm{Tr}_{\overline{\mathsf{T}}}(\psi)(P^{time})^\dagger$ where P^{time} is the restriction of P to the time configuration register. Since

$$\mathrm{Tr}_{\overline{\mathsf{T}}}(\psi) = \frac{1}{|\mathcal{T}|}\sum_\tau |\tau\rangle\langle\tau| \tag{2.5.23}$$

is a convex combination of classical states, and the P_Z operator leaves classical states invariant, we have that $\rho = P_X\,\mathrm{Tr}_{\overline{\mathsf{T}}}(\psi)P_X^\dagger$.

From a similar argument to that in Case 1, we obtain that the probability of sampling a time configuration $\boldsymbol{\tau} = (t_1, \ldots, t_n)$ such that $t_p = t_q = t^*$ is $1/D^2$. Thus

$$\mathrm{Tr}(\Pi P \psi P^\dagger) = \mathrm{Tr}(\Pi \rho) \geqslant \frac{1}{D^2}. \tag{2.5.24}$$

Case 3. Now suppose that $P_{flag} \otimes P_{clock}$ only has σ_Z or identity factors. This means that for all $\boldsymbol{\tau} \in \mathcal{T}$, we have $P_{flag} \otimes P_{clock}|\boldsymbol{\tau}\rangle = b_{\boldsymbol{\tau}}|\boldsymbol{\tau}\rangle$, where $b_{\boldsymbol{\tau}} \in \{\pm 1\}$. Thus we can write $P|\psi\rangle$ as

$$P|\psi\rangle = \frac{1}{\sqrt{|\mathcal{T}|}} \sum_{\boldsymbol{\tau} \in \mathcal{T}} |\boldsymbol{\tau}\rangle \otimes b_{\boldsymbol{\tau}} P_{data}|\psi_{\boldsymbol{\tau}}\rangle. \tag{2.5.25}$$

Case 3.1. First, suppose that $P_{data} \neq I$. Let $p \in \{1, \ldots, n\}$ be a data qubit such that P_{data} is some non-identity Pauli matrix $\sigma \in \{\sigma_X, \sigma_Y, \sigma_Z\}$ on S_p. If $\sigma = \sigma_Y$, let t denote a layer and $q \neq p$ denote a qubit such that $U_t[p, q]$ in the Brown-Fawzi circuit is $S \otimes I$, with S acting on p; otherwise, let t and q be such that $U_t[p, q] = H \otimes I$. Such t, q exist by Proposition 2.5.5. Without loss of generality suppose that $\sigma = \sigma_X$.

Consider the projector $\Pi = H_t[p, q]$ in the spacetime circuit Hamiltonian, which is one of the projectors in the set $\mathcal{D}_0$. Let $\boldsymbol{\tau} = (t_1, \ldots, t_n)$ be a time configuration such that $t_p = t_q = t$. Let $\boldsymbol{\tau}' = (t'_1, \ldots, t'_n)$ be the same as $\boldsymbol{\tau}$ except $t'_p = t'_q = t + 1$ (i.e. it is the configuration after gate $U_t[p, q]$ is applied). Thus $|\psi_{\boldsymbol{\tau}'}\rangle = H|\psi_{\boldsymbol{\tau}}\rangle$. Then notice that

$$\left(b_{\boldsymbol{\tau}}\langle\boldsymbol{\tau}| \otimes \langle\psi_{\boldsymbol{\tau}}| + b_{\boldsymbol{\tau}'}\langle\boldsymbol{\tau}'| \otimes \langle\psi_{\boldsymbol{\tau}'}|\right) P_{data}^\dagger \Pi P_{data} \left(b_{\boldsymbol{\tau}}|\boldsymbol{\tau}\rangle \otimes |\psi_{\boldsymbol{\tau}}\rangle + b_{\boldsymbol{\tau}'}|\boldsymbol{\tau}'\rangle \otimes |\psi_{\boldsymbol{\tau}'}\rangle\right) \tag{2.5.26}$$

$$= \left(b_{\boldsymbol{\tau}}\langle\boldsymbol{\tau}| \otimes \langle\psi_{\boldsymbol{\tau}}| + b_{\boldsymbol{\tau}'}\langle\boldsymbol{\tau}'| \otimes \langle\psi_{\boldsymbol{\tau}}|H\right) (I \otimes \sigma^\dagger)\Pi(I \otimes \sigma) \left(b_{\boldsymbol{\tau}}|\boldsymbol{\tau}\rangle \otimes |\psi_{\boldsymbol{\tau}}\rangle + b_{\boldsymbol{\tau}'}|\boldsymbol{\tau}'\rangle \otimes H|\psi_{\boldsymbol{\tau}}\rangle\right) \tag{2.5.27}$$

This expectation vanishes if and only if

$$\langle\psi_{\boldsymbol{\tau}}|H\sigma^\dagger H\sigma|\psi_{\boldsymbol{\tau}}\rangle = b_{\boldsymbol{\tau}} b_{\boldsymbol{\tau}'}. \tag{2.5.28}$$

However, Proposition 2.5.6 implies that $\langle\psi_{\boldsymbol{\tau}}|H\sigma^\dagger H\sigma|\psi_{\boldsymbol{\tau}}\rangle$ is either 0 or purely imaginary. This implies that (2.5.26) does not vanish, and furthermore, it is exactly equal to 1.

Thus we can evaluate the expectation of $\Pi = H_t[p, q]$ with respect to $P|\psi\rangle$. The expectation $\langle\psi|P^\dagger\Pi P|\psi\rangle$ is equal to

$$\frac{1}{|\mathcal{T}|} \sum_{\substack{\boldsymbol{\tau}, \boldsymbol{\tau}' \\ \boldsymbol{\tau}[p,q] = (t,t) \\ \boldsymbol{\tau} \to_{p,q} \boldsymbol{\tau}'}} \left(b_{\boldsymbol{\tau}}\langle\boldsymbol{\tau}| \otimes \langle\psi_{\boldsymbol{\tau}}| + b_{\boldsymbol{\tau}'}\langle\boldsymbol{\tau}'| \otimes \langle\psi_{\boldsymbol{\tau}'}|\right) P_{data}^\dagger \Pi P_{data} \left(b_{\boldsymbol{\tau}}|\boldsymbol{\tau}\rangle \otimes |\psi_{\boldsymbol{\tau}}\rangle + b_{\boldsymbol{\tau}'}|\boldsymbol{\tau}'\rangle \otimes |\psi_{\boldsymbol{\tau}'}\rangle\right)$$

$$= \sum_{\boldsymbol{\tau} : \boldsymbol{\tau}[p,q] = (t,t)} \frac{1}{|\mathcal{T}|}. \tag{2.5.29}$$

This is equal to the probability that a uniformly random time configuration $\boldsymbol{\tau}$ is such that $\boldsymbol{\tau}[p,q] = (t,t)$. By Lemma 2.5.8, this is at least $1/D^2 = 1/\text{polylog}(N)$. The cases $\sigma = \sigma_Z$ and $\sigma = \sigma_Y$ can be treated as described above by replacing H with S.

Case 3.2. Next, we handle the case of $P_{data} = I$. Since $P \notin \mathcal{S}$, we have that $P_{flag} \neq S_{flag}$.

Case 3.2.1. First suppose that $P_{clock} \neq I$.

Case 3.2.1.1. Suppose there exists a pair of data qubits (p,q) and an index $1 \leqslant j \leqslant X$ such that

1. p and q are neighboring qubits in the circuit C at time at a time t such that $j = t$ or $j = 2X + 1 - t$.
2. P_{clock} has a σ_Z factor acting on $\mathsf{C}_{p,j}$ but has an identity factor acting on $\mathsf{C}_{q,j}$.

With probability at least $1/D^2$ over a uniformly random time configuration $\boldsymbol{\tau}$ such that $\boldsymbol{\tau}[p,q] = (j,j)$, we have that $b_{\boldsymbol{\tau}} b_{\boldsymbol{\tau}'} = -1$, where $\boldsymbol{\tau} \to_{p,q} \boldsymbol{\tau}'$. Consider the projector $\Pi = H_j[p,q]$ in $\mathcal{D}_0$. In order for $\langle\psi|P^\dagger \Pi P|\psi\rangle$ to vanish, we would need that $\langle\psi_{\boldsymbol{\tau}'}|\psi_{\boldsymbol{\tau}}\rangle = -1$ for all such $\boldsymbol{\tau}$ and $\boldsymbol{\tau}'$, which cannot hold. Therefore $\langle\psi|P^\dagger \Pi P|\psi\rangle \geqslant 1/\text{polylog}(N)$.

Case 3.2.1.2. Now assume that for all pairs of data qubits (p,q) and an index $1 \leqslant j \leqslant X$ such that

1. p and q are neighboring qubits in the circuit C at time t for $j = t$ or $j = 2X + 1 - t$, then
2. P_{clock} has a σ_Z factor acting on $\mathsf{C}_{p,j}$ if and only if P_{clock} has a σ_Z factor acting on $\mathsf{C}_{q,j}$.

We argue that if there is a σ_Z factor on $\mathsf{C}_{p,j}$ then there is a σ_Z factor on $\mathsf{C}_{q,j}$ for all q in $rect(p,j)$ (defined earlier). This is because this is precisely the equivalence class of qubits that are neighbors in times involving qubits in the jth layer.

Therefore, P_{clock} is a product of $S_{clock}[rect(p,j)]$ for some subset of rectangles meaning $P_{clock} \in \mathcal{S}$ and we can equivalently consider $P \cdot P_{clock}$ as the logical Pauli error.

Case 3.2.2. Finally, suppose that $P_{clock} = I$ but $P_{flag} \neq I$. Since $P_{flag} \neq S_{flag}$, there exists two data qubits (p,q) such that P_{flag} has a σ_Z factor acting on F_p but has an identity factor acting on F_q. With probability at least $1/D^2$ over a uniformly random time configuration $\boldsymbol{\tau}$ such that $\boldsymbol{\tau}[p,q] = (X,X)$, we have that $b_{\boldsymbol{\tau}} b_{\boldsymbol{\tau}'} = -1$, where $\boldsymbol{\tau} \to_{p,q} \boldsymbol{\tau}'$. This is because it is the transition from time $t = X$ to $t = X + 1$ that the flag qubit on qubits p and

q switch from $|0\rangle$ to $|1\rangle$. Consider the projector $\Pi = H_X[p,q]$ in $\mathcal{D}_0$. In order for $\langle\psi|P^\dagger\Pi P|\psi\rangle$ to vanish, we would need that $\langle\psi_{\boldsymbol{\tau}'}|\psi_{\boldsymbol{\tau}}\rangle = -1$ for all such $\boldsymbol{\tau}$ and $\boldsymbol{\tau}'$, which cannot hold. Therefore $\langle\psi|P^\dagger\Pi P|\psi\rangle \geqslant 1/\mathrm{polylog}(N)$.

Boosting the success probability. We now boost our detection set $\mathcal{D}_0$ to a stronger set of projectors $\mathcal{D}$ that can detect errors with very high probability. We leverage the following result of Marriott and Watrous [70]:

Lemma 2.5.9 (In-place amplification [70])**.** *Let $\delta > 0$. Let A be a circuit on r qubits along with s ancilla bits. Then there exists a circuit A' on r qubits and $s' = s + O(\delta^{-3})$ ancillas, that has size at most $O(\delta^{-3})$ times the size of A, such that the following holds: for all r-qubit states $|\varphi\rangle$, if A accepts $|\varphi\rangle|0^s\rangle$ with probability 0, then A' accepts $|\varphi\rangle|0^{s'}\rangle$ with probability 0. Otherwise, if A accepts $|\varphi\rangle|0^s\rangle$ with probability at least δ, then A' accepts $|\varphi\rangle|0^{s'}\rangle$ with probability at least $1 - 2^{-1/\delta}$.*

Here, we define the acceptance probability of the circuit to be the probability that the first qubit measures $|1\rangle$.

For each projector $\Pi \in \mathcal{D}_0$, we create a "boosted" projector $\Pi' \in \mathcal{D}$ in the following way: let A denote a circuit that performs the projective measurement $\{\Pi, I - \Pi\}$ and records the outcome in an ancilla qubit. The circuit A acts on $9+1$ qubits, and has size $O(1)$. Let A' be the amplified circuit given by Lemma 2.5.9 for $\delta = 1/\mathrm{polylog}(N)$. The circuit A' acts on 9 qubits and $s' = O(\delta^{-3})$ ancillas. Define the projector $\Pi' = A'(|1\rangle\langle 1| \otimes I)(A')^\dagger$ where $|1\rangle\langle 1|$ denotes the projection onto the first qubit being in the state $|1\rangle$.

Then we have that, for all spacetime codewords $|\psi\rangle$, for all Pauli errors $P \in \mathcal{P}_N$,

1. If $\Pi P|\psi\rangle = 0$, then $\Pi'(P|\psi\rangle|0^{s'}\rangle) = 0$, and
2. If $\mathrm{Tr}(\Pi P\psi P^\dagger) \geqslant 1/\mathrm{polylog}(N)$, then $\mathrm{Tr}(\Pi'(P\psi P^\dagger) \otimes |0^{s'}\rangle\langle 0^{s'}|) \geqslant 1 - 2^{-\mathrm{polylog}(N)}$.

Thus we have established that for all non-identity $P \in \mathcal{P}_N$, there exists a projector $\Pi \in \mathcal{D}$ such that $\mathrm{Tr}(\Pi P\psi P^\dagger) \geqslant 1 - 2^{-\mathrm{polylog}(N)}$. For general Pauli channels $\mathcal{E}$, we have that

$$\mathrm{Tr}\left(\Pi\,\mathcal{E}(\psi)\right) = \sum_P c_P \left(\Pi\, P\psi P^\dagger\right) \geqslant \sum_{P\neq I} c_P(1-2^{-\mathrm{polylog}(N)}) = (1-c_I)(1-2^{-\mathrm{polylog}(N)}). \tag{2.5.30}$$

This concludes the first part of the Theorem.

We now establish the "Furthermore" part of the Theorem. Since the spacetime Hamiltonian is spatially local in $\mathrm{polylog}(N)$ dimensions, and each qubit participates

in at most polylog(N) terms, this implies that the projectors in $\mathcal{D}$ can be divided into $K = \text{polylog}(N)$ layers $B_1, \ldots, B_K$ such that the projectors in any set B_j act on disjoint sets of qubits.

The measurement M will consist of measuring the layers $B_1, B_2, \ldots, B_K$ in sequence, and accepting if any of the projectors in the layers accept. Since each projector can be implemented using a size polylog(N) circuit, each layer measurement can be implemented using a depth polylog(N) circuit, so, therefore, M can be implemented using a depth polylog(N) circuit.

Let P denote a non-identity Pauli operator in $\mathcal{P}_N$, and let $|\psi\rangle$ be a spacetime codeword. Let B_j be the first layer that contains a projector $\Pi \in \mathcal{D}$ projector that accepts $P|\psi\rangle$ with positive probability. Then the probability that measuring M rejects on the state $P|\psi\rangle$ is at most the probability that measuring Π rejects $P|\psi\rangle$, which is $2^{-\text{polylog}(N)}$. We do not have to worry about the projectors in earlier layers, because by definition they reject $P|\psi\rangle$ with certainty, and leave the state unchanged.

We can extend this argument to a general Pauli channel $\mathcal{E}$ in the same way as before, and this completes the proof of the Theorem. □

2.6 Alternate constructions and spatial locality

Good approximate QLDPC from weighted FK Hamiltonians

In this section, we describe another closely related construction of approximate LDPC codes, which is based on using the standard Feynman-Kitaev construction with a global clock as well as recently-introduced variants that increase the overlap of the history state with the beginning and end of the computation [14, 25]. The primary advantage of this version of the construction is the significantly simpler analysis of the spectral gap, even in the presence of nonuniform weight distributions on the time steps of the computation. The main disadvantage for this version of the construction is that the increase in energy caused by local errors is significantly reduced in some cases (thereby making them more difficult to detect). In the global clock construction, there are local errors with expected energy scaling like $1/T$ where T is the (polynomial) size of the computation, instead of errors having energy $1/D$ in the spacetime construction where D is the (polylogarithmic) depth. This can be seen as a fulfillment of the intuition that the spacetime circuit-to-Hamiltonian is more robust than its global clock counterpart. Finally, due to the simplification in the analysis for the global clock construction, we can achieve a provably optimal tradeoff between the approximate error ε of the code and the soundness s, using a result that was previously established in [14].

Result. For any $\varepsilon(N) > 0$ there exists an $[[N, k, d, \varepsilon, \ell, s]]$ approximate LDPC code with $k = \Omega(N/\log^5 N)$, $d = \Omega(N/\log^5 N)$, $\ell = 5$, $s = \text{polylog}(N)$. The

spectral gap of the code Hamiltonian is

$$\Delta_H = \Omega\left(\frac{\varepsilon(N)}{N^3\,\mathrm{polylog}(N)}\right). \tag{2.6.1}$$

The encoding circuits analyzed by Brown and Fawzi with depth $D = O(\log^3 n)$ have size $T = O(n\log^2 n)$. To ensure that only a polylogarithmic number of Hamiltonian terms act on each physical qubit we consider the same sequence of random Clifford gates interspersed with bitonic sorting circuits of Section 2.2,

$$\prod_{t=1}^{D}\bigotimes_{(p,q)\in L_t} U_t[p,q] \tag{2.6.2}$$

but now the local gates are each applied individually in sequence. Note that in this section we do not reverse the application of the gates, and the time register is not periodic. For each layer L_t we choose an ordering $(p_1,q_1)...(p_{n/2},q_{n/2})$ for the pairs of qubits interacting within that layer, and we re-index this sequence of $T = nD/2$ gates as simply $U_T...U_1$,

$$U_T...U_1 = U_t[q_{n/2},p_{n/2}]...U_2[p_1,q_1]U_1[p_{n/2},q_{n/2}]...U_1[p_1,q_1] \tag{2.6.3}$$

The code space, which will be the ground space of the code Hamiltonian, is

$$C = \left\{\sum_{t=0}^{T+n}\sqrt{\pi_t}|t\rangle_C\otimes U_{t-n}...U_1|\psi,0,...,0\rangle_S : |\psi\rangle\in\mathbb{C}^{n-k}\right\} \tag{2.6.4}$$

where by convention we define $U_{t-n}...U_1 = \mathbb{1}$ for $t < n$, and the distribution π is defined by

$$\pi_t = \begin{cases}\frac{\varepsilon}{T+n} & , \quad 0\leqslant t < T+n \\ 1-\varepsilon & , \quad t = T+n\end{cases}. \tag{2.6.5}$$

Instead of the standard H_{in} of the form,

$$|0\rangle\langle 0|_C\otimes\mathbb{1}_{S_{1...n-k}}\otimes\left(\sum_{r=n-k}^{n}|1\rangle\langle 1|_{S_r}\right). \tag{2.6.6}$$

we use a "staggered" version of the input check,

$$H_{\mathrm{in}} = \sum_{r=n-k}^{n}|r-n+k\rangle\langle r-n+k|_C\otimes I_{S_{1...n-k}}\otimes|1\rangle\langle 1|_{S_R}. \tag{2.6.7}$$

The point of the staggered input check is to avoid having a nonconstant number of terms acting on the clock bits that represent $t = 0$.

The propagation Hamiltonian for this nonuniform distribution over time steps is based on the method used in [14]. One first considers the Markov chain with Metropolis

transition probabilities (see [65] for a general background on Markov chains) from t to $t-1, t+1$ that is reversible with respect to π. For $0 < t < T-n$ we have

$$P_{t,t+1} = \frac{1}{4}\min\left\{1, \frac{\pi_{t+1}}{\pi_t}\right\} \ , \ P_{t,t-1} = \frac{1}{4}\min\left\{1, \frac{\pi_{t-1}}{\pi_t}\right\} \ , \ P_{t,t} = 1 - P_{t,t+1} - P_{t,t-1} \tag{2.6.8}$$

and also

$$P_{0,t} = \frac{1}{2}\min\left\{1, \frac{\pi_1}{\pi_0}\right\} \quad , \quad P_{0,0} = 1 - P_{0,1} \tag{2.6.9}$$

and

$$P_{T+n,T+n-1} = \frac{1}{2}\min\left\{1, \frac{\pi_{T+n-1}}{\pi_{T+n}}\right\} \quad , \quad P_{T+n,T+n} = 1 - P_{T+n,T+n-1}. \tag{2.6.10}$$

The transition probabilities satisfy $\pi_t P_{t,t'} = \pi_{t'} P_{t',t}$ for all $0 \leqslant t, t' \leqslant T+n$, and so $\sum_{t=0}^{T+n} \pi_t P_{t,t'} = \pi'_t$. Therefore the propagation Hamiltonian defined by $H_{\text{prop}} = \sum_{t=0}^{T+n} H_{\text{prop}}(t)$ with

$$H_{\text{prop}}(t) = \frac{1}{2}\big(P_{t,t}|t\rangle\langle t|_C \otimes \mathbb{1}_S + P_{t-1,t-1}|t\rangle\langle t|_C \otimes \mathbb{1}_S \tag{2.6.11}$$

$$- \pi_t^{1/2}\pi_{t-1}^{-1/2} P_{t,t-1}|t\rangle\langle t-1|_C \otimes U_{t-n} - \pi_{t-1}^{1/2}\pi_t^{-1/2} P_{t-1,t}|t-1\rangle\langle t|_C \otimes U_{t-n}^\dagger\big), \tag{2.6.12}$$

is such that $H_{\text{in}} + H_{\text{prop}}$ has the ground space C in (2.6.4) as claimed.

The locality of the checks $\ell = 5$ follows from the fact that when the clock register is implemented with qubits as in (2.2.9) the local terms in H are at most 5-local, and this is unaffected by the modified coefficients in the propagation terms. The error bound of ε for the code follows from the same argument used in Section 2.3 together with the fact that the distribution π assigns a probability of $1-\varepsilon$ to the final time step of the computation.

