

Composable Quantum Fault-Tolerance

Zhiyang He (Sunny)[†] Quynh T. Nguyen[‡] Christopher A. Pattison^{§¶}

August 15, 2025

Abstract

Proving threshold theorems for fault-tolerant quantum computation is a burdensome endeavor with many moving parts that come together in relatively formulaic but lengthy ways. It is difficult and rare to combine elements from multiple papers into a single formal threshold proof, due to the use of different measures of fault-tolerance. In this work, we introduce *composable fault-tolerance*, a framework that decouples the probabilistic analysis of the noise distribution from the combinatorial analysis of circuit correctness, and enables threshold proofs to compose independently analyzed gadgets easily and rigorously. Within this framework, we provide a library of standard and commonly used gadgets such as memory and logic implemented by constant-depth circuits for quantum low-density parity check codes and distillation. As sample applications, we explicitly write down a threshold proof for computation with surface code and re-derive the constant space-overhead fault-tolerant scheme of [Got14] using gadgets from this library. We expect that future fault-tolerance proofs may focus on the analysis of novel techniques while leaving the standard components to the composable fault-tolerance framework, with the formal proof following the intuitive “napkin math” exactly.

1 Introduction

Quantum fault-tolerance is arguably the central object of study in modern quantum error correction research. Informally, theorems for quantum fault-tolerance (hereafter “threshold theorems”) state that a circuit C can be mapped to a new circuit C_{FT} that produces an output distribution close to that of C even in the presence of some amount of noise.

Threshold theorems have a long history dating back to the mid-90s [Sho96; KLZ96; ABO97; Kit97] as early works justified the plausibility of large-scale fault-tolerant quantum computation. These formal proofs are still relevant today for a variety of reasons. Threshold theorems state how all the pieces (gadgets) of a particular fault-tolerance scheme will “fit together.” This allows users to be sure that no detail is missing from the scheme even if the analytically proven threshold is not

[†]Department of Mathematics, Massachusetts Institute of Technology. szhe@mit.edu.

[‡]Harvard University. qnguyen@g.harvard.edu.

[§]Simons Institute for the Theory of Computing, University of California, Berkeley. cpattison@berkeley.edu.

[¶]Institute for Quantum Information and Matter, California Institute of Technology

quantitatively precise. Such a proof along with numerical estimation of the threshold is the best evidence for fault-tolerance without simulating or running the full circuit experimentally. As we develop novel low-overhead gadgets for fault-tolerant computation, threshold theorems should be proven to illustrate how the new techniques change the full picture.

Unfortunately, proofs of threshold theorems remain a tedious and technical exercise where many details must be handled in relatively standard ways in order to wrap the novel portion of a construction. Worse yet, proofs of threshold theorems are largely *non-composable*: Fault-tolerance constructions are frequently written more or less monolithically with few formally written intermediate results suitable for reuse in other works.

Here, we introduce a framework for composable fault-tolerance where complicated fault-tolerant circuits and schemes may be assembled out of simple ones in a rigorous yet essentially black-box way. On a high-level, we decouple the probabilistic analysis of noise models from the combinatorial analysis of circuit correctness. This decoupling is enabled by the weight enumerator formalism (Section 2), which bounds the set of circuit locations corrupted by noise using the notion of *bad sets*. We briefly elaborate on this central mechanism of our formalism.

On the combinatorial side, consider a gadget (which is a circuit) with a set of corrupted locations and an input state with a set of corrupted qubits. We say that a circuit has *failed* if the set of corrupted locations includes a *bad fault path* and call a state *bad* if the set of corrupted qubits includes a *bad error support*.¹ Then, the correctness properties of a gadget can be easily specified combinatorially: A successful execution of the circuit on a good input state results in a good output state. This specification of the input/output behavior is at the heart of our composability – two gadgets with compatible definitions of bad error supports can now be analyzed independently and later composed to become a larger gadget with no regard to the probability distribution of noise.² The specification of bad fault paths and error supports for this larger gadget can be easily computed from the specifications of the individual gadgets in the weight enumerator formalism.

On the probabilistic side, for noise models such as locally stochastic noise, given the definition of bad sets from the gadgets, the probability of noise corrupting a set of locations that contain a bad set can be bounded by the weight enumerator polynomial. Moreover, computation of bad sets (as needed for composition of gadgets) corresponds straightforwardly to computation with polynomials. Existence of a threshold then follows straightforwardly if the polynomial decays exponentially in some parameter such as code distance.

Our main technical contribution is the rigorous formulation of these definitions and concepts, especially the combinatorial specification of gadgets. The weight enumerator formalism, as well as a precursor collection of combinatorial definitions, were introduced and utilized in [NP24] to prove threshold for a low-overhead fault-tolerant scheme. This work keeps the same weight enumerator algebra while re-designing and generalizing the core definitions, introducing a complete edition of the composable fault-tolerance framework. In particular, we introduce more precise definitions, tools for manipulations and correctness proofs of fault-tolerant circuits, as well as a library of examples.

We demonstrate the general applicability of our composable framework with instructive examples.