To obtain the bound $s = \text{polylog}(N)$ on the number of check terms acting on each physical qubits, we first consider the clock bits. For $n+2 \leqslant t \leqslant T+n-2$ there are 5 terms acting on clock bit t,

$$H_{\text{prop}}(t-2), H_{\text{prop}}(t-1), H_{\text{prop}}(t), H_{\text{prop}}(t+1), H_{\text{prop}}(t+2) \tag{2.6.13}$$

and for $t = n-1, T+n-1, T+n$ the number of propagation terms is even fewer. For $0 \leqslant t \leqslant n$, each clock bit participates in one term from H_{in} and at most 5 terms from H_{prop}. Finally, the number of nontrivial gates acting on each system qubit is at most $D = \text{polylog}(N)$.

The spectral gap of H_{prop} is the same as the spectral gap of the Markov chain P described above, which can be lower bounded by Cheeger's inequality. Since $1 - o(1)$

of the weight in the stationary distribution is concentrated on the final time step $t = T + n$, the subset $S = \{0, ..., T + n - 1\}$ has the minimum conductance which is

$$\Phi = \frac{1}{\pi(S)} \sum_{t \in S, t \notin S} \pi_t P_{t,t'} = \frac{1}{\varepsilon} \left(\frac{\varepsilon}{T+n} \right) P_{T+n-1,T+n} = \frac{1}{4(T+n)} \tag{2.6.14}$$

and since $T = n\,\mathrm{polylog}(n)$ we have

$$\Delta_{H_{\mathrm{prop}}} = \Omega\left(\frac{1}{n^2 \mathrm{polylog}(n)} \right). \tag{2.6.15}$$

To go from $\Delta_{H_{\mathrm{prop}}}$ to Δ_H we apply the same argument as in Section 2.4, and use the geometrical lemma to obtain

$$\Delta_H = \Omega\left(\frac{\varepsilon}{n^3 \mathrm{polylog}(n)} \right). \tag{2.6.16}$$

Finally, we note that because of the dual importance of the spectral gap and the overlap with the endpoint of the computation, which respectively determine the soundness of the code and the infidelity of recovery, the optimality of the distribution π in (2.6.5) follows from Theorem 8 in [14].

Theorem 2.6.1 ([14]). *Let $|\psi\rangle$ be the ground state of a Hamiltonian H with eigenvalues $E \stackrel{\mathrm{def}}{=} E_0 \leqslant E_1 \leqslant \ldots \leqslant E_T$. If H is tridiagonal in the basis $\{|0\rangle, \ldots, |T\rangle\}$,*

$$H \stackrel{\mathrm{def}}{=} \sum_{t=0}^{T} a_t |t\rangle\langle t| + \sum_{t=0}^{T-1} \left(b_t |t+1\rangle\langle t| + b_t^* |t\rangle\langle t+1| \right), \tag{2.6.17}$$

with $|a_t|, |b_t| \leqslant 1$ for $t = 0, \ldots, T$ then the product $\Delta_H \cdot \min\{|\psi|_0^2, |\psi_T|^2\}$ of the spectral gap $\Delta_H = E_1 - E$ and the minimum endpoint overlap is $O(T^{-2})$.

Spatial locality of the Hamiltonian

In this section, we demonstrate that the code Hamiltonian is indeed $\mathrm{polylog}(n)$-spatially local. We also provide a sketch of how to make the construction $O(\log n)$-spatially local at the cost of increasing the locality of the Hamiltonian from 9 to 15.

We give a technical definition for spatial locality that fits the previous descriptions given in other works [20, 44].

Definition 2.6.2. A code defined by a local Hamiltonian $H = \sum_i H_i$ is d-spatially local if there exists an embedding map $\mathrm{emb} : Q \to \mathbb{R}^d$, where Q is the set of qubits, satisfying the following conditions:

1. For all $q_1 \neq q_2 \in Q$, $\|\mathrm{emb}(q_1) - \mathrm{emb}(q_2)\|_2 \geqslant 1$.

2. Let $Q_i \subseteq Q$ be the set of qubits acted on non-trivially by Hamiltonian H_i. There exists a constant $c > 0$ such that for all $q_1, q_2 \in Q_i$,

$$\|\mathrm{emb}(q_1) - \mathrm{emb}(q_2)\|_2 \leqslant c. \tag{2.6.18}$$

We propose such an embedding for $d = O(\log^5 n)$. First, consider the interaction graph of the qubits in a bitonic block architecture $\mathcal{B}_\ell$ for $\ell = \log n$ (there exists an edge between two qubits if they share a gate). We note the following lemma:

Lemma 2.6.3. *The incidence graph of the bitonic block architecture $\mathcal{B}_\ell$ is equivalent to the ℓ-dimensional hypercube.*

Proof. The result is easy to see for $\ell = 1$. For $\ell > 1$, notice that layer $\mathcal{L}_1$ connects matching vertices in two bitonic blocks $\mathcal{B}_{\ell-1}$ (Corollary 2.7.9), which by induction yields a ℓ-dimensional hypercube. □

Therefore, there is an encoding $h : [n] \to \mathbb{R}^\ell$ such that if qubits i and j interact then $\|h(1) - h(2)\|_2 = 1$. Let e_j denote the j-th standard basis vector.

Theorem 2.6.4. *The code defined in this paper is $O(\log^5 n)$-spatially local.*

Proof. We provide the explicit embedding map and prove it satisfies the definition. Our embedding map can be seen as

$$\mathrm{emb} : Q \to \mathbb{R}^\ell \times \mathbb{R}^1 \times \mathbb{R}^X \tag{2.6.19}$$

defined by

$$\textbf{Data registers} \qquad \mathrm{emb}(\mathsf{S}_i) = (h(i), 0, 0^X), \tag{2.6.20}$$

$$\textbf{Flag registers} \qquad \mathrm{emb}(\mathsf{F}_i) = (h(i), 1, 0^X), \tag{2.6.21}$$

$$\textbf{Clock registers} \qquad \mathrm{emb}(\mathsf{C}_{i,j}) = (h(i), 0, e_j). \tag{2.6.22}$$

Note that $\ell + 1 + X = \log n + 1 + O(\log^5 n) = O(\log^5 n)$.

Every coordinate of every qubit is either 0 or 1 and clearly the qubits are distinct. Therefore, the minimal distance between them is indeed 1. We now verify that each of the Hamiltonian terms act on qubits that are only a constant distance of $\sqrt{3}$ apart.

H_{clock} **terms.** All H_{clock} terms are projections of the form $\Pi^{(01)}_{\mathsf{C}_{i,j},\mathsf{C}_{i,j+1}}$. Any pair of clock qubits $\mathsf{C}_{i,j}$ and $\mathsf{C}_{i,j'}$ are distance $\sqrt{2}$ in the ℓ_2 norm.

H_{init} **terms.** All H_{init} terms are projections of the form $\Pi^{(1)}_{\mathsf{C}_{i,0}} \otimes \Pi^{(1)}_{\mathsf{S}_i}$. We note that for any i, any clock qubit $\mathsf{C}_{i,j}$ is distance $\sqrt{2}$ in the ℓ_2 norm from S_i.

H_{prop} **terms.** All H_{prop} terms are interactions between the clock, flag, and data qubits for qubits p and q that share a gate $U_t[p,q]$ in the circuit. We note that $h(p)$ and $h(q)$ have Hamming distance 1 (i.e. differ in only one coordinate). The specific set of qubits involved are $\mathsf{S}_p, \mathsf{S}_q, \mathsf{F}_p, \mathsf{F}_q$, as well as six clock qubits, three $\mathsf{C}_{p,\cdot}$ and three $\mathsf{C}_{q,\cdot}$ (the exact collection depends on t according to (2.2.13)). It is not difficult to see that any two of the embeddings of these qubits differ in at most 3 coordinates, hence a distance of $\sqrt{3}$ in the ℓ_2 norm.

H_{causal} **terms.** A term in H_{causal} compares the clock of a qubit p to an adjacent qubit q. It will involve the flag qubit F_p and up to two clock qubits $\mathsf{C}_{p,a}, \mathsf{C}_{p,a+1}$ (again, a depends on t; see (2.2.13)). In addition, it checks the flag qubit F_q and up to two clock qubits $\mathsf{C}_{q,b}, \mathsf{C}_{q,c}$ (here b and c depend on t; see the case-wise definition of H_{causal}). Likewise, it is not difficult to see that any two of these embeddings of these qubits differ in at most 3 coordinates, hence a distance of $\sqrt{3}$ in the ℓ_2 norm.

□

Alternate construction

We now present an alternate construction which improves the spatial locality of the Hamiltonian. We only provide a sketch of the construction as the majority of the analysis is similar to that presented in the main sections of the paper. In particular, we will demonstrate that a different representation of the time register can be used to make this code $O(\log n)$-spatially local at the cost of worsening the code to being 50-local.

Instead of encoding the time register using a flag and a domain wall, we will encode it using the multi-dimensional clock method used in [73]. At a high level, we express time in its unique representation in base $\bar{D} = \lceil \sqrt[6]{D} \rceil + 1$ and represent each coordinate of the representation using a flag and domain wall.

More specifically, additionally let $\bar{X} = \frac{\bar{D}-2}{2}$. Then $\bar{D} = O(\log n)$ for our construction. For any number $0 \leqslant j \leqslant D$, let $a_0, \ldots, a_5$ be the unique numbers $\in \{0, \ldots, \bar{D}-1\}$ such that

$$j = a_0 + a_1\bar{D} + \ldots + a_5\bar{D}^5. \tag{2.6.23}$$

We then express $|j\rangle_{\mathsf{T}_i}$ as

$$|j\rangle_{\mathsf{T}_i} = |a_0\rangle_{\mathsf{T}_i^{(0)}} \otimes |a_5\rangle_{\mathsf{T}_i^{(5)}} \tag{2.6.24}$$

where $|a_k\rangle_{\mathsf{T}_i^{(k)}}$ is an encoding with a flag and domain wall of times between $\{0, \ldots, \overline{D}-1\}$ as described in the main section of the paper. $\mathsf{T}_i^{(k)}$ consists of a flag register $F_i^{(k)}$ and $\{C_{i,j}^{(k)}\}_{j=0}^{\bar{X}}$.

This adjustment will require the Hamiltonian terms to act on 6 times as many time registers as before; hence the 50-locality. The encoding to demonstrate $O(\log n)$-spatial locality is

$$\textbf{Data registers} \qquad \mathrm{emb}\,(\mathsf{S}_i) = (h(i), 0^6, 0^{6\bar{X}}), \tag{2.6.25}$$

$$\textbf{Flag registers} \qquad \mathrm{emb}\left(\mathsf{F}_i^{(k)}\right) = (h(i), e_k, 0^{6\bar{X}}), \tag{2.6.26}$$

$$\textbf{Clock registers} \qquad \mathrm{emb}\left(\mathsf{C}_{i,j}\right) = (h(i), e_k, \varepsilon_k \otimes e_j). \tag{2.6.27}$$

Acknowledgments

We thank Winton Brown, Aram Harrow, and Umesh Vazirani for helpful discussions. Author TB acknowleges support from NSERC through a PGSD award. Author CN is supported by ARO Grant W911NF-12-1-0541 and NSF Grant CCF-1410022. Part of this work was completed while authors EC, CN, and HY were visitors at the Simon's Institute 2018 summer cluster *Challenges in Quantum Computation*.

2.7 Partially applied configurations of a bitonic sorting circuit

In this Appendix, we provide the mathematical foundations required for analyzing the spectral gap of the Hamiltonian and generating the encoding circuit. We explore the properties of the bitonic block [13] (see Definition 2.2.7) and prove results about the space of *valid configurations* of partial computations of a bitonic block (see Definition 2.3.4).

Configurations and width

We study the structure and combinatorics of the valid configurations (see Definition 2.3.4). One can think of a valid partial configuration as being represented visually on the architecture as a string which partitions the architecture into two halves: gates that have and have not been applied. Depending on the configuration of the circuit in question, there is a maximum *width*, a number of layers, that such a string can have.

Definition 2.7.1. The *width* of a partial configuration τ is

$$w(\tau) = \max(\tau) - \min(\tau). \tag{2.7.1}$$

Lemma 2.7.2. *For bitonic block $\mathcal{B}_\ell$,*

$$w(\tau) < \ell \tag{2.7.2}$$

for any valid partial configuration τ.

Proof. Suppose that we have a configuration of width ℓ, then at least one gate in the final layer $\mathcal{L}_\ell$ must be applied (and corresponding qubits are at time $t = \ell$),

and at least one gate in the first layer $\mathcal{L}_1$ must not have been applied (so that its corresponding qubits are at time $t = 0$). Let g be a gate in $\mathcal{L}_\ell$ that has been applied.

Consider the light cone Λ of the gate g; there are two gates of $\mathcal{L}_{\ell-1}$ in Λ. Precisely, these are the two gates of the $\mathcal{B}_2$ block which connect the $\mathcal{B}_1$ block containing g to its neighboring $\mathcal{B}_1$ block (recall Definition 2.2.7). Likewise, there are 2^j gates of layer $\mathcal{L}_{\ell-j}$ in Λ as they are the 2^{j-1} gates of a $\mathcal{B}_j$ block connecting the block $\mathcal{B}_{j-1}$ to its neighboring $\mathcal{B}_{j-1}$ block.

Carrying this until the first layer, there are $2^{\ell-1}$ gates of Λ in $\mathcal{L}_1$. Since all of Λ must be applied, every gate of $\mathcal{L}_1$ is applied. Therefore, the assumption of a configuration of width ℓ is false.

□

The proof of Lemma 2.7.2 illustrates an interesting and important property of bitonic blocks; the light cone of any gate in the architecture doubles in size each layer. Additionally,

Corollary 2.7.3. *Any valid configuration of a bitonic block $\mathcal{B}_\ell$ must satisfy at least one of the following:*

1. *Every gate in layer $\mathcal{L}_1$ is activated.*
2. *Every gate in layer $\mathcal{L}_\ell$ is* not *activated.*

Permutations and the splitting property

In this subsection, we demonstrate some important combinatorial properties of bitonic blocks.

Fact 2.7.4. *The number of valid configurations of any architecture is invariant under permutation of the qubit labels.*

Definition 2.7.5. Two architectures $\mathcal{A}_1$ and $\mathcal{A}_2$ both acting on n qubits are called *isomorphic* (denoted $\mathcal{A}_1 \simeq \mathcal{A}_2$) if one can be obtained from the other by some permutation of the qubit labels.

Therefore, isomorphic architectures have the same number of valid configurations.

Fact 2.7.6. *For a bitonic block $\mathcal{B}_\ell$, the first $\ell - 1$ layers, $\{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_{\ell-1}\}$, can be viewed as $\mathcal{B}_{\ell-1}^{\otimes 2}$ where the first smaller block acts on odd indexed qubits and the second on even indexed qubits.*

Proof. It is easy to see for $\ell = 2$. For $\ell > 2$, by induction, we know the odd indexed qubits in layers $\{\mathcal{L}_2, \ldots, \mathcal{L}_{\ell-1}\}$ form $\mathcal{B}_{\ell-2}^{\otimes 2}$. Recall that $\mathcal{L}_1$ contains gates between qubit i and $i + 2^{\ell-1}$ for $i \leqslant 2^{\ell-1}$. This produces a block $\mathcal{B}_{\ell-1}$ on the odd indexed qubits; a similar argument holds for even indexed qubits. □

Lemma 2.7.7. *A single layer left cyclic shift of the layers of a bitonic block $\mathcal{B}_\ell$ is isomorphic to the bitonic block $\mathcal{B}_\ell$. The isomorphism is described by the permutation π_ℓ:*

$$\pi_\ell(i) \stackrel{\text{def}}{=} \begin{cases} 2i - 1 & \text{if } i \leqslant 2^{\ell-1} \\ 2i - 2^\ell & \text{if } i > 2^{\ell-1}. \end{cases} \tag{2.7.3}$$

A single layer right cycle shift isomorphism is described by the permutation π_ℓ^{-1}:

$$\pi_\ell^{-1}(i) \stackrel{\text{def}}{=} \begin{cases} \frac{i+1}{2} & \text{if } i \text{ odd} \\ \frac{i+2^\ell}{2} & \text{if } i \text{ even}. \end{cases} \tag{2.7.4}$$

Figure 2.9 illustrates the above permutations for $\ell = 3$.

Proof. By Fact 2.7.6, the first $\ell - 1$ layers of a bitonic block $\mathcal{B}_\ell$ form $\mathcal{B}_{\ell-1}^{\otimes 2}$ where one is the collection of odd indexed rows and other the collection of even indexed rows. By the definition, the last $\ell - 1$ layers form $\mathcal{B}_{\ell-1}^{\otimes 2}$ where one is the collection of the first 2^{k-1} rows and the other the collection of $2^{\ell-1}$ rows. Therefore, any permutation for a single layer left cyclic shift must permute these $\mathcal{B}_{\ell-1}$ blocks onto each other. It then is easy to check that the permutation π will also send the first layer to the last layer. The single layer right cycle shift is just the inverse permutation, π^{-1}. □

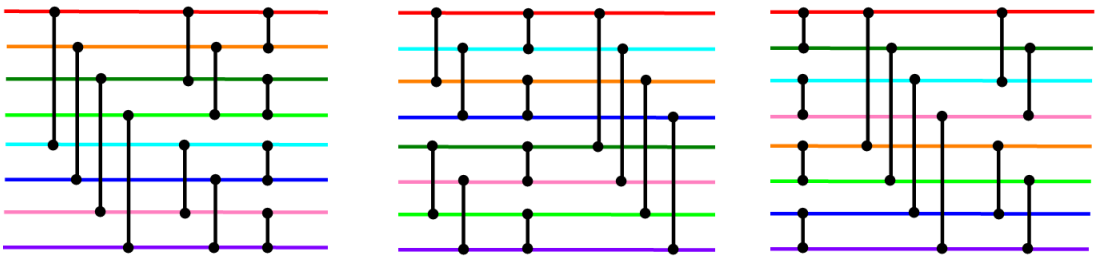


Figure 2.9: A demonstration of Lemma 2.7.7. The colored wires represent the permutation mapping each of the shifted bitonic blocks $\mathcal{B}_3$ back to the original bitonic block.

Corollary 2.7.8. *A j layer left (or right) cyclic shift of the layers of a bitonic block $\mathcal{B}_\ell$ is isomorphic to a bitonic block $\mathcal{B}_\ell$. The permutation describing the isomorphism is π_ℓ^j (or π_ℓ^{-j}).*

This yields the following important corollary.

Corollary 2.7.9. *Let $\mathcal{A}_\ell^j$ be a sub-architecture formed by taking any distinct j layers (in any order). Then,*

$$\mathcal{A}_\ell^j \simeq \bigotimes_{i=1}^{2^{k-j}} \mathcal{B}_j. \tag{2.7.5}$$

Proof. Consider any excluded layer. By the previous corollary, we can assume it to be the first layer. Thus, the remaining layers decompose into the tensor product of smaller bitonic blocks. We can repeat for each excluded layer. □

Counting configurations

Configurations of a bitonic block

We now recursively count the number of valid configurations of a block $\mathcal{B}_\ell$. This will be useful in the encoding circuit and the spectral gap analysis.

Theorem 2.7.10. *Let a_ℓ be the total number of valid partial configurations of $\mathcal{B}_\ell$. This number is described by the recurrence relation*[15]

$$a_\ell \stackrel{\text{def}}{=} 2a_{\ell-1}^2 - a_{\ell-2}^4, \tag{2.7.7}$$

with initial conditions $a_1 = 2$, $a_2 = 7$.

Proof. The initial cases can be counted by hand. For $\ell > 2$, by Corollary 2.7.3, we know that the first layer is entirely activated or the last layer is entirely not activated. Corollary 2.7.9, tells us that, in either case, the remaining layers are isomorphic to $\mathcal{B}_{\ell-1}^{\otimes 2}$. Therefore, aside from double-counting between the two cases, there are $2a_{\ell-1}^2$ valid configurations. The set of double counted configurations are all configurations that lie entirely in the middle $\ell - 2$ layers. Again we apply Corollary 2.7.9, to argue that this set of layers is isomorphic to $\mathcal{B}_{\ell-2}^{\otimes 4}$, and therefore, has $a_{\ell-2}^4$ valid configurations. □

Corollary 2.7.11. *The total number of valid partial configurations of $\mathcal{B}_\ell$ with some gate in layer $\mathcal{L}_1$ not activated is $a_\ell - a_{\ell-1}^2$.*

Proof. We need to ignore the valid partial configurations which have the entire $\mathcal{L}_1$ layer activated. The last $\ell - 1$ layers are isomorphic to $\mathcal{B}_{\ell-1}^{\otimes 2}$ by Corollary 2.7.9. □

[15] This recurrence relation does not have a known solution. It is, however, known to scale as

$$a_\ell \sim \frac{\omega^{2^\ell}}{\phi} \tag{2.7.6}$$

where ϕ is the golden ratio, and $\omega = 1.8445\ldots$, a number with no known form.

Configurations of products of bitonic blocks

Consider an architecture composed of m consecutive copies of a bitonic block of rank ℓ.

Definition 2.7.12 (Linear product of bitonic blocks)**.**

$$\mathcal{B}_\ell^{\times m} \stackrel{\text{def}}{=} \prod_{i=1}^{m} \mathcal{B}_\ell. \tag{2.7.8}$$

Theorem 2.7.13. *The total number of configurations of $\mathcal{B}_\ell^{\times m}$ is*

$$a_\ell^{\times m} \stackrel{\text{def}}{=} ((m-1)\,\ell + 1)\,a_\ell - (m-1)\ell a_{\ell-1}^2. \tag{2.7.9}$$

Proof. Notice, that *any* ℓ consecutive layers – henceforth called a window – of $\mathcal{B}_\ell^{\times m}$ is isomorphic to a bitonic block $\mathcal{B}_\ell$. Then by Lemma 2.7.2, we know that any valid configuration is contained within a window. We can, therefore, count the number of valid configurations by considering the first window it appears in. For every window except the last, all configurations corresponding to the window must have some gate in the first layer not activated; otherwise, they would correspond to a later window. By Corollary 2.7.11, there are $a_\ell - a_{\ell-1}^2$ configurations for every window except the last. For the last, there are no restrictions, so there are a_ℓ configurations. It is easy to see that there are $(m-1)\ell + 1$ windows. Then,

$$a_\ell^{\times m} = (m-1)\ell(a_\ell - a_{\ell-1}^2) + a_\ell = ((m-1)\,\ell + 1)\,a_\ell - (m-1)\ell a_{\ell-1}^2. \tag{2.7.10}$$

□

Definition 2.7.14. Let $\mathcal{B}_\ell^{\leftrightarrow m}$ be the circular architecture defined by taking m copies of the bitonic block and wrapping it around the cylinder.

Theorem 2.7.15. *The total number of configurations of $\mathcal{B}_\ell^{\leftrightarrow m}$ is*

$$a_\ell^{\leftrightarrow m} \stackrel{\text{def}}{=} (a_\ell - a_{\ell-1}^2)m\ell. \tag{2.7.11}$$

Proof. We can consider a similar argument as that of Theorem 2.7.13. In this case, there are ℓm windows as windows can wrap around the circular architecture. Since we identify each configuration with the first window containing it, every window must have some gate in the first layer not activated. Each layer of the architecture can be the start of a window since the architecture is circular. Therefore, there are $m\ell$ windows, completing the proof. □

We now provide the proof of Lemma 2.3.2 which was omitted from the main article.

Proof of Lemma 2.3.2. Let $\mathcal{T}_{comp}$ be the set of valid configurations for whom all clocks were past $D_1\ell^2$. Since all the gates in the circuit past time $D_1\ell^2$ are identity gates, $|\psi_\tau\rangle$ is constant. The subcircuit of identity gates has a depth of $(3/\varepsilon - 1)D_1\ell^2$ depth. By Lemma 2.7.2, we know that any valid configuration has a width of at most ℓ and by the counting argument of Lemma 2.7.17 and Theorem 2.7.15, we know that we can get a lower bound on $|\mathcal{T}_{comp}|/|\mathcal{T}|$ by counting the fraction of windows purely contained in the subcircuit of identity gates. To avoid any configurations that cross outside the region of identity gates, we will ignore the first and last $D_1\ell^2$ gates. Then the fraction of windows purely contained is at least

$$\frac{\left(\frac{3}{\varepsilon} - 1\right) D_1\ell^2 - 2D_1\ell^2}{\frac{3D_1\ell^2}{\varepsilon}} = 1 - \varepsilon. \tag{2.7.12}$$

□

Configurations overlapping the initial state

Lemma 2.7.16. *Let i be a fixed qubit. Then the number of valid configurations of $\mathcal{B}_\ell^{\times m}$ such that the clock of qubit i is at 0 is $\prod_{j=1}^{\ell-1} a_j$.*

Proof. We only need consider the first block $\mathcal{B}_\ell$ of $\mathcal{B}_\ell^{\times m}$ due to Lemma 2.7.2. By Corollary 2.7.3, we know that no gate in the last layer of $\mathcal{B}_\ell$ is activated. Therefore, we only need to consider the first $\ell - 1$ gates which are isomorphic to $\mathcal{B}_{\ell-1}^{\otimes 2}$ (Corollary 2.7.9). The block $\mathcal{B}_{\ell-1}$ corresponding to the set of qubits of which i is not a member has $a_{\ell-1}$ valid configurations. The set containing i can be recursively seen to have $\prod_{j=1}^{\ell-2} a_j$ valid configurations. □

Lemma 2.7.17. *Let i be a fixed qubit. Then the number of valid configurations of $\mathcal{B}_\ell^{\leftrightarrow m}$ such that the clock of qubit i is at 0 is $(a_\ell - a_{\ell-1}^2)$.*

Proof. By symmetry (Corollary 2.7.9), this corresponds to one of the $m\ell$ windows described in Theorem 2.7.15. □

Isomorphism with dyadic tilings

Dyadic Tilings

The numbers a_ℓ count the number of valid partial circuit configurations of $\mathcal{B}_\ell$, but they also happen to enumerate a different combinatorial structure: the number of dyadic tilings of the unit square of rank ℓ [27]. To facilitate the analysis of the gap of our code Hamiltonian, we will describe an explicit isomorphism between the two sets and the Markov chains defined on them.

Definition 2.7.18 (Dyadic tiling)**.** A dyadic tiling of rank ℓ is a tiling of the unit square by 2^ℓ *equal-area* dyadic rectangles, which are rectangles of the form $[a2^{-s}, (a+1)2^{-s}] \times [b2^{-t}, (b+1)2^{-t}]$, where a, b, s, t are nonnegative integers for some positive integer ℓ.

Figure 2.10 shows some examples for $\ell = 4$:

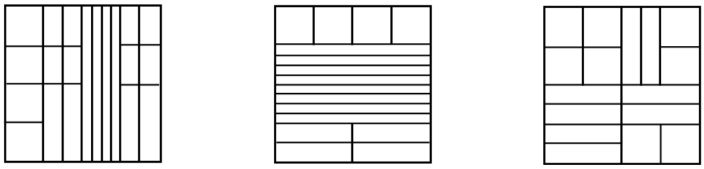


Figure 2.10: Examples of rank 4 dyadic tilings.

Each tiling of rank ℓ can be described recursively: beginning from the unit square, draw a line that is either a horizontal or vertical bisector. This divides the square into two rectangles of equal area.Then, choose two (not necessarily distinct) dyadic tilings of rank $\ell - 1$ and scale them to overlay with the two rectangles.

Definition 2.7.19. Let $\mathcal{T}_\ell$ be the set of dyadic tilings of rank ℓ.

There is a natural Markov chain on $\mathcal{T}_\ell$ called the *edge-flip* Markov chain [56]. Given a dyadic tiling, there is a distinguished set of edges in the tiling which can be removed and replaced by their perpendicular bisector to obtain another valid dyadic tiling of the same size. So, the transitions between states of the Markov chain are described by choosing one of the *flippable* edges uniformly at random and flipping it, obtaining a new tiling.

We can formally define the edge-flip Markov chain as follows:

Definition 2.7.20 (Edge-flip Markov chain [27, 56])**.** The edge-flip Markov Chain, $\mathcal{M}_\ell$ on state space $\mathcal{T}_\ell$ is defined with the following transition rule. Starting from state $m_i \in \mathcal{T}_\ell$:

- Choose a rectangle in the tiling m_i uniformly at random.
- Choose an edge e of the four edges of the rectangle (left, right, top, or bottom) uniformly at random.
- If e can be flipped to produce a new dyadic tiling in $\mathcal{T}_\ell$, then flip the edge and let m_{i+1} be the resulting tiling.
- If e cannot be flipped to produce a new dyadic tiling in $\mathcal{T}_\ell$, then choose a new edge at random and return to the previous step.

There exists an isomorphism between the valid configurations of $\mathcal{B}_\ell$ and the set of dyadic tilings of rank ℓ.That is, adding or removing a single gate from a partial configuration of $\mathcal{B}_\ell$ corresponds directly to a unique valid edge flip of the corresponding dyadic tiling (and vice-versa).

We will describe this isomorphism in a way that will build a visual intuition for it. First, we identify the empty configuration of $\mathcal{B}_\ell$ on 2^ℓ qubits with the tiling consisting entirely of horizontal cuts, t_0 s(the tiling consisting of 2^ℓ horizontal rectangles stacked on top of each other).

The tiling t_0 can be described by $2^\ell - 1$ horizontal cuts. We now describe these horizontal cuts as the disjoint union of smaller components we call c-segments. The following recursive procedure describes the segments:

- Begin with an empty square and initialize a counter $c = \ell$.
- Make a horizontal cut through the square and subdivide this cut into 2^{c-1} equal length segments, each a c-segment.
- If $c > 1$, decrement $c \to c - 1$ and repeat the previous step for *each* of the two empty rectangles (using the same c for each of the two) produced by the cut made in the last previous step. Otherwise, stop.

In the resulting representation of t_0, there are always $2^{\ell-1}$ c-segments for $c \in \{1, ..., \ell\}$. The set of $2^{\ell-1}$ c-segments are distributed evenly across 2^c horizontal cuts of t_0 in groups of $2^{\ell-c-1}$. We call the resulting representation of t_0 a *dressed* tiling. We give an example of t_0 for $\ell = 3$ in Figure 2.11 below, with the number labels for the c-segments $c = 1, 2, 3$ replaced with the colors blue, green, and red, respectively, for visual clarity:

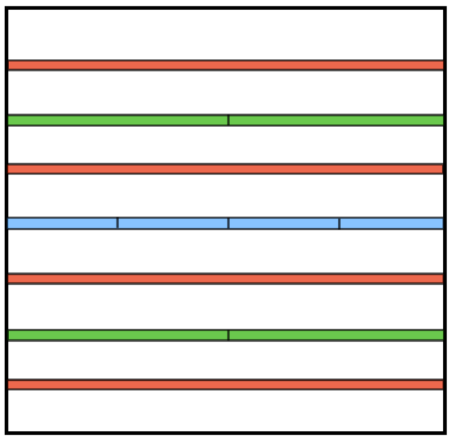


Figure 2.11: The dressed tiling t_0 for $\ell = 3$

To understand the edge-flip Markov chain, we need to understand how to determine which edges can be flipped (flipping that edge will take you from one dyadic tiling to another), and which can not.

Fact 2.7.21 ([27])**.** *A* flippable *edge of a tiling $t \in \mathcal{T}_\ell$ is a* c-segment *which, when considered alone, forms an entire edge of* both *dyadic rectangles it borders.*

This fact tells us that starting from t_0, and flipping flippable c-segments, we obtain the edge-flip Markov chain.

Next, we establish a bijection between the gates of $\mathcal{B}_\ell$ with the c-segments of the all horizontal tiling $t_0 \in \mathcal{T}_\ell$ in the following way: the set of $2^{\ell-1}$ c-segments corresponds to the set of $2^{\ell-1}$ gates in the cth layer of $\mathcal{B}_\ell$. Consider the following procedure for assigning each particular gate to a particular c-segment:

- Assign each ℓ-segment of t_0, from left to right, to the gates in layer $\mathcal{L}_\ell$, from top to bottom.
- For each gate in $\mathcal{L}_\ell$, identify the two gates in its past light-cone in $\mathcal{L}_{\ell-1}$ with the two nearest $\ell - 1$-segments sitting above and below the ℓ-segment in question.
- Continue this procedure recursively for the $\ell - 1$-segments all the way down to the 1-segments.

Figure 2.12 illustrates this bijection for $\ell = 3$.

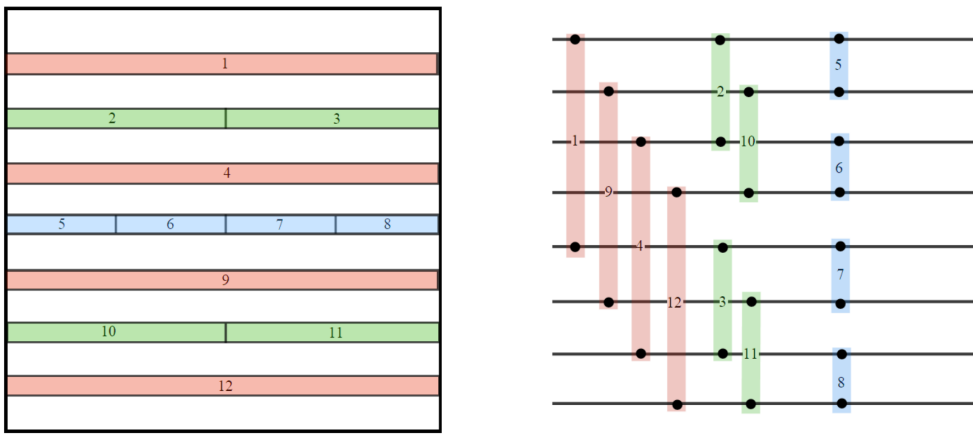


Figure 2.12: This example for $\ell = 3$ illustrates the correspondence between c-segments and their associated gates in t_0

Now, consider the following bijection between the partial configurations of $\mathcal{B}_\ell$ and dyadic tilings $\mathcal{T}_\ell$: For a tiling $t \in \mathcal{T}_\ell$, identify it with the circuit in which the only

gates applied are those for which the corresponding c-segments of the tiling are vertical. To see that the edge flip and circuit Markov chains are isomorphic, we need only see that the valid edge flips of a tiling correspond directly to the possible gate activations and deactivations of the corresponding partial configuration of $\mathcal{B}_\ell$.

But this is clear by the gate to c-segment identification procedure described above: for a c-segment to be flippable from horizontal to vertical (vertical to horizontal), it must comprise an entire edge of both dyadic rectangles that it borders. This only happens once the two nearest $c-1$-segments ($c+1$-segments) – the one directly above and the one directly below (directly left and directly right of) – have been flipped to be vertical (horizontal). By the bijection procedure outlined above, those two $c-1$ ($c+1$) segments correspond to the gates that directly precede the gate corresponding the c-segment in its past (future) light-cone. Figure 2.13, below, illustrates an example with $\ell = 3$.

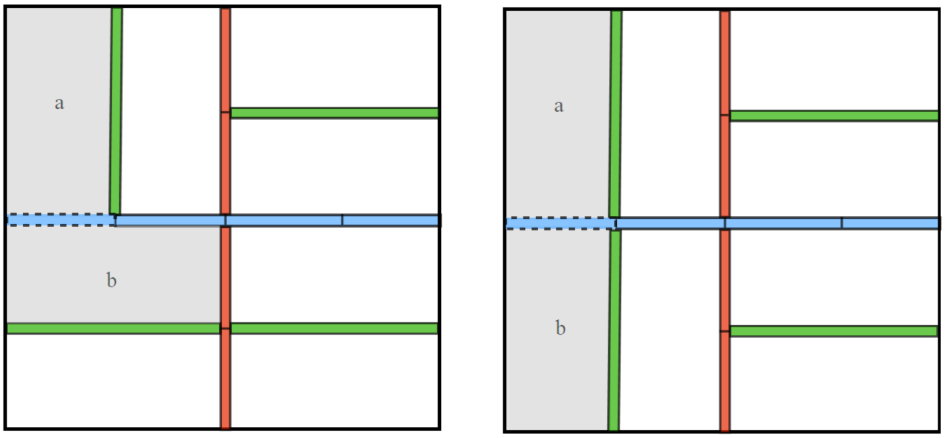


Figure 2.13: Consider the left-most 3-segment in both of these dyadic tilings. In the left tiling, we see that it can not be flipped from horizontal to vertical because only one of the 2-segments directly above and below it has been flipped to vertical. Consequently, this 3-segment is a complete edge of the dyadic rectangle a but not rectangle b. In the right tiling we see that this is remedied by flipping the remaining nearest 2-segment.

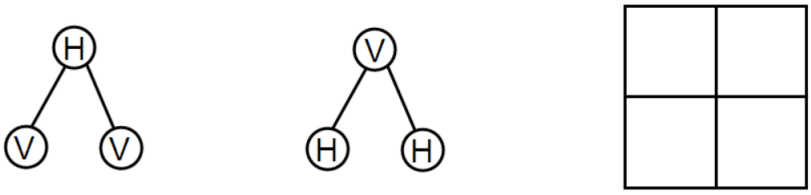
HV-Trees

In order to give a more complete understanding of the isomorphism between valid partial configurations of $\mathcal{B}_\ell$ and $\mathcal{T}_\ell$, we introduce an alternate representation of dyadic tilings called *HV*-trees [56].

Janson, Randall, and Spencer showed that the recursive description of a tiling gives an easy isomorphism to a graph called a *HV*-tree [56]. Consider a complete binary tree of depth ℓ where each vertex is labeled either H or V. There is a clear mapping from such trees to dyadic tilings: starting with the unit square and the root of the tree, draw a horizontal (H-cut) bisector or vertical (V-cut) bisector depending on the

label of the root. Then recursively draw H- or V-cuts by the labels of the children on the two generated rectangles.

This mapping isn't a bijection, however. It is easy to see that the following two HV-trees produce the same dyadic tiling.



[56] noticed that collisions only occur when there is a cross in the dyadic tiling, either a H-cut followed by 2 V-cut children or a V-cut followed by 2 H-cut children. By disallowing any H-vertex to have both children be V-vertexes, we obtain an isomorphism.

Definition 2.7.22 (HV-trees)**.** An HV-tree of depth ℓ is a complete binary tree of depth ℓ with each vertex labeled either H or V and the restriction that no H-vertex has both children labeled V.

Theorem 2.7.23 ([56])**.** *There is an isomorphism between $\mathcal{T}_\ell$, the set of dyadic tilings of rank ℓ, and the set of HV-trees of depth ℓ.*

Proof. We previously described the mapping from HV-trees to dyadic tilings. For the other direction, look at the unit square. If there is a V-cut, label the root V and proceed recursively. Otherwise, there must exist a H-cut (Theorem 1.1 of [56]) label the root H and proceed recursively. Note that by choosing a V-cut if both cuts exist ensures that the generated tree satisfies the HV-condition. □

We now prove the isomorphism between valid configurations of the bitonic block $\mathcal{B}_\ell$ and $\mathcal{T}_\ell$; it suffices to show the isomorphism between bitonic blocks and HV-trees. Recall Corollary 2.7.3: Any valid configuration of $\mathcal{B}_\ell$ must have every gate in $\mathcal{L}_1$ activated or every gate in layer $\mathcal{L}_\ell$ is not activated. Call valid configurations satisfying the first property v-configurations and call configurations satisfying the second property h-configurations.

Notice that given a v-configuration, we can recursively specify it by describing the configuration of the last $\ell - 1$ layers of $\mathcal{B}_\ell$, which are isomorphic to $\mathcal{B}_{\ell-1}^{\otimes 2}$ (Corollary 2.7.9). Similarly, given a h-configuration, we can recursively specify it by describing the configuration of the first $\ell - 1$ layers of $\mathcal{B}_\ell$. A configuration that is both a v-configuration and h-configuration, can be recursively specified by the configuration of the middle $\ell - 2$ layers which are isomorphic to $\mathcal{B}_{\ell-2}^{\otimes 4}$.

Theorem 2.7.24. *There is an isomorphism between the set of valid configurations of $\mathcal{B}_\ell$ and the set of HV-trees of depth ℓ. With Theorem 2.7.23, this proves an isomorphism between the set of valid configurations of $\mathcal{B}_\ell$ and $\mathcal{T}_\ell$.*

Proof. Given a HV-tree, if the root is a V-vertex, we activate all the gates in $\mathcal{L}_1$. We proceed recursively using the two children of the root to describe the configuration on the last $\ell - 1$ layers with each child describing the configuration on one of the blocks $\mathcal{B}_{\ell-1}$. Likewise, if the root is a H-vertex, we set all the gates in $\mathcal{L}_\ell$ as not activated and proceed recursively on the first $\ell - 1$ layers.

Given a configuration, we know it must be a v-configuration or h-configuration. If it is a v-configuration, we set the root as a V-vertex and build the tree recursively with the blocks $\mathcal{B}_{\ell-1}$ in the last $\ell - 1$ layers describing the children sub-trees. Likewise, if it is a h-configuration, we set the root as a H-vertex and build the tree recursively with the blocks $\mathcal{B}_{\ell-1}$ in the first $\ell - 1$ layers describing the children sub-trees.

Notice, that by checking if a configuration is a v-configuration before checking if it is a h-configuration, we ensure the HV-tree property. □

Indexing configurations

Inspired by the uniform sampling algorithm for dyadic tilings of Janson, Randall, and Spencer [56], we adapt, using the isomorphisms in Theorem 2.7.24, them to generate an indexing algorithm for valid configurations of a bitonic block.

Theorem 2.7.25. *There exists an isomorphism between $[a_\ell] = \{1, 2, \ldots, a_\ell\}$ and the set of valid configurations of bitonic block $\mathcal{B}_\ell$. Furthermore, both maps are efficiently calculable.*

Proof. Theorem 2.7.10, tells us that these sets have the same magnitude. Corollary 2.7.11, tells us that the number of v-configurations is $a_{\ell-1}^2$ and the the number of h-configurations which are not v-configurations is $a_\ell - a_{\ell-1}^2$.

Divide the set $[a_\ell]$ into $S_V = [a_{\ell-1}^2]$ and $S_H = a_{\ell-1}^2 + [a_\ell - a_{\ell-1}^2]$. Given an index $i \in [a_\ell]$, we use these disjoint sets to decide whether the configuration is a v-configuration or h-configuration. If $i \in S_V$, then we set $\mathcal{L}_1$ as activated and we express i uniquely as $i_L a_{\ell-1} + i_R$ and then recursively, using i_L and i_R as indices, decide the configurations on the two bitonic blocks $\mathcal{B}_{\ell-1}$ generating the last $\ell - 1$ layers. The case of $i \in S_R$ is a bit more subtle. Since we are choosing a h-configuration, we know its children cannot both be v-configurations. Therefore, we divide S_R into 3 parts, corresponding to the bitonic blocks on the first $\ell - 1$ layers being both h-configurations, or one being a h-configuration and the other a v-configuration. It is not difficult to check that there are $(a_{\ell-1} - a_{\ell-2}^2)^2$ configurations with both children being h-configurations and $a_{\ell-2}^2(a_{\ell-1} - a_{\ell-2})^2$ for the other two cases. Now, we can proceed recursively.

For the other direction, given a valid configuration, we decide if it is a v-configuration or h-configuration. If a v-configuration, we recursively decide the index within S_V and output it. Otherwise, we recursively decide the index within S_R, add $a^2_{\ell-1}$ and output it.