1. We build a library of standard and commonly used gadgets such as error correction in quan-

¹Bad fault paths and bad error supports are collections of sets defined with respect to the gadget (including a decoder) and generally “witness” some configuration of faults that may lead to failure of the gadget.

²In particular, whether the distribution of noise across the two gadgets is independent or not is irrelevant.

tum low-density parity check codes using d -rounds of syndrome extraction, logical gates implemented by constant depth circuits, and distillation. These gadgets are analyzed independently and combinatorially, formulated in terms of bad sets.

2. Utilizing gadgets from this library, we assemble a constant space-overhead quantum fault-tolerance scheme, providing an alternative proof of the results in [TKY24] and [Got14] up to differences in the classical computation model. In place of the concatenated codes fault-tolerant schemes used in [TKY24; Got14], we use a threshold theorem for surface codes built with magic state distillation and transversal gates. To our knowledge, this is the first explicit proof that surface code using magic state distillation has a constant noise threshold.³
3. To facilitate concatenated coding and fault-tolerant schemes, we prove a *level reduction* theorem similar to the one in [AGP05] where a fault tolerant circuit can be shown to noisily simulate another fault-tolerant circuit on the logical level. The bad sets and weight enumerator polynomials for such a concatenated circuit (equivalently gadget) are easily computable as the composition of the bad sets and polynomials of its component gadgets.

With the composable fault-tolerance framework, we expect that future threshold proofs of fault-tolerant schemes may focus on the combinatorial analysis of error propagation and correction in novel gadgets, while leaving the standard components and rigorous formality to the framework. For instance, while previously schemes may prove that a fault-tolerant circuit with locally stochastic noise on the input will produce the correct output subject to a (different) locally stochastic noise, future schemes may analyze such a gadget fully combinatorially with no regard to the probabilistic distribution of noise. Moreover, a novel fault-tolerant gadget which implements a particular logical operation can now be easily extended into a full fault-tolerant universal computation scheme by composing with standard gadgets. Such extensions are often invoked cursorily in the literature to support the utility of novel gadgets, without rigorous analysis of noise distribution. The composable fault-tolerance framework provides structure to fill in these rigorous details and supports new constructions with a unified formal foundation.

1.1 Reader’s guide

While this paper is lengthy, we have attempted to ensure that readers will benefit from reading only subsets of the contents. Footnotes are provided when subtle details of definitions are relevant or to explain generalizations. A nomenclature table is included at the end of the paper. On a light reading, we recommend readers read Section 2 (defining the bad sets, weight enumerator polynomials and their algebra) carefully while skipping Definition 2.7 until it becomes relevant. Section 3 may be skimmed. It formalizes a relatively intuitive notion of a quantum circuit in the presence of additional classical or quantum processing.⁴ We also formally define the notion of faults, which captures how noise may corrupt gates in a circuit.⁵ The tools in later part of the paper work with circuits subject to particular faults.

³We note that this is essentially a formalization of a proof sketch in [Den+02] using magic state distillation [BK05].

⁴For conceptual simplicity, we are not restricting the classical computation circuit. The definitions carry over to the case of a restricted depth classical computation per layer of quantum operations although the constructions given here usually do not carry over without modifications to the algorithms used for classical processing.

⁵Note that a fault is not a probabilistic distribution of noise (or a noise model per se), but rather an instantiation of a noise model corrupting a circuit.

Section 4 contains the core definitions of gadgets⁶ suitable for composability, facilitated by the formulation of bad error supports and fault paths. We state and prove the parallel and sequential composition of gadgets (Proposition 4.9), as well as the *level reduction* theorem (Theorem 4.11) that allows a fault-tolerant quantum circuit to simulate another one.⁷ We recommend that readers read and interpret the statements of definitions and lemmas on an intuitive level, leaving the detailed mathematical formalities and proofs to a deeper read.

In Section 5, we perform the probabilistic analysis which bounds the failure probability of fault-tolerant circuits under various noise models by the weight enumerator polynomials of the (combinatorial) bad sets, which are computed from the composition of gadgets. The analysis for locally stochastic noise is straightforward. We further study coherent noise, and present partial progress on obtaining a constant threshold bound for computation using qLDPC codes with sublinear distance such as surface codes. At the moment, we are able to show a threshold *for computation* that is vanishing polylogarithmically with the circuit volume.

Section 6 is where we begin building the library of standard gadgets with combinatorial analysis. We begin by formulating the bad error supports for a LDPC code, and then construct error correction gadgets and transversal gate gadgets analogous to those of [Got14] under the assumption of a minimum-weight decoder. We recommend that readers read the statements of the definitions and lemmas, and relate them to those in Section 4. The proofs involve some intuitive yet technically involved combinatorics, which would be relevant examples for readers who wish to use our framework to prove fault-tolerance for novel gadgets. They may be skipped on a light reading.