We note that in both directions, the smaller bitonic blocks will involve permuted indexes. However, since Lemma 2.7.7 is efficient, this is not an issue. □

A similar proof holds for $\mathcal{B}^{\times m}_\ell$ and $\mathcal{B}^{\leftrightarrow m}_\ell$.

Theorem 2.7.26. *There exists an isomorphism between $[a^{\times m}_\ell]$ and the set of valid configurations of architecture $\mathcal{B}^{\times m}_\ell$. Likewise, there exists an isomorphism between $[a^{\leftrightarrow m}_\ell]$ and the set of valid configurations of circular architecture $\mathcal{B}^{\leftrightarrow m}_\ell$. Furthermore, both maps are efficiently calculable.*

Proof. The proof is nearly identical except the initial partition of $[a^{\times m}_\ell]$ is based on the window that the configuration lies in. We note configurations of all windows except the last must be h-configurations. For the case of circular architecture, all windows are h-configurations. □

Chapter 3

TENSOR NETWORK MAXIMUM LIKELIHOOD DECODERS FOR CIRCUIT-LEVEL NOISE

3.1 Introduction

In this final chapter we study quantum error correcting codes in a much more concrete and practical setting. As experimental progress in quantum computing hardware marches forwards, exciting proposals for fault-tolerant quantum computing architectures based on such hardware are being proposed [28]. One of the major challenges in quantum computing is the design of fault tolerant error correction (FTEC) schemes that will perform well enough to implement demonstrations of error corrected quantum computing with near term (and hopefully long term) hardware.

The performance of a FTEC scheme can only be as good as its decoder: the map which takes information measured from some encoded quantum state that may have suffered some error and infers a correction that can be applied to the state to eliminate the error. The design of a decoder has two primary goals that often result in a trade off: accuracy and speed. First we want the decoder to be correct as often as possible, yielding the lowest logical failure rates that we can achieve. Second, we require that the decoder be implementable in a way that it can produce corrections fast enough to actually be able to correct errors as they occur on the time scales relevant to the specific application.

Usually, implementing an optimal decoding scheme is impractical, so in practice efficient decoders are designed based on heuristics chosen based on some underlying mathematical structure of the code in question. One example being the surface code and minimum weight perfect matching decoders [37] [28]. Such efficient purpose-built decoders perform well enough that they are helping shape the paradigm for how we think of what can be achieved with current and near term architecture proposals [28].

Given an efficient sub-optimal decoder for a quantum error correcting code, we can ask two questions:

1. How large is the gap in performance compared to an optimal decoder?
2. Can we design an efficient decoder that performs better in this particular application?

In this chapter we provide a tool for answering both questions. We study optimal and approximately optimal decoders for general stabilizer codes under circuit level

noise models. We describe how implement optimal maximum likelihood (ML) decoders using a tool called the Circuit History Code (CHC) based on the sparse code construction seen in [12]. The CHC allows us to reformulate the problem of maximum likelihood decoding as the contraction of a tensor network, in a sense extending the work of Bravyi, Suchara, and Vargo [22] to the setting of circuit level noise. This maps optimal decoding of circuit level noise into a problem that has been thoroughly studied, granting us access to techniques that have been developed to make it more tractable.

Exact ML decoding and tensor network contraction are generally difficult problems whose complexity scales exponentially in the problem size. However, many different techniques for approximate tensor network contraction exist and allow a tradeoff in complexity for accuracy by setting a fixed bond dimension during the contraction of the tensor network. This reduces the time complexity of tensor network contraction to be polynomial in the size of the network and the fixed bond dimension. As demonstrated in [22] and [31] for code capacity noise models this allows one to perform *approximate* maximum likelihood decoding which may provide near-optimal performance at relatively low fixed bond dimension.

In section 3.2, we remind the reader of important definitions and properties of stabilizer codes that will be needed to describe our construction. Next, in section 3.3 we discuss circuit level noise models in detail, establishing mathematical structure that we will be needed to understand the CHC construction. In section 3.4 we describe the CHC informally by way of example in order to help give the reader intuition for the formal mathematical description of the construction which we give in section 3.5. In section 3.6 we describe the decoding problem for circuit level noise, define maximum likelihood decoding in this setting, and give a construction that use the CHC to produce a tensor network whose contraction implements maximum likelihood decoding. Finally, in section 3.7, we describe preliminary numerical results comparing the performance of our ML decoder to MWPM for a standard rotated surface code, and also give evidence that in a biased noise model, using an XY surface code with approximate ML decoding can give significant improvements over using a standard XZ surface code with a matching decoder.

3.2 Algebraic Structure of Quantum Stabilizer Codes

Note: for this entire chapter we will be ignoring phases on pauli operators. Thus when we refer to the pauli group $\mathcal{P}_n$, we really mean the Pauli group quotiented by $\{\pm I, \pm iI\}$.

Definition 3.2.1. A stabilizer code on n qubits (referred to as physical qubits), is defined by choosing a set of $m = n - k$ ($k \geqslant 0$) independent commuting elements $\{M_i\}_{i=1}^m$ of the n-qubit Pauli group, $\mathcal{P}_n$. These commuting generators generate the *stabilizer group* of the code, $\mathcal{S} = \langle M_i \rangle_{i=1}^m$, an abelian subgroup of $\mathcal{P}_n$. The

simultaneous +1 eigenspace of the stabilizer generators $\{M_i\}_{i=1}^m$ is a subspace of the n-qubit Hilbert space, $\mathcal{H}_{code} \simeq \mathbb{C}^{\otimes 2^k} \subseteq \mathbb{C}^{\otimes 2^n}$, referred to as the *codespace* of the code. We say that such a stabilizer code encodes k logical qubits into n physical qubits.

Definition 3.2.2. The *logical group* of a stabilizer code $\mathcal{S}$ is defined as the quotient of the normalizer of $\mathcal{S}$ by $\mathcal{S}$ itself:

$$\mathcal{L} = N(\mathcal{S})/\mathcal{S}. \tag{3.2.1}$$

We may describe $\mathcal{L}$ by choosing $2k$ generators $\{L_{X,i}, L_{Z,i}\}_{i=1}^k$ from $N(\mathcal{S})$, which we think of as specific representatives of the cosets of $\mathcal{S}$ in $N(\mathcal{S})$, and which satisfy the following relations:

$$\{L_{X,i}, L_{Z,i}\} = 0, \ \forall\, i \tag{3.2.2}$$
$$[L_{X,i}, L_{Z,j}] = 0, \ \forall\, i \neq j \tag{3.2.3}$$
$$[L_{X,i}, L_{X,j}] = 0, \ \forall\, i, j \tag{3.2.4}$$
$$[L_{Z,i}, L_{Z,j}] = 0, \ \forall\, i, j. \tag{3.2.5}$$

Definition 3.2.3. Suppose we have a stabilizer code on n physical qubits with stabilizer group $\mathcal{S} = \langle S_i \rangle_{i=1}^m$, and assume fixed representatives of the logical operators $\mathcal{L} = \langle L_{X_i}, L_{Z_i} \rangle_{i=1}^{n-m}$. Given this fixed set of generators for $\mathcal{S}$ and $\mathcal{L}$, we can define a unique (up to certain stabilizer transformations) set of *pure error generators* which generate the group of pure errors, $\mathcal{T} = \langle T_i \rangle_{i=1}^m$. The pure error generators T_i are specifically chosen such that

$$[T_i, S_j] = 2T_i S_j \delta_{ij} \quad \forall\, i, j \tag{3.2.6}$$

and

$$[T_i, T_j] = 0 \quad \forall\, i, j \tag{3.2.7}$$

and

$$[T_i, L] = 0 \quad \forall\, i \quad \forall\, L \in \mathcal{L}. \tag{3.2.8}$$

The above commutation relations say that the pure error generator T_i anticommutes *only* with the stabilizer generator S_i and commutes with all other generators of $\mathcal{S}$, $\mathcal{L}$, and $\mathcal{T}$. In this sense they can be thought of as canonical conjugates of the stabilizer generators M_i.

The generators of the three groups $\mathcal{S}$, $\mathcal{L}$ and $\mathcal{T}$ give $m + m + 2(n-m) = 2n$ indepent Pauli operators in $\mathcal{P}_n$ and therefore the generators of the three groups $\mathcal{S}$, $\mathcal{L}$ and $\mathcal{T}$ form a minimal generating set for the Pauli group on n qubits $\mathcal{P}_n$. This allows us a unique decomposition of any Pauli operator $E \in \mathcal{P}_n$ (up to $\pm$ phase):

Definition 3.2.4. Given a description of a stabilizer code with pure errors and logical operators ($\mathcal{S}$,$\mathcal{T}$,$\mathcal{L}$) on n qubits, any $E \in \mathcal{P}_n$ has a unique decomposition of the following form:

$$E = LTS, \quad L \in \mathcal{L} \quad T \in \mathcal{T} \quad S \in \mathcal{S}. \tag{3.2.9}$$

The above will be referred to as *the LTS decomposition* of E.

For notational convenience and clarity, we define the following maps which isolate different components of a Pauli operator's LTS decomposition:

Definition 3.2.5. Suppose we have $E \in \mathcal{P}_n$ with LTS decomposition

$$E = LTS, \quad L \in \mathcal{L} \quad T \in \mathcal{T} \quad S \in \mathcal{S}. \tag{3.2.10}$$

Then we define the maps $\mathbb{T}_a : \mathcal{P}_n \to \mathcal{T}$ and $\mathbb{L} : \mathcal{P}_n \to \mathcal{L}$ to project the error onto the corresponding subgroup:

$$\mathbb{T}_a(E) = T \tag{3.2.11}$$

$$\mathbb{L}(E) = L. \tag{3.2.12}$$

Definition 3.2.6. The *actual syndrome* associated with a Pauli error E is an m-bit string whose ith bit is 1 if and only if E anti-commutes with the ith stabilizer generator M_i. We introduce the map $\mathfrak{t}_a$ which maps E to its actual syndrome:

$$\mathfrak{t}_a : \mathcal{P}_n \to \{0,1\}^m. \tag{3.2.13}$$

When a fixed Pauli error E is understood, its actual syndrome will be referred to as $t_a = \mathfrak{t}_a(E)$.

Note that the pure error part T of the Pauli error $E \in \mathcal{P}_n$ completely determines the syndrome that one would obtain for E by measuring each M_i perfectly (assuming no measurement errors). In particular, given a syndrome $t_a = (t_{a_1}, t_{a_2}, ..., t_{a_m})$ with $t_{a_i} \in \{0,1\}$ we can refer to the corresponding pure error

$$T_{t_a} = \prod_{i=1}^{m} T_i^{t_{a_i}}. \tag{3.2.14}$$

Clearly all pure errors $T \in \mathcal{T}$ can be written this way, so for any syndrome $t_a \in \{0,1\}^m$ there is a corresponding pure error $T_{t_a} \in \mathcal{T}$, and vice-versa. From this point onward we will always label a pure error operator with its corresponding syndrome as a subscript (i.e. T_{t_a}).

A simple example of a stabilizer code that we will return to and build upon throughout this chapter is be the 3-qubit phase-flip repetition code. This code has a stabilizer with two generators:

$$\mathcal{S} = \langle M_1 = XXI, M_2 = IXX \rangle. \tag{3.2.15}$$

There are two associated pure error generators

$$\mathcal{T} = \langle T_1 = ZII, T_2 = IIZ \rangle, \tag{3.2.16}$$

and we may choose logical operator generators

$$\mathcal{L} = \langle L_Z = ZZZ, L_X = IXI \rangle. \tag{3.2.17}$$

In summary: when referring to a Pauli error $E \in \mathcal{P}_n$ with LTS decomposition $E = LT_{t_a}S$, we will refer to the operator $\mathbb{L}(E) = L \in \mathcal{L}$ as E's *logical class* or *logical component*, the operator $\mathbb{T}(E) = T_{t_a} \in \mathcal{T}$ as its *pure error*, and $\mathfrak{t}_a(E) = t_a \in \{0,1\}^m$ as its corresponding *actual* syndrome. We may then refer to the equivalence class of Pauli errors with logical class L and pure error T_{t_a} (equivalently syndrome t_a) as

$$\mathcal{E}_{L,t_a} = \{E = LT_{t_a}S | S \in \mathcal{S}\}. \tag{3.2.18}$$

Extended Stabilizer Code

To improve the clarity with which we discuss the circuit history code construction, we also introduce the notion of an *extended stabilizer code*. The idea is as follows: for stabilizer code $\mathcal{S}$ on n qubits with m generators $\{M_i\}_{i=1}^m$, we may construct a stabilizer code on $n + rm$ physical qubits.

Definition 3.2.7. The extended stabilizer group $\mathcal{S}^{ext}$ has generators $\{M_i\}_{i=1}^m \cup \{M_i^{ext}\}_{i=1}^{rm}$. The original generators $\{M_i\}_{i=1}^m$ just act on the first n physical qubits, while each additional new generator M_i^{ext} acts non-trivially on only the $n + i$th physical qubit of the extended code via an X or Z operator. It is clear that this is a stabilizer code no matter what non-trivial Pauli M_i^{ext} acts on qubit $n + i$ with.

The idea is that the rm additional qubits that extend the code, and their associated stabilizers, model observed syndromes learned from imperfect measurements of the base stabilizer generators $\{M_i\}_{i=1}^m$ in a circuit-level noise model. We allow for $r \geqslant 1$ to account for the case where syndrome measurements are repeated. When stabilizer measurements are noisy, we refer to an *observed syndrome*, denoted t_o, which may differ from the actual syndrome t_a that one would have observed if there were no measurement errors. The violation of extended stabilizer M_i^{ext} indicates that the outcome of some possibly faulty measurement of base stabilizer generator M_i was 1, and is otherwise assumed to be 0.

Definition 3.2.8. For each extended stabilizer generator, M_i^{ext}, we have an *extended pure error* T_i^{ext} which is simply the canonical conjugate of M_i^{ext}.

For example, if $M_i^{ext} = Z_{n+i}$, then $T_i^{ext} = X_{n+i}$. As with any stabilizer code, we may use the LTS decomposition on an extended stabilizer code. When we do so, we will make a point of expressing the pure error in two parts corresponding to the actual and observed syndromes explicitly.

$$T = T_{t_a} \otimes T_{t_o}. \tag{3.2.19}$$

It will be helpful to define two more maps $\mathbb{T}$ and $\mathbb{T}_o$:

Definition 3.2.9. When discussing the LTS decomposition of an extended stabilizer code we'll use the map $\mathbb{T}$ to project onto the entire pure error

$$\mathbb{T}(E) = T_{t_a} \otimes T_{t_o}, \tag{3.2.20}$$

and the map $\mathbb{T}_o$ to project onto the extended part of the pure error corresponding to the observed syndrome

$$\mathbb{T}_o(E) = T_{t_o}. \tag{3.2.21}$$

Finally, we define the map $\mathfrak{t}_o$ which isolates the observed error of an error of an extended stabilizer code:

$$\mathfrak{t}_o(E) = t_o \in \{0,1\}^{rm}. \tag{3.2.22}$$

We illustrate the concept using the 3-qubit phase-flip repetition code. We may construct an extended version of this code has a stabilizer with four generators:

$$\mathcal{S}^{ext} = \langle M_1 = XXIII, M_2 = IXXII, M_1^{ext} = IIIXI, M_2^{ext} = IIIIX \rangle. \tag{3.2.23}$$

There are two associated pure error generators

$$\mathcal{T}^{ext} = \langle T_1 = ZIIII, T_2 = IIZII, T_1^{ext} = IIIZI, T_2^{ext} = IIIIZ \rangle, \tag{3.2.24}$$

and we may choose logical operator generators

$$\mathcal{L} = \langle L_Z = ZZZII, L_X = IXIII \rangle. \tag{3.2.25}$$

3.3 Measurement Circuits, Circuit Level Noise, and Error Histories

Noise Models

Now we briefly review the most common types of error models used to study and assess the performance of stabilizer codes.

Code Capacity Noise

A code capacity error model is specified by a probability distribution over the n-qubit Pauli group,

$$\pi_{CC} : \mathcal{P}_n \to [0, 1]. \tag{3.3.1}$$

therefore, any pauli error E occurs with probability $\pi_{CC}(E)$. In this model we assume the capability of being able to perfectly measure the stabilizer generators $\{M_i\}$ and obtain the actual syndrome $\mathfrak{t}_a(E) = t_a$. The decoder may then use the actual syndrome t_a to decide on a correction.

Phenomenological Noise

A phenomenological noise model is a code capacity noise model in which each measurement of a stabilizer generator M_i fails with some probability q, flipping that syndrome bit. This means that the observed syndrome t_o that one actually receives from the stabilizer measurements differs from the actual syndrome associated with the error, t_a. This inequivalence between the observed and actual syndromes adds considerable complication to the decoding problem, as we shall discuss in section 3.6.

Circuit Level Noise

Although phenomenological noise models capture the notion of faulty measurements, they ignore the fact that the quantum circuits used to perform stabilizer measurements are made up of individual hardware components, each of which can be prone to failure. Such failures that occur at different points throughout the execution of the stabilizer measurement circuit create errors that propagate and accumulate, leading to correlations between the observed syndrome and final error on the data qubits. Phenomenological noise models fail to capture these correlations, but circuit level noise models succeed.

A circuit level noise model is based on the quantum circuit that one would use to carry out stabilizer measurements for the underlying code. Every state preparation, measurement, and logic gate (including qubit idles) involved in this circuit is assigned a probability of producing a Pauli error.

In order to proceed in defining a circuit level noise model, we must discuss the circuits whose noise we are trying to model. First we standardize our description of a stabilizer code measurement circuit.

Definition 3.3.1. A *measurement protocol* for a stabilizer code, $\mathcal{M}$, is a list of stabilizers operators selected from $\mathcal{S}$ to measure. The operators in $\mathcal{M}$ need not be generators and may appear in $\mathcal{M}$ more than once. For the purposes of this manuscript, we will be interested in the case where the measurement protocol $\mathcal{M}$ consists of

the generators $\{M_i\}_{i=1}^m$ of $\mathcal{S}$, repeated an integer r multiple of times. For a fixed stabilizer code $\mathcal{S}$, we will denote such a measurement protocol as a *standard repeated measurement protocol* $\mathcal{M}_{\mathcal{S},r}$.

Given a standard repeated measurement protocol, we can then build an associated measurement circuit $C_{\mathcal{S},r}$ that implements this measurement protocol:

Definition 3.3.2. A *measurement circuit* $C_{\mathcal{S},r}$ for a standard repeated measurement protocol $\mathcal{M}_{\mathcal{S},r}$, is a quantum circuit on $n+m$ qubits, n of which are data qubits of the base stabilizer code, and the remaining m qubits are ancillas used to produce measurement outcomes for the operators in the measurement protocol $\mathcal{M}_{\mathcal{S},r}$. A circuit for r measurement rounds consists of an $r = 1$ circuit repeated r times, with measurement ancillas re-initialized between each round. An $r = 1$ circuit is almost completely specified by the following:

- In the first time step, data qubits idle while each ancilla qubit is initialized in the $|+\rangle$ state.
- In the following D time steps, data and ancilla qubits are entangled using $CX/CY/CZ$ gates as follows: For each data qubit j on which M_i acts non-trivially with an $X/Y/Z$ operator, data qubit j and ancilla qubit $n+i$ are coupled via $CX_{n+i,j}/CY_{n+i,j}/CZ_{j,n+i}$ (first qubit is control, second is target). (TODO: footnote about fault-tolerance).
- In the final time step, data qubits idle while ancilla qubits are measured in the X basis.

The only ambiguity in this description is how the different individual entangling gates are scheduled during the D time steps. Different code architectures benefit from different gate scheduling specifics, and the choice of specific schedule can be important to preserving fault-tolerance. Moving forward, we will assume that some schedule has been specified or provide one. We note, however, that details of gate scheduling do not change the validity of what follows. Finally, we remark that any circuit as specified above is a *Clifford circuit*, which will allow us to easily understand the cumulative effect of errors that occur during the execution of the circuit.

We illustrate an example of $C_{\mathcal{S},1}$, where $\mathcal{S}$ is the 3-qubit phase-flip repetition code, in figure 3.1, below. Figure 3.2, below, illustrates $C_{\mathcal{S},2}$.

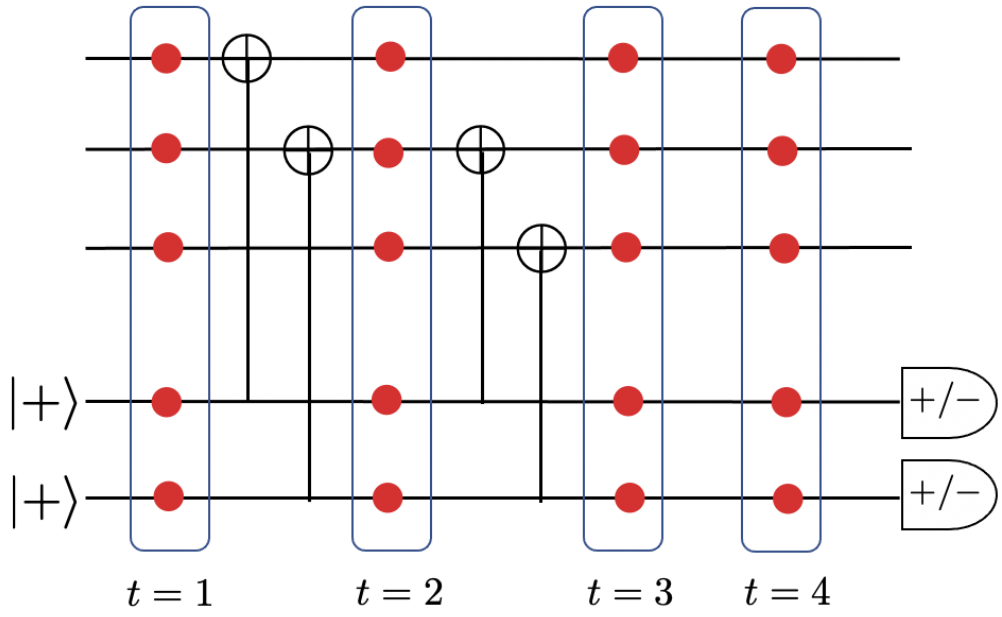


Figure 3.1: The circuit $C_{\mathcal{S},1}$ for the standard measurement protocol $\mathcal{M}_{\mathcal{S},1}$, where $\mathcal{S}$ is the three-qubit phase-flip repetition code. We note that there are four time slices per qubit during which faults may occur: During $t = 1$ the data qubits may experience idling faults that occur while the ancilla qubits are being initialized, and the ancilla qubits may experience failures from their initialization. During $t = 2$ and $t = 3$, all qubits may experience faults from the execution of CNOT gates or from idling. During $t = 4$, the data qubits experience idling errors while the ancilla qubits are measured, and the fault locations on the ancilla qubits model faults that occur during these measurement.

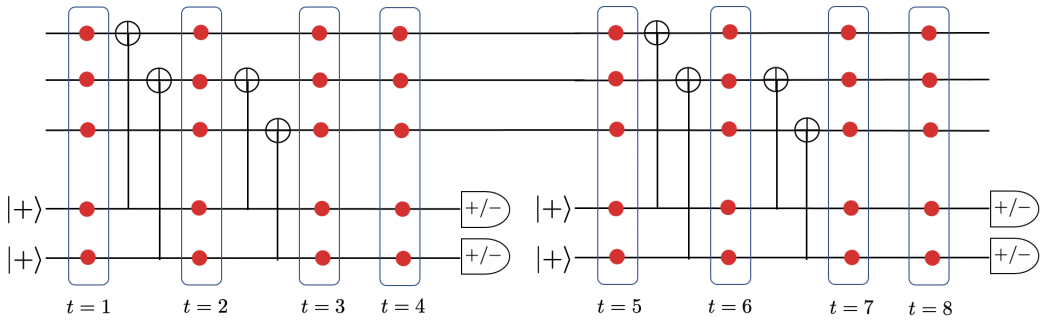


Figure 3.2: The circuit for the standard measurement protocol $\mathcal{M}_{\mathcal{S},2}$, where $\mathcal{S}$ is the three-qubit phase-flip repetition code. Between measurement repetitions, ancilla qubits are re-initialized.

In the above examples, we have specified each of the space-time locations throughout the execution of the circuit where we would like to model the possibility of a failure. We will call each of these locations *fault locations*, and enumerate them $f_i \in F$. Generally speaking, a fault location is meant to model a fault that occurs on a qubit

from: faulty execution of a single or two-qubit gate, decoherence during idling, faulty ancilla state preparation, or faulty ancilla state measurement.

As we can see in the figures above, we can organize fault locations into time slices. Each measurement repetition features a time slice related to ancilla initialization, a time slice related to ancilla measurement, and one time slice following each of the $D = 2$ layers of entangling gates between data and ancilla qubits.

We see that, in general, the number of fault locations in the circuit $C_{\mathcal{S},r}$ for the standard measurement protocol $\mathcal{M}_{\mathcal{S},r}$ is

$$N_{fault} = r(D+2)(n+m). \tag{3.3.2}$$

Specifying a single qubit pauli error which occurs at each fault location $f_i \in F$ specifies what we will refer to as an error history:

Definition 3.3.3. Suppose that the circuit has a total of K time-slices during which faults can occur. At the kth time slice of such a circuit, some pauli error $E^k \in \mathcal{P}_{n+m}$ acts on the data and ancilla qubits. This allows us to define an *error history* for the circuit, $E_{hist} \in \mathcal{P}_{n+m}^{\otimes K}$,

$$E_{hist} = \bigotimes_{k=1}^{K} E^k. \tag{3.3.3}$$

Definition 3.3.4. A circuit level error model for a measurement circuit $C_{\mathcal{S},r}$ is specified by assigning a set of probability distributions $\{\pi_i\}_i$ over single or multiple-qubit Pauli groups such that the product of these distributions yields a probability distribution π_{CC} over error histories for the circuit:

$$\pi_{CC} : \mathcal{P}_{n+m}^{\otimes K} \to [0,1] \tag{3.3.4}$$

$$\pi_{CC} = \prod_i \pi_i. \tag{3.3.5}$$

The individual probability distributions π_i may be over single-qubit or multi-qubit faults. As an example, for faults that occur on qubits due to a faulty CX gate, one may prefer to specify a single distrubtion over two-qubit faults $\pi : \mathcal{P}_2 \to [0,1]$, rather than two single-qubit fault distributions $\pi : \mathcal{P} \to [0,1]$, so that the error model can capture correlations between the errors on the two qubits that interact through the CNOT gate.

So, we see that drawing an error history from a circuit level error model tells us an error that occurs on each qubit at each timestep of the circuit's execution.

The next question we examine is: how does an error history ultimately cause an error on the encoded data qubits, and what information about this error can we learn

from our measurement circuit? Each time slice of an error history carries forward to future timesteps and eventually, at the end of the circuit's execution, each qubit has an error which is the result of the accumulation of all individual errors in the error history. Next we discuss the dynamics of these errors and how they give rise to the final accumulated error.

Definition 3.3.5. The *boundary* $\mathcal{B}$ of a standard repeated measurement protocol circuit $C_{\mathcal{S},r}$ consists of the set of final fault locations of each data qubit of the base code, as well as each fault location of an ancilla qubit directly preceding a measurement. The boundary $\mathcal{B}$ of $C_{\mathcal{S},r}$ consists of $n + rm$ fault locations.

We distinguish the fault locations in $\mathcal{B}$ because they are where all faults in an error history will accumulate to by the end of the circuit's execution. We illustrate the boundary of $C_{\mathcal{S},2}$ in figure 3.3, below.

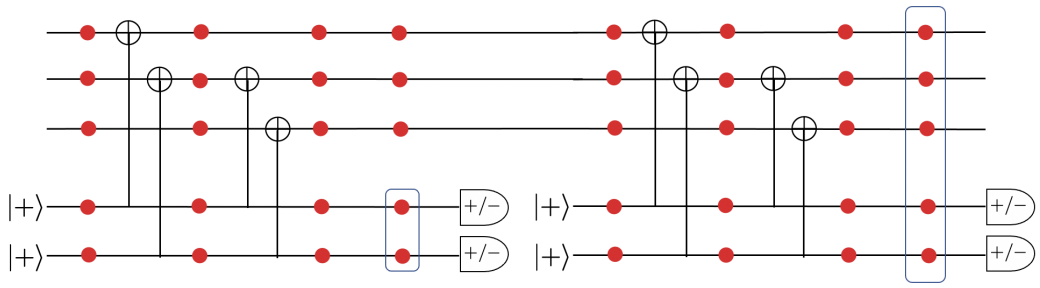


Figure 3.3: The circuit for the standard measurement protocol $\mathcal{M}_{\mathcal{S},2}$, where $\mathcal{S}$ is the three-qubit phase-flip repetition code. The 8 boundary locations are circled.

Definition 3.3.6. A circuit error history $E_{hist} \in \mathcal{P}_{n+m}^{\otimes K}$ can be mapped to an *accumulated error* of the circuit E_{acc} by taking each time slice E^k, propagating it through the remaining gates of the circuit to the output boundary of the circuit, and taking the product of the results for all E^k.

The accumulated error $E_{acc} \in \mathcal{P}_{n+rm}$ represents the entire accumulated error on both the data and ancilla qubits throughout the execution of the measurement circuit $C_{\mathcal{S},r}$, and can be analyzed using the LTS decomposition of the extended stabilizer code of the base code, $\mathcal{S}^{ext}$, and it will be helpful to analyze it from this point of view. The accumulated error can be broken into two parts, one acting on the data qubits/codeblock (an error of the base stabilizer code) and the other part acting on each repeated copy of the ancilla qubits (which together form the extension of the base stabilizer code):

$$E_{acc} = E_{acc,data} \otimes E_{acc,anc} \quad E_{acc,data} \in \mathcal{P}_n,\ E_{acc,anc} \in \mathcal{P}_{rm} \tag{3.3.6}$$

.

Now we describe how to compute the accumulated error E_{acc} of a given error history E_{hist}.

Definition 3.3.7. The *circuit unitary* of a single-repetition protocol $\mathcal{M}_{\mathcal{S},1}$ is the unitary operator $U_{\mathcal{S}}$ acting on $n+m$ qubits obtained from taking the circuit $C_{\mathcal{S},1}$ and removing all qubit preparations and measurements. The circuit unitary can be decomposed into a sequence of $D+1$ unitary operator time slices: $\{U_{\mathcal{S},k}\}_{k=1}^{D+1}$, where $U_{\mathcal{S},k}$ is the layer of the circuit acting after the kth set of fault locations. Then,

$$U_{\mathcal{S}} = U_{\mathcal{S},D+1}...U_{\mathcal{S},1}. \tag{3.3.7}$$

With this decomposition of the circuit unitary understood, we may use the individual $U_{\mathcal{S},k}$ to propagate a time-slice of the error history E^k forward by a single time-step.

Definition 3.3.8. Denote the action by conjugation of a unitary slice on a single error history time slice as $U_{\mathcal{S},k}E^jU_{\mathcal{S},k}^{\dagger} = U_{\mathcal{S},k} \bullet E^j$.

Lemma 3.3.9. *If U is a Clifford gate, then $U\bullet$ is a group homomorphism of $\mathcal{P}_{n+m}$ to itself.*

Proof. This is evident since $U\bullet$ is a group action of an element of the unitary group on a subgroup of the unitary group. □

The effect that the individual qubit errors occuring at a specific single time-slice of the error history, E^k, have caused by the end of the circuit's execution can be computed by using the $U_{\mathcal{S},k}$ operators to propagate E^k forward in time to the boundary of $C_{\mathcal{S},1}$.

Definition 3.3.10. The *accumulation of circuit error history time slice E^k*, $\mathcal{A}_k(E^k)$ is the result of conjugating E^k through to the end of the circuit past all of the layers $U_{\mathcal{S},k}$ that come after it:

$$\mathcal{A}_k(E^k) = U_{\mathcal{S},D+1} \bullet U_{\mathcal{S},D} \bullet ...U_{\mathcal{S},k} \bullet E^k \tag{3.3.8}$$

Lemma 3.3.11. *For a a clifford circuit, the map $\mathcal{A}_k$ is a group homomorphism of $\mathcal{P}_{n+m}$ to itself.*

Proof. This is evident since $\mathcal{A}_k$ is a composition of homomorphisms. □

We can now compute the accumulated error of error history E_{hist} through a measurement circuit of a single-repetition measurement protocol $C_{\mathcal{S},1}$ by taking the product of the accumulation of each time-slice at the boundary:

Definition 3.3.12. The *accumulation map* $\mathcal{A} : \mathcal{P}_{n+m}^{\otimes(D+2)} \to \mathcal{P}_{n+m}$ maps error histories to their accumulated error:

$$\mathcal{A}(E_{hist}) = \prod_{k=1}^{D+2} \mathcal{A}_k(E_k) = E_{acc} \tag{3.3.9}$$

Lemma 3.3.13. *For a a clifford circuit, the accumulation map $\mathcal{A}$ is a group homomorphism of $\mathcal{P}_{n+m}^{\otimes(D+2)}$ to $\mathcal{P}_{n+m}$.*

Recall that the accumulated error breaks into data and ancilla factors: $E_{acc} = E_{acc,data} \otimes E_{acc,anc}$. The accumulation map as described above correctly calculates the accumulated error for an error history of a single-repetition protocol $C_{\mathcal{S},1}$. For an error history of a repeated measurement protocol $C_{\mathcal{S},r}$, we can calculate the accumulated error as follows:

1. Factor the error history E_{hist} into its r different sections (one for each measurement repetition):

$$E_{hist} = \bigotimes_{i=1}^{r} E_{hist,i} \tag{3.3.10}$$

2. For the first factor, $E_{hist,1}$ we calculate the accumulated error

$$\mathcal{A}(E_{hist,1}) = E_{acc,data,1} \otimes E_{acc,anc,1} \tag{3.3.11}$$

3. For factors $i = 2...r$: Interpret $E_{acc,data,i-1}$ as acting non-trivially on the very first data qubit fault locations of $E_{hist,i}$ and calculate the accumulated error

$$\mathcal{A}(E_{acc,data,i-1}E_{hist,i}) = E_{acc,data,i} \otimes E_{acc,anc,i} \tag{3.3.12}$$

4. The final accumulated error on the data qubits will be

$$E_{acc,data} = E_{acc,data,r} \in \mathcal{P}_n \tag{3.3.13}$$

   and the final accumulated error on the ancilla qubits will be

$$E_{acc,anc} = \bigotimes_{i=1}^{r} E_{acc,anc,i} \in \mathcal{P}_{rm} \tag{3.3.14}$$

Definition 3.3.14. The repeated accumulation map, $\mathcal{A}^r$, calculates the accumulated error of an error history E_{hist} of a repeated measurement protocol $\mathcal{M}_{\mathcal{S},r}$ using the above algorithm.

$$\mathcal{A}^r(E_{hist}) = E_{acc} = E_{acc,data} \otimes E_{acc,anc} \tag{3.3.15}$$

We think of $\mathcal{A}^r$ as pushing an error history of a circuit to an error of the extended stabilizer code on its boundary. Figure 3.5 gives a fully worked example of how we can use the extended stabilizer code and the definitions given in this section to interpret the output of the accumulation map acting on an error history (using CX propagation rules illustrated in figure 3.4).

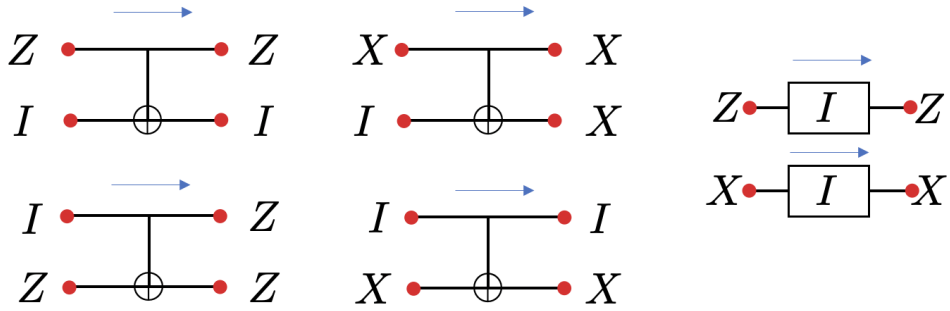


Figure 3.4: Gate propagation relations for CX and identity gates. These relations depict how individual gates of measurement circuits spread and propagate errors through the circuit. Each relation also tells us how to build an error history that has no effect on the data qubits or ancilla measurement outcomes: consider each relation as an error history where the errors before the gate occur, *and* the errors after the gate also occur at the next time step. Then the error pattern before the gate will propagate through the gate to become exactly the same as the error pattern that occurs after the gate, and will cancel it out.

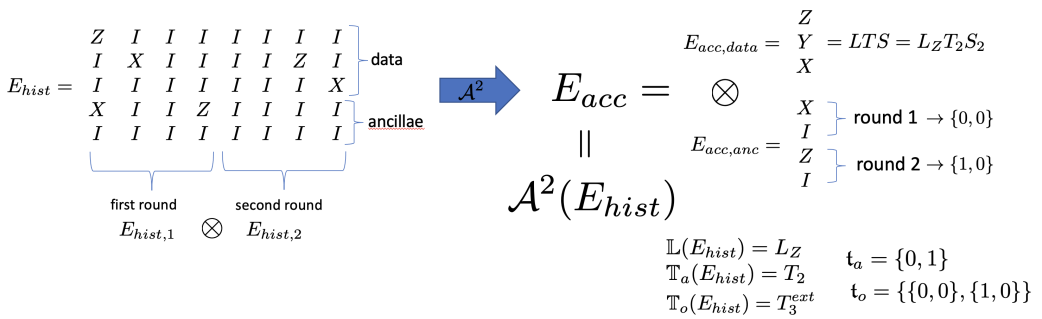


Figure 3.5: An example depicting how an error history for $C_{\mathcal{S},2}$ can be analyzed to understand what syndrome will be observed, and what actual error will have accumulated on the data qubits. The error history E_{hist} can be broken into the first and second measurement rounds. The accumulation map $\mathcal{A}^2$ is used to map E_{hist} to the equivalent accumulated error on the boundary of the circuit, E_{acc}. The accumulated error factors into components on the boundaries of the data and ancilla qubits respectively. The accumulated error on the data qubits is $E_{acc} = ZYX$. Referring to our chosen generators of the stabilizer, pure error, and logical groups of the base code, one can see that the LTS decomposition of this accumulated error is $L_Z T_2 S_2$. This means that if this accumulated error on the data qubits were subjected to a perfect (noise free) syndrome measurement we would obtain the *actual* syndrome, $t_a = \{0,1\}$. The observed syndrome, however, is a different story. The accumulated error on the ancilla qubits has two components, one for each round of syndrome measurements performed. The accumulated error for the first round is $E_{acc,anc,1} = XI$, which commutes with the $X \otimes X$ measurement being performed, resulting in observed syndrome $t_{o,1} = \{0,0\}$. The accumulated error for the second round is $E_{acc,anc,2} = ZI$, which anti-commutes with $X \otimes X$ on the first qubit, resulting in observed syndrome $t_{o,2} = \{1,0\}$.

We may now finally classify circuit error histories according to the LTS of the equivalent error of the extended stabilizer code:

$$\mathcal{E}_{L,t_a,t_o} = \{E_{hist} \in \mathcal{P}_{n+r}^{\otimes K} | \mathbb{L}(E_{hist}) = L, \mathfrak{t}_a(E_{hist}) = t_a, \mathfrak{t}_o(E_{hist}) = t_o\}. \quad (3.3.16)$$

Given a circuit-level error model π_{CL} for measurement circuit $mathcalC_{\mathcal{S},r}$, we can express the probability that an error history which occurs during the execution of $C_{\mathcal{S},r}$ produces a specific L and T_{t_a} on the data via $E_{acc,data}$, and produces a specific observed syndrome t_o on the ancilla qubits via $E_{acc,anc}$:

$$\pi_{CL}(\mathcal{E}_{L,t_a,t_o}) = \sum_{E_{hist} \in \mathcal{E}_{L,t_a,t_o}} \pi_{CL}(E_{hist}). \quad (3.3.17)$$

3.4 Circuit History Code by Example

Before describing the circuit history code construction in general detail, we will describe simplest possible example of the Circuit History Code (CHC) based on the 3-qubit phase-flip repetition code with $r = 1$. Recall, the 3-qubit phase-flip repetition code has a stabilizer with two generators:

$$\mathcal{S} = \langle M_1 = XXI, M_2 = IXX \rangle. \quad (3.4.1)$$

There are two associated pure errors

$$\mathcal{T} = \langle T_1 = ZII, T_2 = IIZ \rangle, \tag{3.4.2}$$

and we may choose logical operators

$$\mathcal{L} = \langle L_Z = ZZZ, L_X = IXI \rangle. \tag{3.4.3}$$

Consider the following simplified measurement circuit for this code in figure 3.6, where we omit the final time step to reduce the size of the associated groups:

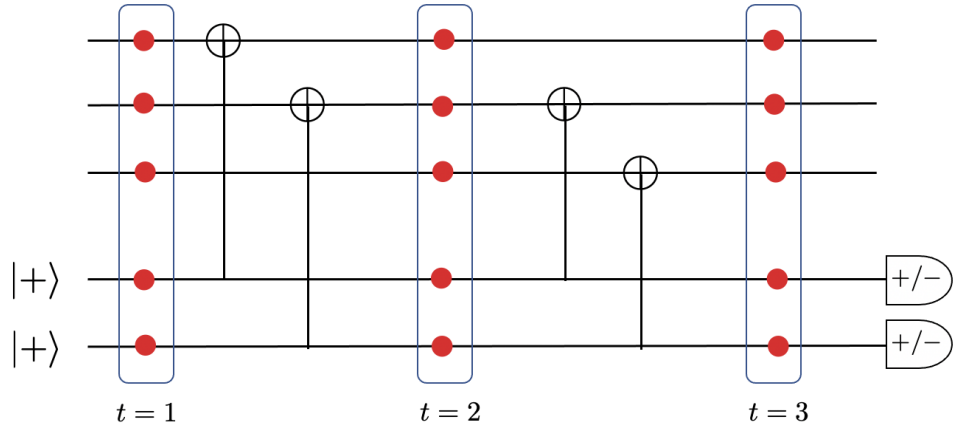


Figure 3.6: Standard *simplified* measurement circuit for the three bit phase-flip repetition code, $r = 1$. We omit the fourth time step of idle and measurement failure locations for simplicity in describing the construction in this example. The first three qubits are the data qubits of the repetition code. The other two are ancilla qubits used to measure the stabilizer generators $M_1 = XXI$ and $M_2 = IXX$ respectively. Ancilla qubits are prepared in the $|+\rangle$ state and measured in the X basis. Red dots denote fault locations where a single Pauli error may occur during the execution of the circuit. Each of these fault locations are viewed as physical qubits of the CHC.

The first three qubits are the physical data qubits of the code, and the other two qubits are meausrement ancilla qubits initialized in the $|+\rangle$ state and measured in the X basis after the final timestep. We can divide the circuit into three time-slices, each slice containing a possible fault location for each qubit.

The circuit history code assigns a physical qubit to each fault location of the base code's measurement circuit. So, in this example, the CHC is defined on $N = 15$ physical qubits.

The guiding principle behind the design of the CHC is that it treats dynamic circuit-level error histories of its base code as the static instantaneous errors of a code-capacity error model on the corresponding CHC, a gauge code. This is useful

because it allows us to understand the underlying algebraic group structure of the set of dynamic error histories, which we shall see allows us to better understand properties the circuit-level noise model and the fault-tolerant properties of a code's measurement protocols.

Below, we give a list of generators for the gauge group of the CHC. We note three different types of generators.

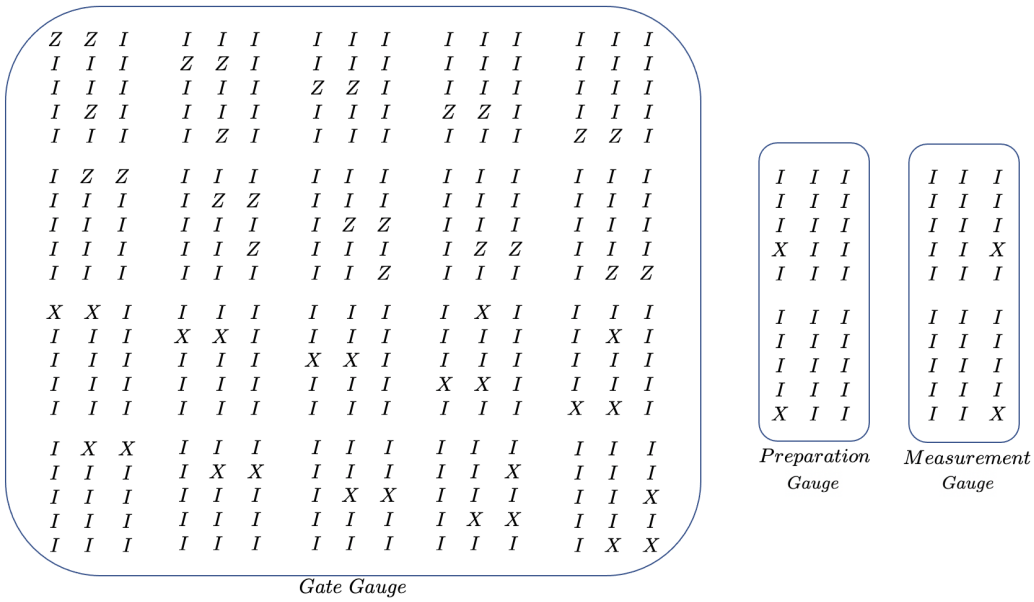


Figure 3.7: 24 independent generators of the gauge group of the CHC for the measurement circuit in figure 3.6. Gate gauge generators are determined using the error propagation relations of CX and identity gates seen in figure 3.4, below. Preparation gauge generators express that an X error doesn't affect a $|+\rangle$ state preparation. Measurement gauge generators express that an X error won't affect the outcome of an X-basis measurement.

The first are gate-type generators. These are constructed using the knowledge of how errors propagate through CX and identity gates, as seen in figure 3.4, above.

These obviously correspond to error histories which have a trivial effect on the accumulated error on the data qubits. This is equivalent to the fact that the image of each gate-type generator under the accumulation map $\mathcal{A}$ is the identity.

The second type of gauge generators are measurement-type generators. These are X errors that occur on an ancilla qubit just before performing an X-basis measurement of the ancilla. Obviously, such an error has no effect on the data qubits nor the observed measurement outcome.

The third type of gauge generators are preparation-type generators. These are X errors that occur on an ancilla qubit just after being initialized in the $|+\rangle$ state. Such an error will have no effect on the measurement outcome of the ancilla qubit. We can see, however, that propagating this error forward in time through the circuit has the

effect of applying a stabilizer generator to the data qubits (in particular, the stabilizer generator that this ancilla qubit is meant to measure).

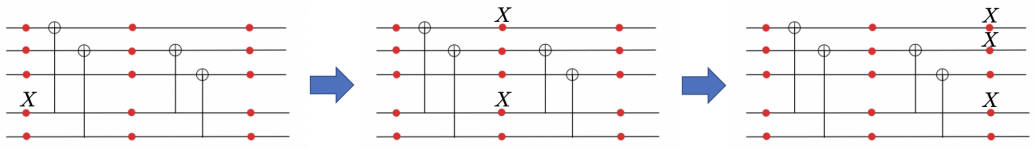


Figure 3.8: A single X error occuring on a freshly initialized $|+\rangle$ ancilla state propagates to apply a stabilizer generator to the data qubits.

All gauge operators share the common feature that, when considered as error histories of the measurement circuit, they do not change the measurement outcomes of any ancilla qubit measurements, and leave the accumulated error on the data qubits unchanged, up to a stabilizer operator of the base repetition code. The gauge group is exactly the subgroup of error histories that share this feature.

The Stabilizer group $\mathcal{S}'$ of the CHC is the center of the gauge group: $\mathcal{S}' = \mathcal{Z}(\mathcal{G}) \subseteq \mathcal{G}$. In this particular case, the stabilizer group can be generated using the operators below. We build these operators by taking the *worldsheets* of the base code stabilizer operators as they propagate through the circuit, as well as the worldsheets of a trivial initialization error on each ancilla qubit.

$$S'_1 = \begin{matrix} X & X & X \\ X & X & X \\ I & I & I \\ I & I & I \\ I & I & I \end{matrix}, \quad S'_2 = \begin{matrix} I & I & I \\ X & X & X \\ X & X & X \\ I & I & I \\ I & I & I \end{matrix}, \quad S'_3 = \begin{matrix} I & X & X \\ I & I & X \\ I & I & I \\ X & X & X \\ I & I & I \end{matrix}, \quad S'_4 = \begin{matrix} I & I & I \\ I & X & X \\ I & I & X \\ I & I & I \\ X & X & X \end{matrix} \tag{3.4.4}$$

One can check that these four operators commute with each of the 24 gauge generators in figure 7, and therefore form at least a subgroup of the center of the gauge group.

Next, we may describe generators for the group of pure errors of the CHC, $\mathcal{T}'$, using generators which canonically anti-commute with the above stabilizer generators. The non-trivial elements of $\mathcal{T}'$ are error histories which will accumulate non-trivial pure errors of the base code on the data qubits, as well as generate measurement outcome flips on the ancilla qubits. A generating set for these pure errors is given below.

$$T'_1 = \begin{matrix} I & I & Z \\ I & I & I \\ I & I & I \\ I & I & Z \\ I & I & I \end{matrix}, \quad T'_2 = \begin{matrix} I & I & I \\ I & I & I \\ I & I & Z \\ I & I & I \\ I & I & Z \end{matrix}, \quad T'_3 = \begin{matrix} I & I & I \\ I & I & I \\ I & I & I \\ I & I & Z \\ I & I & I \end{matrix}, \quad T'_4 = \begin{matrix} I & I & I \\ I & I & I \\ I & I & I \\ I & I & I \\ I & I & Z \end{matrix} \tag{3.4.5}$$

We can see that this set of generators canonically anti-commutes with the set of stabilizer generators. The intuition behind this choice of pure error generators is as follows: the first two are pure errors of the base code occurring on the boundary data qubits, along with a measurement outcome flip on the boundary of the corresponding ancilla qubit. These pure errors can be interpreted as generating the pure errors of the base code whose presence on the data qubits is also correctly detected in the measurement outcome on the ancilla. The other two pure errors are simply measurement bit flip errors on the individual ancilla boundary qubits, which are to be interpreted as generating measurement errors.

Finally, we may describe a set of logical operators for the CHC. These are error histories whose accumulation leave a logical operator of the base code on the data qubits. They can simply be chosen to be the logical operators of the base code placed on the boundary data qubits:

$$L'_Z = \begin{matrix} I & I & Z \\ I & I & Z \\ I & I & Z \\ I & I & I \\ I & I & I \end{matrix}, \quad L'_X = \begin{matrix} I & I & I \\ I & I & X \\ I & I & I \\ I & I & I \\ I & I & I \end{matrix}. \tag{3.4.6}$$

These chosen logical operators obviously commute with all stabilizers and pure errors and anti-commute with each other.

We have now provided 30 independent Pauli operators, which form a complete generating set for the Pauli group on 15 qubits, which is the number of qubits of the CHC in this example. This tells us that the error histories in the gauge group are the only ones which have a trivial effect on the data qubits and ancilla measurement outcomes, since any nontrivial combination of of the pure error or logical operator generators must have a non-trivial effect on the accumulated error.

Let's use the CHC take a closer look at the algebraic structure of the error histories of the underlying circuit-level error model.

As previously mentioned, the gauge group $\mathcal{G}$ consists of all possible error histories which have trivial logical and pure error accumulation on the data qubits, and cause no measurement errors on the data qubits.

All error-histories that are not in $\mathcal{G}$ introduce at least one of: an accumulated pure and/or logical error on the data qubits, or a measurement error.

We can use non-trivial elements of the CHC pure error and logical groups, $\mathcal{T}'$ and $\mathcal{L}'$, to compute exactly which sets of error histories cause which errors in our data and measurements: they will be co-sets of the gauge group in the Pauli group $\mathcal{P}_{15}$.

For example: to find *every* error history which causes a fixed accumulated logical error L and pure error T_{t_a} on the data qubits, and results in an observed syndrome t_o, one need only compute the coset

$$\mathcal{E}_{L,t_a,t_o} = LT'_{t_a}T'_{t_o}\mathcal{G}. \tag{3.4.7}$$

3.5 The Circuit History Code

In this section we give a formal description of the construction of the Circuit History Code (CHC). We will assume that we have a stabilizer code with stabilizers, pure errors, and logical operators $\mathcal{S}$, $\mathcal{T}$ and $\mathcal{L}$. Further, we will assume a standard repeated measurement protocol $\mathcal{M}_{\mathcal{S},r}$ with circuit $\mathcal{C}_{\mathcal{S},r}$.

We will describe a gauge/subsystem code defined on $N = K(n+m)$ physical qubits, the same as the number of fault locations in the measurement circuit $\mathcal{C}_{\mathcal{S},r}$. When we refer to the *base code* $\mathcal{S}$ and any of its n physical qubits, this will correspond to a set of K physical qubits of the CHC which model all fault locations for that base code data qubit. When we refer to an ancilla qubit, this will correspond to a set of K physical qubits of the CHC which model fault locations for the ancilla qubit being used to measure a particular stabilizer generator (including repetitions).

Boundary locations of the underlying measurement circuit distinguish corresponding *boundary qubits* of the CHC. Thus, a CHC of a standard repeated measurement circuit $\mathcal{C}_{\mathcal{S},r}$ will have $n + rm$ boundary qubits.

Error histories of the associated measurement circuit correspond to static errors of the physical qubits of the CHC (as one might encounter in a code capacity error model). However the algebraic structure of the dynamics of error histories we covered in section 3.3 will allow us to understand the algebraic structure of the CHC in a useful way.

Recall that we can interpret the repeated accumulation map $\mathcal{A}^r$ as 'pushing' a circuit error history E_{hist} to the boundary qubits. This map takes an error history and produces an accumulated error on the boundary qubits which allow us to perform an LTS decomposition to classify its effect. For notational convenience, we also define an *inclusion map* which turns an error from the extended base code into an error history which only has support on the boundary qubits of the CHC.

Definition 3.5.1. The inclusion map $\mathcal{I} : \mathcal{P}_{n+rm} \to \mathcal{P}_{n+m}^{\otimes K}$ maps a Pauli error on the boundary qubits of a CHC to the equivalent error on all CHC qubits (or error history of the measurement circuit) by adding identity operators to all other qubits.

The Gauge Group

At the core of the definition of any gauge/subsystem code is the gauge group. In the circuit history code, all gauge operators in $\mathcal{G}$ share the property that they result in a trivial accumulated logical error, pure error, and observed syndrome when

interpreted using the accumulation map $\mathcal{A}^r$. In section 3.4, we saw the intuition behind the three different types of generators of $\mathcal{G}$ that we now define for general circuit history codes.

The first type of generator is the gate gauge generator. As we discussed, these generators reflect that the effect of an X or Z error at any non-boundary fault location can be cancelled out by an appropriate error at the next time step, depending on the Clifford gate connecting those time steps. We may systematically construct all gate gauge generators with the following definitions:

Definition 3.5.2. Given a single qubit gate $U_{a\to b}$ whose input qubit of the CHC has label a and output qubit has label b, we can define two *gate gauge generators*,

$$G^{gate,E}_{a\to b} = (E)_a \otimes (U_{a\to b} \bullet E)_b,\ E \in \{X, Z\} \tag{3.5.1}$$

where identity operators on all other qubits of the CHC are implicit.

Definition 3.5.3. Given a two-qubit gate $U_{ab\to cd}$ whose input qubits of the CHC have labels a and b and output qubits have labels c and d, we can define four *gate gauge generators*,

$$G^{gate,E}_{ab\to cd} = (E)_{ab} \otimes (U_{ab\to cd} \bullet E)_{cd},\ E \in \{XI, IX, ZI, IZ\} \tag{3.5.2}$$

where identity operators on all other qubits of the CHC are implicit.

Lemma 3.5.4. *The image of any gate gauge generator under any accumulation map is the identity operator.*

Proof. By construction, we have that

$$\mathcal{A}_k((E)_{ab})(U_{ab\to cd} \bullet E)_{cd} = (U_{ab\to cd} \bullet E)_{cd}(U_{ab\to cd} \bullet E)_{cd} \tag{3.5.3}$$

$$= I, \tag{3.5.4}$$

where the final equality is due to the fact that $(U_{ab\to cd} \bullet E)_{cd})$ is a phaseless pauli operator. □

Next are the preparation gauge generators. Ancilla state preparation locations have a distinguished role as CHC qubits in that there is always an individual Pauli error which leaves a successfully prepared ancilla qubit unchanged. Therefore if such a Pauli error occurs directly after an ancilla preparation, it is a gauge operator of the CHC.

Definition 3.5.5. For the X or Z basis ancilla qubit preparations preceding the very first preparation location of a CHC qubit labeled a, we can define a *preparation gauge generator*,

$$G^{prep,E}_{a} = E_a \tag{3.5.5}$$

where E is X if a $|+\rangle$ state was prepared, or Z if a $|0\rangle$ state was prepared. Identity operators on all other qubits of the CHC are implicit.

Finally, we have measurement gauge generators. These can be understood in the same manner as the preparation gauge generators: a Pauli X/Z error occurring just before making an X/Z-basis measurement will have no effect on the outcome. Such an error is, therefore, a gauge generator.

Definition 3.5.6. For any X or Z basis measurement following CHC qubit labeled a, we can define a *measurement gauge generator*,

$$G_a^{meas,E} = E_a \tag{3.5.6}$$

where E is X or Z depending on the basis measured. Identity operators on all other qubits of the CHC are implicit.

With all proposed generators of $\mathcal{G}$ defined, we need to prove that they indeed are a minimal generating set for the set of error histories we've described: those whose image under $\mathcal{A}^r$ have trivial logical component, pure error, and and observed syndrome.

First, we note an obvious fact about the measurement gauge generators:

Lemma 3.5.7. *The image of each measurement gauge generator under the accumulation map is an extended stabilizer generator of the underlying extended stabilizer code:*

$$\mathcal{A}^r(\langle G_a^{meas,E}\rangle_a) = \langle M_i^{ext}\rangle_i. \tag{3.5.7}$$

Proof. This is obvious from the fact that $G_a^{meas,E}$ only act non-trivially on the boundary of the CHC. □

Next, we see that we can use specific combinations of measurement and preparation gauge generators to generate base stabilizer generators under $\mathcal{A}^r$:

Lemma 3.5.8. *The image of each preparation gauge generator together with its corresponding measurement gauge generator, $G_a^{prep,E}G_b^{meas,E}$ under the accumulation map $\mathcal{A}^r$ is simply the stabilizer generator that the corresponding ancilla qubit of the measurement circuit measures:*

$$\mathcal{A}^r(G_a^{prep,E}G_b^{meas,E}) = M_i \tag{3.5.8}$$

Proof. This is by construction of our measurement circuit $C_{\mathcal{S},r}$ as defined in definition 3.3.2. We will focus our attention on the case in which all ancilla qubits are prepared

and measured in the X basis. CX and CY gates propagate an X error on the control qubit (ancilla) to an X and Y respectively on the target qubit (data). A CZ gate propagates an X error on the target qubit (ancilla) to a Z on the control qubit (data). Therefore it is clear that $\mathcal{A}^r(G_a^{prep,E}) = M_i M_i^{ext}$. Therefore, we can see that $\mathcal{A}^r(G_a^{prep,E} G_b^{meas,E}) = M_i$, as inclusion of $G_b^{meas,E}$ in the error history will simply cancel out the accumulated M_i^{ext} on the boundary. □

Combining these facts, we have:

Lemma 3.5.9. *The image of the gauge group $\mathcal{G}$ under the accumulation map $\mathcal{A}^r$ is the extended stabilizer group of the base code, $\mathcal{S}^{ext}$:*

$$\mathcal{A}^r(\mathcal{G}) = \mathcal{S}^{ext}. \tag{3.5.9}$$

At this point, a simple counting argument suffices to prove that $\mathcal{G}$ is exactly the set of all non-trivial $E_{hist} \in \mathcal{P}_{n+rm}^{\otimes K}$ which map into $\mathcal{S}^{ext}$ under $\mathcal{A}^r$, justifying our definitions of the generators of $\mathcal{G}$:

Lemma 3.5.10. *For $E_{hist} \neq I$,*

$$\mathcal{A}^r(E_{hist}) \in \mathcal{S}^{ext} \iff E_{hist} \in \mathcal{G}. \tag{3.5.10}$$

Proof. We need only prove the forward direction. Note that there are $2N - 2m(r-1) - 2(n+m)$ gate gauge generators and $(r+1)m$ preparation and measurement gauge generators, making a total of $2N - (r-1)m - 2n$ independent gauge generators. Note that the logical group $\mathcal{L}$ and extended pure error group $\mathcal{T}^{ext}$ of the extended base code have exactly $2(n-m) + mr = 2n + (r-1)m$ independent generators. The images of these independent generators under the inclusion map $\mathcal{I}$ are obviously $2n + (r-1)m$ independent error histories which do not map to $\mathcal{S}^{ext}$ under $\mathcal{A}^r$ and are therefore not in $\mathcal{G}$. This means that no non-trivial error history outside of $\mathcal{G}$ can map to $\mathcal{S}^{ext}$ under the accumulation map. □

Definition 3.5.11. The *Gauge Group* of the CHC is the non-abelian group generated by all gate, preparation, and measurement gauge generators of the measurement protocol $\mathcal{M}_{\mathcal{S},r}$ associated to the CHC:

$$\mathcal{G} = \langle G_{a\to b}^{gate,E}, G_{ab\to cd}^{gate,EE'}, G_a^{prep,E}, G_a^{meas,E} \rangle. \tag{3.5.11}$$

CHC Gauge Pure Errors

Here we will describe the set of generators of the group of gauge pure errors $\mathcal{T}'$ of the CHC. We call them gauge pure errors because they do not correspond, strictly speaking, to pure errors of the gauge code, but rather to *dressed* pure errors which are equivalent to the true pure errors of the CHC up to gauge operators.

Definition 3.5.12. To every pure error generator $T_i^{(ext)}$ of the base code, we assign a gauge pure error $T_i^{(ext)\prime} = \mathcal{I}(T_i^{(ext)})$. These $(r+1)m$ independent generators generate the group of gauge pure errors of the CHC:

$$\mathcal{T}' = \langle T_i', T_i^{ext\prime} \rangle \tag{3.5.12}$$

That these generators are independent from those of $\mathcal{G}$ should be clear from the fact that they do not map to $\mathcal{S}^{ext}$ under $\mathcal{A}^r$. In fact:

Lemma 3.5.13. *The image of $\mathcal{T}'$ under the accumulation map $\mathcal{A}^r$ is the extended pure error group of the base code:*

$$\mathcal{A}^r(\mathcal{T}') = \mathcal{T}^{ext} \tag{3.5.13}$$

We will use the same notation for indication actual and observed syndromes on gauge pure errors of the CHC as we do with the extended base code.

CHC Gauge Logical Operators

Here we will describe the set of generators of the group of gauge logical operators $\mathcal{L}'$ of the CHC. We call them gauge logical operators because they do not correspond, strictly speaking, to logical operators of the gauge code, but rather to *dressed* logical operators which are equivalent to the true bare logical operators of the CHC up to gauge operators.

Definition 3.5.14. To every logical operator generator L_{X/Z_i} of the base code, we assign a gauge logical operator $L'_{X/Z_i} = \mathcal{I}(L_{X/Z_i})$. These $2(n-m)$ independent generators generate the group of gauge logical operators:

$$\mathcal{L}' = \langle L'_{X/Z_i} \rangle \tag{3.5.14}$$

As was the case with the gauge pure errors, it is obvious that these gauge logical operators are independent from any oeprators in $\mathcal{G}$ or $\mathcal{T}'$. And, in fact:

Lemma 3.5.15. *The image of $\mathcal{L}'$ under the accumulation map $\mathcal{A}^r$ is the logical group of the base code:*

$$\mathcal{A}^r(\mathcal{L}') = \mathcal{L} \tag{3.5.15}$$

The LTG Decomposition

We have now described $2N - (r-1)m - 2n$ independent gauge generators, $(r+1)m$ independent gauge pure error generators, and $2(n-m)$ independent gauge logical error generators, making for a total of $2N$ independent error histories which therefore generate the entirety of the group of error histories.

As a consequence, just as any error of the base code admits an LTS decomposition, any error history of the CHC admits an LTG decomposition:

Lemma 3.5.16. *Any error history $E_{hist} \in \mathcal{P}_{n+m}^{\otimes K}$ has a unique decomposition into a product of operators from $\mathcal{G}$, $\mathcal{T}'$, and $\mathcal{L}'$, and we call this decomposition the* LTG *decomposition of the error history:*

$$E_{hist} = LT_{t_a,t_o}G,\ L \in \mathcal{L}',\ T_{t_a,t_o} \in \mathcal{T}',\ G \in \mathcal{G} \tag{3.5.16}$$

As a result, we may group error histories that cause different combinations of syndromes and logical errors into different cosets of $\mathcal{G}$:

Lemma 3.5.17. *The set of error histories $\mathcal{E}_{L,t_a,t_o}$ with fixed logical component L, and fixed actual syndrome t_a and observed syndrome t_o are a conjugacy class of the Gauge group $\mathcal{G}$ in $\mathcal{P}_{n+m}^{\otimes K}$:*

$$\mathcal{E}_{L,t_a,t_o} = \{L'T'_{t_a,t_o}G | G \in \mathcal{G}\} \tag{3.5.17}$$

Therefore, assuming a circuit level noise model π_{CL}, we may organize the calculation of the occurrence of an error history with a specific syndrome pattern and accumulated logical component as a summation over the gauge group of the CHC:

Lemma 3.5.18. *The probability of an error history in conjugacy class $\mathcal{E}_{L,t_a,t_o}$ occurring is calculated by fixing L', T'_{t_a,t_o}, and summing over the gauge group:*

$$\pi_{CL}(\mathcal{E}_{L,t_a,t_o}) = \sum_{E_{hist} \in \mathcal{E}_{L,ta,to}} \pi_{CL}(E_{hist}) = \sum_{G \in \mathcal{G}} \pi_{CL}(L'T'_{t_a,t_o}G) \tag{3.5.18}$$

3.6 Maximum Likelihood Decoding with Tensor Networks

In this section, we apply the CHC to construct a tensor network implementation of maximum likelihood decoding for stabilizer codes under circuit level noise models. First we review the decoding problem in the settings of perfect and imperfect syndrome measurements. Next we describe and analyze the tensor network construction which can implement both exact and approximate maximum likelihood decoding.

Decoding Code Capacity Noise

The simplest definition of a decoder for stabilizer codes arises in the setting of a code capacity noise. In this setting, we have some stabilizer code $\mathcal{S}$ on n physical qubits and can then specify the decoding problem as follows:

- A pauli error E is drawn from from the noise model's probability distribution π_{CC}
- Each generator of the stabilizer group $\mathcal{S}$ is measured with perfect fidelity which produces the syndrome of error E, $\mathfrak{t}_a(E) = t_a \in \{0,1\}^m$, so that $\mathbb{T}_a(E) = T_{t_a}$

- The decoder, δ_{CC}, takes this syndrome t_a and uses it to determine a correction operator $C \in \mathcal{P}_n$

Generally, we can think of the code capacity decoder as a map

$$\delta_{CC} : \{0,1\}^m \to \mathcal{P}_n, \tag{3.6.1}$$

but there is one additional property it should satisfy: it should return the error state to the codespace. Algebraically, this condition can be stated

Definition 3.6.1. For an error $E \in \mathcal{P}_n$ with actual syndrome t_a, the code capacity decoder should produce correction $\delta(t_a)_{CC} = C$ such that

$$CE \in \mathcal{SL}, \tag{3.6.2}$$

or, equivalently, the correction C also has actual syndrome t_a and pure error component T_{t_a} so that CE has trivial actual syndrome and pure error component.

The above condition fixes the pure error component of $\delta_{CC}(t_a)$ to be exactly T_{t_a}. So the only non-trivial decision to be made in specifying the decoder map is in specifying the logical component of $\delta_{CC}(t_a)$.

The decoder δ_{CC} has successfully decoded syndrome t_a whenever the correction returns the error state back to the original code state. Algebraically, this occurs when the logical component of the correction $\delta_{CC}(t_a) = C$ matches the logical component of E, so that

$$CE \in \mathcal{S}. \tag{3.6.3}$$

In summary, it is trivial for a code capacity decoder to always return the state to the code space, and the question of success or failure depends entirely on whether or not it inadvertently applies an unwanted logical operator in doing so. As such, a code capacity decoder can be described as a map that takes the actual syndrome and simply produces a guess at the logical operator of the error:

$$\delta_{CC} : \{0,1\}^m \to \mathcal{L}. \tag{3.6.4}$$

Decoding With Faulty Measurements

The code capacity noise setting is highly idealistic. Although studying decoding under code capacity noise models can give impressions about how well a QECC will perform, we must consider more realistic noise models in order to understand more realistic performance and how performance should even be assessed.

One important deficiency of code capacity noise models is the assumption that stabilizer generators measurements can be carried out with perfect fidelity, and

therefore that the decoder has access to the *actual* syndrome of the data qubit error that it is trying to correct, t_a. Of course, in a real quantum computer, it is highly unlikely that there will be a way of performing perfect syndrome measurements, and we use phenomenological and circuit-level noise models to account for this.

The possibility of disagreement between observed and actual syndromes complicates the decoding process. It is no longer trivial to correct the pure error component of E and return the state to the codespace. Suppose, under a phenomenological noise model, that error E occurs which has actual syndrome t_a, and the syndrome t_o is measured, which may differ from t_a. If one naively believes that t_o was the correctly measured syndrome, and uses the code capacity decoder, then one would apply the correction

$$\delta_{CC}(t_o) = T_{t_o}\mathbb{L}(\delta_{CC}(t_o)). \tag{3.6.5}$$

The residual error will then be

$$\begin{aligned} CE &= (T_{t_o}\mathbb{L}(\delta_{CC}(t_o)))(T_{t_a}\mathbb{L}(E)) && (3.6.6)\\ &= T_{t_o}T_{t_a}\mathbb{L}(\delta_{CC}(t_o))\mathbb{L}(E) && (3.6.7)\\ &= T_{t_o\oplus t_a}\mathbb{L}(\delta_{CC}(t_o))\mathbb{L}(E), && (3.6.8) \end{aligned}$$

which has non-trivial pure error component unless $t_o = t_a$, which is only true if no syndrome measurement errors occurred. Therefore, naively using a code capacity decoder in the presence of measurement faults will mean that the state will not, in general, be returned to the codespace.

One way of mitigating the effects of faulty measurements is to repeat syndrome measurements several times. If the probability of measurement error q is sufficiently low, for example, taking a majority vote result of many repetitions for each stabilizer measurement to construct t_o can significantly increase the probability that $t_o = t_a$ and that the state will be returned to the codespace.

Correctly specifying and implementing repeated measurement schemes is a subtly technical problem in itself. One common model for specifying the decoding problem in a way which guarantees the decoder will return the state to the codespace is to allow for a single round of 'perfect' syndrome measurement after performing r rounds of imperfect measurements. This perfect round of syndrome measurement can be motivated in some experimental settings e.g. quantum memory experiments where the code state does not need to be fed forward into a continued quantum computation [37]. Therefore we may directly (destructively) measure each individual data qubit which gives enough information to reconstruct the actual syndrome.

Maximum Likelihood Decoding

In what follows, we focus on decoding in this scenario where the decoder has access both to r imperfect syndrome measurements, t_o as well as one final perfect syndrome

measurement t_a. In this setting the decoder is a map

$$\delta : \{0,1\}^{rm} \times \{0,1\}^{m} \to \mathcal{L}. \tag{3.6.9}$$

Because the input to the decoder includes the actual syndrome t_a, we know that it is trivial for the decoder to select the correct pure error T_{t_a} needed to return the state to the codespace, and therefore the only non-trivial decision for the decoder to make is what logical error it is trying to correct based on the syndrome information it has received.

In this setting if one knows the underlying error model producing the data qubit errors and syndromes, then there is an obvious strategy for maximizing the success of the decoding algorithm: given observed syndrome t_o and actual syndrome t_a, simply use the error model to determine which logical error matches the syndromes with the highest probability on average. This strategy is known as *maximum likelihood decoding*. Given a circuit level error model π_{CL} a maximum likelihood decoder δ_{ML} performs the following calculation:

$$\delta_{ML}(t_o, t_a) = \text{argmax}_{L' \in \mathcal{L}'} \sum_{E_{hist} \in \mathcal{E}_{L', t_a, t_o}} \pi_{CL}(E_{hist}). \tag{3.6.10}$$

Using the CHC's gauge group, we may re-write this as

$$\delta_{ML}(t_o, t_a) = \text{argmax}_{L' \in \mathcal{L}'} \sum_{G \in \mathcal{G}} \pi_{CL}(L' T'_{t_o} T'_{t_a} G). \tag{3.6.11}$$

In this setting, it is obvious by mathematical definition that the maximum likelihood decoder δ_{ML} is the *optimal* decoder: it will have the lowest possible rate of leaving a logical error on the data qubits after applying its correction.

Tensor Network Construction

Implementing a the maximum likelihood decoder δ_{ML} involves a complicated summation that crucially depends on the underlying mathematical structure of the stabilizer code and its measurement circuit. This structure is captured succinctly by the gauge group $\mathcal{G}$ of the associated CHC. We now describe how to reduce the ML decoding problem to the problem of contracting a tensor network by describing a construction which uses the structure of the CHC to produce a tensor network whose contractions implement the summation in equation (3.6.11).

The tensor network is built from two types of tensors: fault tensors and gauge tensors. We define fault tensors for single-qubit and two-qubit fault locations, but in principle these definitions extend in a straightforward way to allow for more highly correlated Pauli errors.

Definition 3.6.2. A single qubit fault tensor is defined using a probability distribution π over the single qubit Pauli group, and two sets of binary indices which we refer to as x-indices, $\{x_i\}$, and z-indices, $\{z_i\}$. We may then define the tensor:

$$\mathcal{F}^{z_1,z_2\cdots}_{x_1,x_2\ldots} = \pi(X^{\sum_i x_i} Z^{\sum_i z_i}). \tag{3.6.12}$$

Similarly, for a two-qubit fault location, we use a probability distribution π over the two-qubit Pauli group $\mathcal{P}_2$, two sets of x-indices, $\{x_i^1\}$ and $\{x_i^2\}$, and two sets of z-indices, $\{z_i^1\}$ and $\{z_i^2\}$. Then, we have

$$\mathcal{F}^{z_1^1,z_2^1\cdots z_1^2,z_2^2}_{x_1^1,x_2^1\ldots,x_1^2,x_2^2\ldots} = \pi(X_1^{\sum_i x_i^1} Z_1^{\sum_i z_i^1} \otimes X_2^{\sum_i x_i^2} Z_2^{\sum_i z_i^2}). \tag{3.6.13}$$

Figure 3.9 below shows graphical representations of single and two-qubit fault tensors.

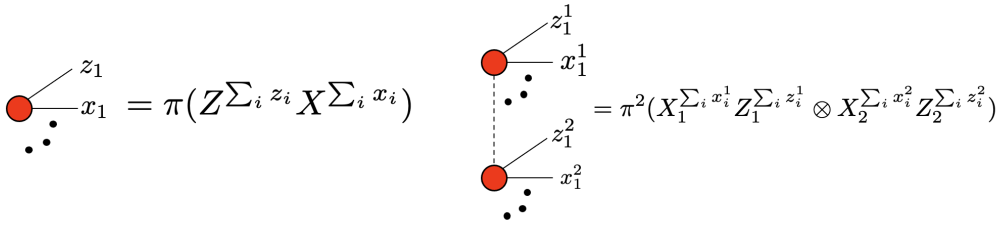


Figure 3.9: Single and two-qubit fault tensors. Observe that when two single fault locations are joined into a single fault tensor with a correlated probability distribution, it is denoted graphically using a dotted line.

Gauge tensors are modeled upon the gauge generators of the underlying CHC.

Definition 3.6.3. Given a gauge generator $g \in \mathcal{G}$, its corresponding gauge tensor is defined to be an identity tensor with exactly one binary x or z index for each location where g acts non-trivially with an X or Z operator, respectively. For this purpose, any location where g acts with a Y operator is understood to be a location where g acts with both an X operator *and* a Z operator (and has separate x and z indices corresponding to each).

Figure 3.10, below, gives an example of how to convert a simple gauge operator into a gauge tensor.

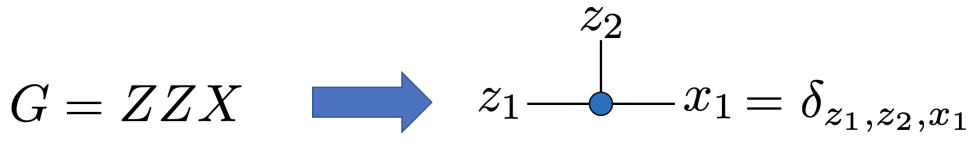


Figure 3.10: The gauge generator $G = ZZX$ is easily converted into its corresponding gauge tensor.

We now give an algorithm to build a maximum-likelihood decoder tensor network (MLTN). This algorithm takes as input a CHC with gauge generators $\mathcal{G}$ and set of fault locations F with corresponding probability distributions over the Pauli group.

1. For each gauge generator G, create a corresponding gauge tensor g according to definition 3.6.3 above.
2. For each probability distrubition π_i of the circuit-level noise model, create a single-qubit or two-qubit fault tensor $\mathcal{F}$ for the associated fault locations $f \in F$. The number of x-indices and z-indices of each fault-tensor is determined by the number of x and z-indices of gauge generators in step 1 which are associated to the underlying fault locations of $\mathcal{F}$.
3. Connect the x/z-indices of the gauge tensors g of step 1 with the appropriate x/z-indices of the fault tensors of step 2

The output of this algorithm is a tensor network $\mathcal{N}_G$ with no free indices, and therefore contracts to a scalar. In particular, we have that upon contraction,

$$\mathcal{N}_{\mathcal{G}} = \sum_{G \in \mathcal{G}} \pi_{CL}(G). \tag{3.6.14}$$

This tensor network therefore computes the probability of the identity coset of $\mathcal{G}$ in $\mathcal{P}_{n+m}^{\otimes K}$.

In figure 3.11 we show how to construct $\mathcal{N}_{\mathcal{G}}$ for a simple circuit consisting of a single CX gate (only Z type gate generators used for simplicity).

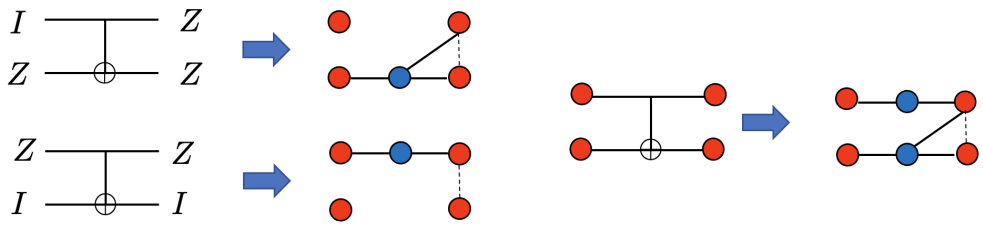
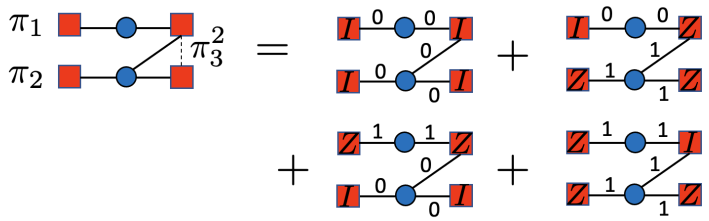


Figure 3.11: Here we show how to construct $\mathcal{N}_{\mathcal{G}}$ for a simple circuit consisting of a single CX gate, assuming for simplicity that there are only Z-type gate gauge generators. On the left we see how to connect the gauge tensors to the fault tensors for both Z-type gate gauge generators. On the right we see how to construct the full tensor from both generators. Here the dashed line between the fault tensors after the CX gate indicates that we are using a single joint distribution over Pauli errors on those locations in order to account for correlated errors.

In figure 3.12 we demonstrate how contracting the example of $\mathcal{N}_{\mathcal{G}}$ from figure 3.11 indeed sums over all four combinations of the two different gauge generators and their associated probabilities furnished by the circuit level error model.



$$= \pi_1(I)\pi_2(I)\pi_3^2(II) + \pi_1(I)\pi_2(Z)\pi_3^2(ZZ)$$
$$+\pi_1(Z)\pi_2(I)\pi_3^2(ZI) + \pi_1(Z)\pi_2(Z)\pi_3^2(IZ)$$

Figure 3.12: Here we show how the contraction of the example of $\mathcal{N}_{\mathcal{G}}$ given in figure 3.11.

We now describe how to turn $\mathcal{N}_{\mathcal{G}}$ into a tensor network that allows us to calculate the probability of any coset of $\mathcal{G}$:

Definition 3.6.4. The *circuit level tensor network* or CLTN $\mathcal{N}$ of a circuit level error model π_{CL} is built by taking the gauge tensor network $\mathcal{N}_{\mathcal{G}}$, and augmenting it as follows

1. Identify all fault tensors $\mathcal{F}$ associated with boundary locations of the circuit
2. To each fault tensor, add a single open x index and a single open z index

The open indices on each boundary fault tensor allow one to insert any pauli error whose support is entirely on the boundary. If one sets these indices according to boundary error $E \in \mathcal{P}_{n+rm}$, then contracting $\mathcal{N}$ gives

$$\mathcal{N}(E) = \sum_{G \in \mathcal{G}} \pi_{CL}(EG), \tag{3.6.15}$$

the probability of an an error history in the coset $E\mathcal{G}$. Therefore, by restricting the inputs to the boundary indices to be combinations of operators from $\mathcal{L}'$ and $\mathcal{T}'$ and considering the setting of the boundary indices of $\mathcal{N}$ as inputs to a function $\mathcal{N}$, we see that contracting $\mathcal{N}$ gives us a function

$$\mathcal{N} : \mathcal{T}' \times \mathcal{L}' \to [0, 1] \tag{3.6.16}$$
$$\mathcal{N}(T'_{t_o,t_a}, L') = \sum_{G \in \mathcal{G}} \pi_{CL}(GL'T'_{t_o}T'_{t_a}) \tag{3.6.17}$$

We illustrate the construction of $\mathcal{N}$ (restricting to a model with Z errors only for simplicity) for a 2-bit repetition code with a single stabilizer in figure 3.13, below:

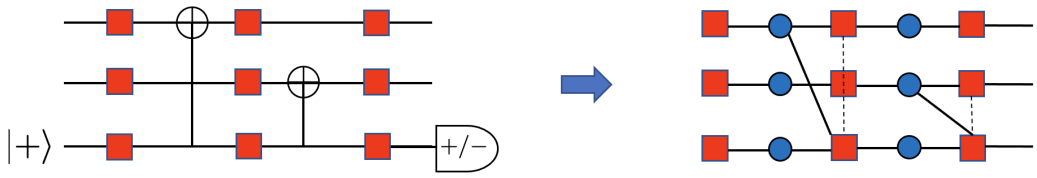


Figure 3.13: Here we show how the construction of $\mathcal{N}$ (restricting to a model with Z errors only for simplicity) for a 2-bit repetition code with a single stabilizer.

Therefore we have given a construction for a tensor network whose contractions allow us to implement maximum-likelihood decoding by fixing the observed and actual syndromes and repeating the contraction for all $L \in \mathcal{L}$, outputting the logical operator L' that which maximizes the value of $\mathcal{N}(T'_{t_o,t_a}, L')$.

Exact and Approximate Contraction Cost

Optimal exact contraction of a tensor network is generally a difficult problem, as determining the optimal contraction path is itself an NP-hard problem [29]. We can say, however, that the cost of contracting $\mathcal{N}$ will generally scale exponentially in the number of qubits that the measurement circuit acts on:

$$\mathcal{N} \in O(2^n). \tag{3.6.18}$$

This already indicates that exact maximum likelihood decoding using tensor network contraction does not scale in a practical way, and such a decoder is unsuitable for real-time decoding. However, the fact that we have reduced MLD to the problem of tensor network contraction allows us to use numerical techniques for controlling the complexity of contraction through use of *approximate* contraction. The idea is that the complexity of contraction is to a large degree controlled by the bond dimensions of the tensors being contracted. By using an appropriately chosen accuracy cutoff, one can reduce bond dimensions of all contractions to a fixed dimension in a way that minimizes loss in accuracy. This is usually accomplished using singular value decomposition.

By actively truncating to a fixed bond dimension χ throughout contraction, one can achieve a time complexity of $O(N\chi^3)$, where N is the number of tensors in the tensor network. In stark contrast to exact contraction, for a fixed number of repetitions, this grows *linearly* in the size n of the underlying stabilizer code.

Previous work [22] and [31] has shown that such approximate approaches to MLD have been able to yield high accuracy for low fixed bond dimensions in code capacity settings. It remains to be seen to what degree approximate decoding with fixed bond dimension works in the circuit-level noise settings, where the tensor networks are considerably more complex.

We implemented approximate tensor network ML decoding using PastaQ [42] a numerical package built on top of ITensor [43] that uses matrix product state (MPS) based algorithms to simulate quantum circuit evolution. Although PastaQ is designed to approximately contract tensor networks made up of unitary quantum gates, its methods are equally compatible with tensors described by stochastic matrices, like those in our tensor networks. We briefly describe preliminary results in the next section.

3.7 Applications and Future Work

Here we briefly discuss simulation efforts currently in progress. As discussed in this chapter's introduction, we expect that efficient approximate ML decoding may be able to outperform efficient yet sub-optimal decoders that are currently promising candidates for certain experimental architectures, e.g. minimum-weight perfect matching in surface code architectures.

To this end we perform simulations to compare with the thin strip rotated surface code MWPM simulations performed for the biased-noise concatenated cat-qubit architecture considered in [28]. Using an identical Z-biased error model as that used in [28], we perform Monte Carlo simulations to estimate the probability of logical Z failure in a $d_x = 3$ rotated surface code for various values of d_z after performing d_z rounds of noisy measurements and exact ML decoding.

As seen in figures 3.14 and 3.15, even exact ML decoding only marginally outperforms MWPM in this setting, with a relative percentage improvement of up to roughly 25% at $d_z = 9$.

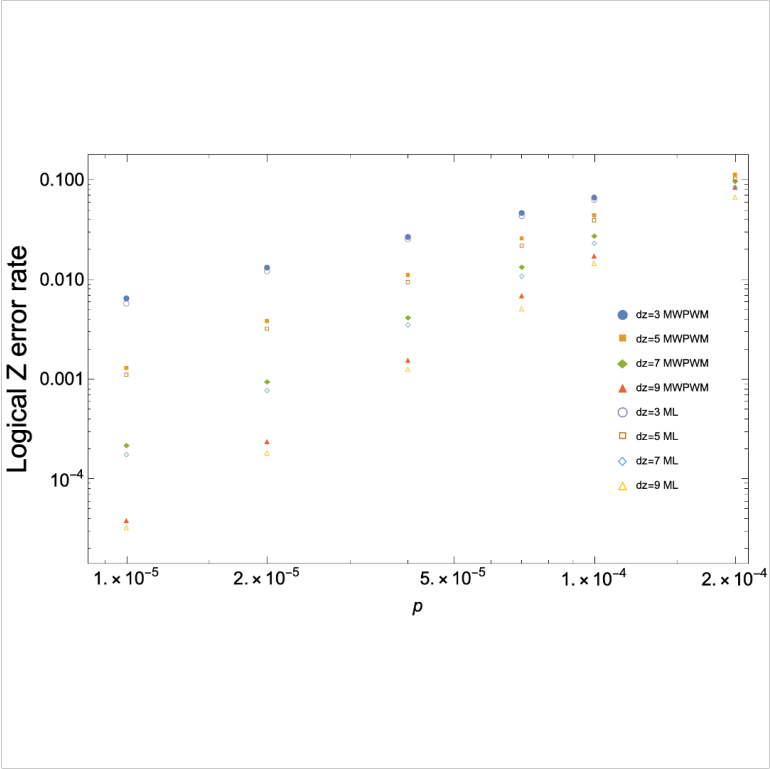


Figure 3.14: Total logical Z failure rate as a function of physical failure rate parameter p for thin strip rotated surface codes of width $d_x = 3$ and various lengths d_z. For each data point, the number of faulty measurement repetitions is the same as the length of the surface code patch, $r = d_z$. The error model used is described in the first column of Tables I and II of [28] (our parameter p is their κ_1/κ_2), and the MWPM data points are taken from Figure 8 of [28]. All data points collected via monte carlo simulation with standard error less than 5%. We note that this particular error model is very strongly Z-biased, with a bias of roughly 10^7.

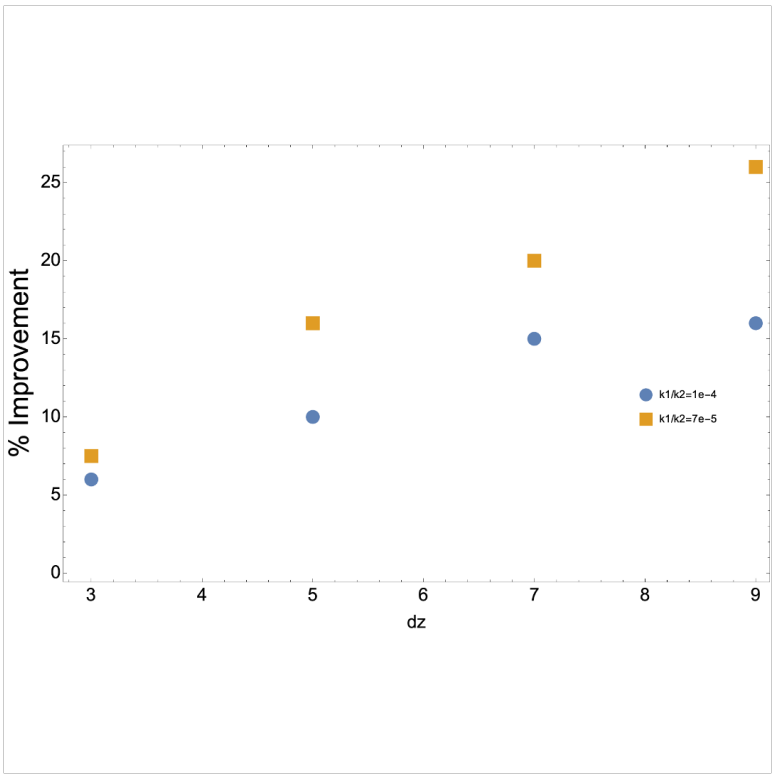


Figure 3.15: This figure shows the percent improvement in logical Z failure rate of ML decoding relative to MWPM decoding using the data from figure 3.14 for two different values of $p = \kappa_1/\kappa_2$.

If one is committed to using a specific type of decoder (e.g. MWPM), then this may also restrict the class of quantum codes that one can use. As an example, MWPM decoders take advantage of the CSS nature of the standard rotated surface code (whose stabilizer generator come in all X or all Z varieties) by matching X and Z defects separately. As a result, a MWPM decoder would have difficulty decoding in a situation where the independence of different types of defects is a poor assumption.

We can demonstrate one such example in the context of the biased noise cat-qubit architecture of [28]. This work studies fault-tolerance architectures based on physical qubits built with hardware whose noise model can be strongly biased to make Z errors far more likely than X or Y. This work uses a standard rotated surface code with X and Z type stabilizer generators, and takes advantage of the noise bias by only growing the d_z instead of both side lengths of the surface code patch. One could ask if using a different variant of the surface code whose stabilizer generators, themselves, are better suited to the biased noise might yield better results on the same hardware architecture.

One example of such a surface code variant is the XY rotated surface code, in which the Z-type stabilizer generators of the XZ surface code are replaced by Y-type generators. If the noise is dominated by Z errors, then only measurements of the X-type stabilizer generators of the XZ surface code are providing us with

any information about the occurrence of Z errors. One can imagine that at certain bias strengths, measuring the Z-type stabilizers, which only identify rare X and Y errors, may be doing more harm than good by adding more possible fault locations to the measurement circuit. By replacing the Z-type measurements with Y-type measurements, every observed syndrome bit is telling us about a possible Z error, which is what we need the most protection against. However, in order to actually use the XY code, we need a better suited decoder than the MWPM decoder as used in [28].

Using a realistic error model for the implementation of controlled-Y gates [55] in the cat-qubit architecture of [28], we simulate the performance of the rotated XY surface code with an approximate ML tensor network decoder of fixed bond dimension $\chi = 40$, and compare this performance to that of an XZ surface code with a MWPM decoder and identical hardware noise parameters.

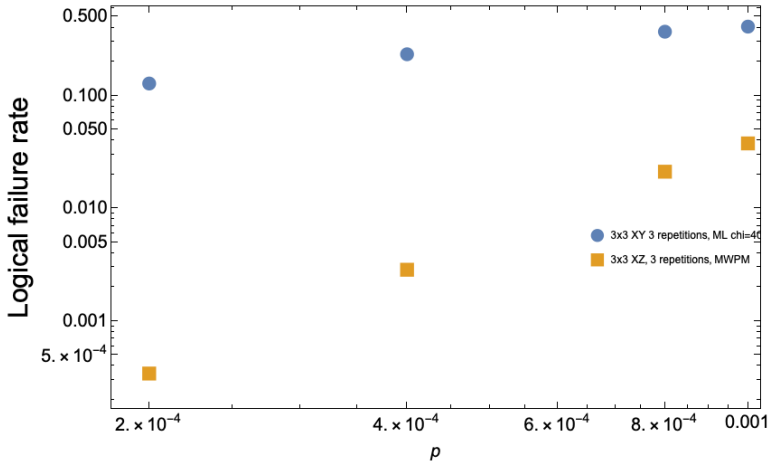


Figure 3.16: This figure compares the performance of two different 3×3 rotated surface codes (3 faulty measurement rounds) with different decoders. We see that using an XY code and approximate ML decoding, performance is improved by at least an order of magnitude compared to the standard XZ code using MWPM. These simulations assume nearly identical circuit level noise models. The XZ simulations use the same error model as in figure 3.14, and the XY simulations use that same noise model with added CY gate failure rates [55] compatible with the hardware parameters employed in this model.

We see that, as expected, there is a considerable improvement in logical failure rates in using an XY surface code over an XZ surface code in this setting. More

simulations could help us assess what kinds of physical overhead reductions using an *XY* code in this architecture could provide in experiments where a specific logical failure rate is being targeted.

While the limited simulation results we have presented are promising in making a case for studying approximate ML decoding with tensor networks, we have found that simulation code needs to be considerably more efficient in order to be able to realistically carry out informative simulations for larger system sizes. Although it's true that choosing a fixed bond dimension renders the scaling of the time complexity linear in the size of the tensor network, there is considerable overhead involved in performing the tensor network contractions, which make it difficult to probe larger code sizes without incurring massive simulation costs. As we grow the size of a surface code patch to even 5×3, we have contraction times on the order of 1 second per trial running on AWS cloud machines. It will be an ongoing challenge to further reduce the time cost per contraction in PastaQ in order to be able to use it more practically to perform Monte Carlo simulations.

BIBLIOGRAPHY

[1] We thank Elizabeth Crosson for suggesting this modification.

[2] Scott Aaronson. The complexity of quantum states and transformations: from quantum money to black holes. *eprint arXiv:quantph/1607.05256*, 2016.

[3] Scott Aaronson and Leonard Susskind. *in preparation*, 2017.

[4] Dorit Aharonov. *private communication*, 2017.

[5] Dorit Aharonov and Lior Eldar. Quantum locally testable codes. *SIAM Journal on Computing*, 44(5):1230–1262, 2015.

[6] Dorit Aharonov and Leo Zhou. On gap-simulation of Hamiltonians and the impossibility of quantum degree-reduction. *arXiv preprint arXiv:1804.11084*, 2018.

[7] Dorit Aharonov, Wim Van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, 50(4):755–787, 2008.

[8] Dorit Aharonov, Daniel Gottesman, Sandy Irani, and Julia Kempe. The power of quantum systems on a line. *Comm. Math. Physics*, 287(1):41–65, 2009.

[9] Dorit Aharonov, Itai Arad, and Thomas Vidick. Guest column: the quantum PCP conjecture. *ACM SIGACT news*, 44(2):47–79, 2013.

[10] Ahmed Almheiri, Xi Dong, and Daniel Harlow. Bulk locality and quantum error correction in AdS/CFT. *Journal of High Energy Physics*, 2015(4):163, 2015.

[11] Yosi Atia and Dorit Aharonov. Fast-forwarding of hamiltonians and exponentially precise measurements. *arXiv preprint arXiv:1610.09619*, 2016.

[12] Dave Bacon, Steven T Flammia, Aram W Harrow, and Jonathan Shi. Sparse quantum codes from quantum circuits. *IEEE Transactions on Information Theory*, 63(4):2464–2479, 2017.

[13] K. E. Batcher. Sorting networks and their applications. In *Proceedings of the April 30–May 2, 1968, Spring Joint Computer Conference*, AFIPS '68 (Spring), pages 307–314, New York, NY, USA, 1968. ACM. doi: 10.1145/1468075.1468121. URL http://doi.acm.org/10.1145/1468075.1468121.

[14] Johannes Bausch and Elizabeth Crosson. Analysis and limitations of modified circuit-to-Hamiltonian constructions. *Quantum*, 2:94, September 2018. ISSN 2521-327X. doi: 10.22331/q-2018-09-19-94. URL https://doi.org/10.22331/q-2018-09-19-94.

[15] Cédric Bény and Ognyan Oreshkov. General conditions for approximate quantum error correction and near-optimal recovery channels. *Physical review letters*, 104(12):120501, 2010.

[16] Dominic W Berry, Andrew M Childs, and Robin Kothari. Hamiltonian simulation with nearly optimal dependence on all parameters. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 792–809. IEEE, 2015.

[17] Fernando GSL Brandao, Aram W Harrow, and Michal Horodecki. Local random quantum circuits are approximate polynomial-designs. *arXiv preprint arXiv:1208.0692*, 2012.

[18] Fernando GSL Brandao, Elizabeth Crosson, M Burak Şahinoğlu, and John Bowen. Quantum error correcting codes in eigenstates of translation-invariant spin chains. *arXiv preprint arXiv:1710.04631*, 2017.

[19] Sergey Bravyi and Matthew B Hastings. Homological product codes. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 273–282. ACM, 2014.

[20] Sergey Bravyi and Barbara Terhal. A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes. *New Journal of Physics*, 11(4):043029, 2009.

[21] Sergey Bravyi, Matthew B Hastings, and Spyridon Michalakis. Topological quantum order: stability under local perturbations. *Journal of mathematical physics*, 51(9):093512, 2010.

[22] Sergey Bravyi, Martin Suchara, and Alexander Vargo. Efficient algorithms for maximum likelihood decoding in the surface code. *Phys. Rev. A*, 90:032326, Sep 2014. doi: 10.1103/PhysRevA.90.032326. URL https://link.aps.org/doi/10.1103/PhysRevA.90.032326.

[23] Nikolas P Breuckmann and Barbara M Terhal. Space-time circuit-to-Hamiltonian construction and its applications. *Journal of Physics A: Mathematical and Theoretical*, 47(19):195304, 2014.

[24] Winton Brown and Omar Fawzi. Short random circuits define good quantum error correcting codes. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 346–350. IEEE, 2013.

[25] Libor Caha, Zeph Landau, and Daniel Nagaj. Clocks in Feynman's computer and Kitaev's local Hamiltonian: Bias, gaps, idling, and pulse tuning. *Physical Review A*, 97(6):062306, 2018.

[26] A Robert Calderbank and Peter W Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54(2):1098, 1996.

[27] Sarah Cannon, David A. Levin, and Alexandre Stauffer. Polynomial Mixing of the Edge-Flip Markov Chain for Unbiased Dyadic Tilings. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017)*, volume 81 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 34:1–34:21, 2017. ISBN 978-3-95977-044-6. doi: 10.4230/LIPIcs.APPROX-RANDOM.2017.34. URL http://drops.dagstuhl.de/opus/volltexte/2017/7583.

[28] Christopher Chamberland, Kyungjoo Noh, Patricio Arrangoiz-Arriola, Earl T. Campbell, Connor T. Hann, Joseph Iverson, Harald Putterman, Thomas C. Bohdanowicz, Steven T. Flammia, Andrew Keller, Gil Refael, John Preskill, Liang Jiang, Amir H. Safavi-Naeini, Oskar Painter, and Fernando G. S. L. Brandão. Building a fault-tolerant quantum computer using concatenated cat codes, 2020.

[29] Lam Chi-Chung, P Sadayappan, and Rephael Wenger. On optimizing a class of multi-dimensional loops with reduction for parallel execution. *Parallel Processing Letters*, 7(2):157–168, 1997.

[30] Andrew M Childs, Debbie Leung, Laura Mančinska, and Maris Ozols. Characterization of universal two-qubit hamiltonians. *arXiv preprint arXiv:1004.1645*, 2010.

[31] Christopher T. Chubb. General tensor network decoding of 2d pauli codes, 2021.

[32] Claude Crépeau, Daniel Gottesman, and Adam Smith. Approximate quantum error-correcting codes and secret sharing schemes. In *Proceedings of the 24th annual international conference on Theory and Applications of Cryptographic Techniques*, pages 285–301. Springer-Verlag, 2005.

[33] Elizabeth Crosson and John Bowen. Quantum ground state isoperimetric inequalities for the energy spectrum of local Hamiltonians. *arXiv preprint arXiv:1703.10133*, 2017.

[34] Toby Cubitt, David Perez-Garcia, and Michael M. Wolf. Undecidability of the spectral gap (full version). *arXiv preprint arXiv:1502.04573*, 2015.

[35] Toby Cubitt, Ashley Montanaro, and Stephen Piddock. Universal quantum hamiltonians. *eprint arXiv:quantph/1701.05182*, 2017.

[36] C. M. Dawson and M. A. Nielsen. The Solovay-Kitaev algorithm. *eprint arXiv:quantph/0505030*, 2005.

[37] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505, 2002.

[38] David Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London A*, 4, 1985.

[39] Lior Eldar and Aram W Harrow. Local Hamiltonians whose ground states are hard to approximate. In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, pages 427–438. IEEE, 2017.

[40] Lior Eldar, Maris Ozols, and Kevin F Thompson. The need for structure in quantum LDPC codes. *arXiv preprint arXiv:1610.07478*, 2016.

[41] Richard Phillip Feynman. Simulating physics with computers. *Int. J. Theor. Phys.*, 21, 1982.

[42] Matthew Fishman and Giacomo Torlai. PastaQ: A package for simulation, tomography and analysis of quantum computers, 2020. URL https://github.com/GTorlai/PastaQ.jl/.

[43] Matthew Fishman, Steven R. White, and E. Miles Stoudenmire. The ITensor software library for tensor network calculations, 2020.

[44] Steven T Flammia, Jeongwan Haah, Michael J Kastoryano, and Isaac H Kim. Limits on the storage of quantum information in a volume of space. *Quantum*, 1:4, 2017.

[45] Michael H Freedman, David A Meyer, and Feng Luo. Z2-systolic freedom and quantum codes. *Mathematics of quantum computation, Chapman & Hall/CRC*, pages 287–320, 2002.

[46] Robert G. Gallager. Low-density parity-check codes, 1963.

[47] Anand Ganti and Rolando Somma. On the gap of Hamiltonians for the adiabatic simulation of quantum circuits. *International Journal of Quantum Information*, 11(07):1350063, 2013.

[48] Carlos E González-Guillén and Toby S Cubitt. History-state Hamiltonians are critical. *arXiv preprint arXiv:1810.06528*, 2018.

[49] David Gosset, Barbara M. Terhal, and Anna Vershynina. Universal adiabatic quantum computation via the space-time circuit-to-Hamiltonian construction. *Phys. Rev. Lett.*, 114:140501, Apr 2015. doi: 10.1103/PhysRevLett.114.140501. URL https://link.aps.org/doi/10.1103/PhysRevLett.114.140501.

[50] Daniel Gottesman. Fault-tolerant quantum computation with constant overhead. *arXiv preprint arXiv:1310.2984*, 2013.

[51] Daniel Gottesman and Sandy Irani. The quantum and classical complexity of translationally invariant tiling and hamiltonian problems. *arXiv preprint arXiv:0905.2419*, 2010.

[52] Mathew B Hastings. Weight reduction for quantum codes. *Quantum Information & Computation*, 17(15-16):1307–1334, 2017.

[53] Matthew B Hastings. Trivial low energy states for commuting Hamiltonians, and the quantum PCP conjecture. *Quantum Information & Computation*, 13 (5-6):393–429, 2013.

[54] Matthew B Hastings. Quantum codes from high-dimensional manifolds. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 67, 2017.

[55] Joseph Iverson. personal communication.

[56] Svante Janson, Dana Randall, and Joel Spencer. Random dyadic tilings of the unit square. 21:225–251, 10 2002.

[57] Leo P. Kadanoff. Scaling laws for ising models near tc. *Physics*, 2(6), 1966.

[58] A Yu Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003.

[59] Alexei Yu Kitaev. Quantum computations: algorithms and error corrections. *Russian Mathematical Surveys*, 52(6):1191–1249, 1997.

[60] Emanuel Knill and Raymond Laflamme. A theory of quantum error-correcting codes. 1996. doi: 10.1103/PhysRevLett.84.2525.

[61] Tohru Koma and Bruno Nachtergaele. The spectral gap of the ferromagnetic XXZ-chain. *Letters in Mathematical Physics*, 40(1):1–16, 1997.

[62] Jeffrey C Lagarias, Joel H Spencer, and Jade P Vinson. Counting dyadic equipartitions of the unit square. *Discrete mathematics*, 257(2-3):481–499, 2002.

[63] Yi-Chan Lee, Courtney G Brell, and Steven T Flammia. Topological quantum error correction in the Kitaev honeycomb model. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(8):083106, 2017.

[64] Debbie W Leung, Michael A Nielsen, Isaac L Chuang, and Yoshihisa Yamamoto. Approximate quantum error correction can lead to better codes. *Physical Review A*, 56(4):2567, 1997.

[65] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[66] Seth Lloyd and Barbara M Terhal. Adiabatic and Hamiltonian computing on a 2d lattice with simple two-qubit interactions. *New Journal of Physics*, 18(2): 023042, 2016.

[67] Seth Lloyd, Peter Shor, and Kevin Thompson. polylog-LDPC capacity achieving codes for the noisy quantum erasure channel. *arXiv preprint arXiv:1703.00382*, 2017.

[68] Neal Madras and Dana Randall. Markov chain decomposition for convergence rate analysis. *Ann. Appl. Probab.*, 12(2):581–606, 05 2002. doi: 10.1214/aoap/1026915617. URL https://doi.org/10.1214/aoap/1026915617.

[69] Juan Maldacena. The large-N limit of superconformal field theories and supergravity. *Adv. Theor. Math. Phys.*, 2, 1998.

[70] Chris Marriott and John Watrous. Quantum Arthur–Merlin games. *Computational Complexity*, 14(2):122–152, 2005.

[71] Ari Mizel, Daniel A Lidar, and Morgan Mitchell. Simple proof of equivalence between adiabatic quantum computation and the circuit model. *Physical review letters*, 99(7):070502, 2007.

[72] Daniel Nagaj and Pavel Wocjan. Hamiltonian quantum cellular automata in one dimension. *Phys. Rev. A*, 78(3), 2008.

[73] Chinmay Nirkhe, Umesh Vazirani, and Henry Yuen. Approximate Low-Weight Check Codes and Circuit Lower Bounds for Noisy Ground States. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 91:1–91:11, 2018. ISBN 978-3-95977-076-7. doi: 10.4230/LIPIcs.ICALP.2018.91. URL http://drops.dagstuhl.de/opus/volltexte/2018/9095.

[74] Andrew Steane. Multiple-particle interference and quantum error correction. *Proc. R. Soc. Lond. A*, 452(1954):2551–2577, 1996.

[75] Leonard Susskind. Computational complexity and black hole horizons. *eprint arXiv:hepth/1402.5674*, 2014.

[76] Jean-Pierre Tillich and Gilles Zémor. Quantum LDPC codes with positive rate and minimum distance proportional to the square root of the blocklength. *IEEE Transactions on Information Theory*, 60(2):1193–1202, 2014.

[77] K. G. H. Vollbrecht and J. I. Cirac. Quantum simulators, continuous-time automata, and translationally invariant systems. *Phys. Rev. Lett.*, 100(1), 2008.