Section 7 is where we showcase the applicability of the composable fault-tolerance framework by assembling independently analyzed gadgets from Section 6 into threshold proofs for fault-tolerant schemes in a black-box fashion. The first threshold theorem is for surface codes using magic state distillation and transversal gates. Utilizing this fault-tolerant scheme, we provide an alternative proof of constant space overhead quantum fault-tolerance closely following [TKY24] and [Got14]. We note that [TKY24] counts the runtime of the classical computation which introduces additional requirements on their gadgets. [NP24] reduces the time overhead to essentially logarithmic in a similarly restricted model of classical computation. Proofs in Section 7 are split into construction and proof. We suggest that readers focus on the constructions on a first read. We further note that the constructions and proofs in Section 6 and Section 7 are asymptotic in nature, which means there was no attempt to optimize the constants. While such asymptotic results have found applications in complexity theory, it is our hope that asymptotic analysis of fault-tolerant should provide evidence and potential guidance to practical constructions of large-scale fault-tolerant quantum computers, much as the recent studies on theoretical and practical qLDPC codes have demonstrated.

Finally, we remark that the current framework and manuscript is the result of extensive revisions, as formulating a rigorous and applicable set of definitions necessarily encounters many subtleties. Seemingly obvious definitions may either be inaccurate or insufficient to prove important results such as gadget composition or level reduction. Consequently, some of the core definitions, such as those surrounding gadgets in Section 4, are quite technical and may not seem intuitive on first sight. We try our best to point out the relevant intuition and subtleties in discussions and footnotes, and

⁶By gadget, we mean a quantum circuit that may act on or output a quantum register that has been shown to perform some useful operation.

⁷This provides a proof of a claim made in [NP24] about the composability of certain fault-tolerance schemes (in particular, those with “friendly” gadgets such as used in concatenated code fault-tolerance).

kindly ask that readers contact us for any questions or comments.

1.2 Prior Works

[CFG24] provide an alternative formalism for quantum fault-tolerance based on channels and approximate simulation as opposed to the exact simulation and combinatorial analysis considered here. Many of the applications considered here have equivalent statements in [CFG24]. Our formalism is heavily inspired from [ABO97] and [Kit97]. It is possible to include the “extended rectangles” from [AGP05] however, we elected not to in order to simplify the presentation. The “level reduction” theorem (Theorem 4.11) is a vast generalization of the one proved in [AGP05].

[NP24] introduced an early version of the weight enumerator formalism in the minimal form required to prove the result. Our work greatly expands the framework in an approachable way and provides justification of a claim made about recursive simulation of fault-tolerance schemes required for amplification of the threshold to constant in [NP24].

1.3 Future Directions

We expect the composable fault-tolerance framework to serve as the formal foundation in a large number of future fault-tolerance results. We briefly discuss some potential directions here.

1. Constant coherent noise thresholds are known for concatenated constructions [ABO97; AGP05; CFG24], but can one prove a constant noise threshold against coherent noise without code concatenation? [CFG24] has shown that constant noise threshold is possible when using linear-distance qLDPC codes with a single-shot decoder, but it is reasonable to conjecture that the linear-distance requirement can be dropped. Prior works have analytically shown a $O(1/d)$ coherent noise threshold for distance- d surface codes in the memory setting [IP20], but numerical simulations suggest the threshold to be a constant [Bra+18].
2. In this work, we impose no constraints on the connectivity of the physical qubits and allow arbitrary blocks of qubits to interact. It would be useful to prove threshold theorems for computation in restricted models of connectivity using techniques such as qubit routing [GB25; PKP23] or lattice surgery [Hor+12]. We expect that the fault-tolerance of code surgery gadgets (see Remark 6.26) can be proved similarly to the qLDPC error correction gadget (Theorem 6.23).
3. Asymptotically, the spacetime overhead of quantum fault-tolerance has been brought down substantially against worst-case circuits. An obvious next step would be to customize fault-tolerance schemes to circuits of interest. Then, perhaps it is possible to construct practically relevant quantum fault-tolerance schemes with lower asymptotic overhead on particular circuits.
4. While only a handful of applications require extended rectangles (see [AGP05]) and our definitions are compatible with them, we have found it to be notationally burdensome to work with in full generality. Perhaps some improved notation could be developed.
5. It would be a useful resource to establish a threshold theorem for the surface code under minimal runtime assumptions for the classical decoder e.g. using a parallel window decoder [Sko+23; Tan+23] or the parallel decoder from [TY25].

6. We expect our techniques to also be useful for providing an end-to-end threshold proof of low time overhead quantum fault-tolerance using transversal gates in surface codes [Zho+24; SPST25; Cai+25].

1.4 Acknowledgments

The authors would like to thank Alfred, Thiago Bergamaschi, Ken Brown, Kathleen Chang, Nicolas Delfosse, Anirudh Krishna, Urmila Mahadev, John Preskill, Armands Strikis, and Eugene Tang for useful discussions and encouragement.

This work was done in part while the authors were visiting at the Quantum Algorithms, Complexity, and Fault-Tolerance workshop hosted at the Simons Institute for the Theory of Computing (supported by DOE QSA grant #FP00010905 and NSF QLCI Grant No. 2016245), the Fault-Tolerant Quantum Technologies workshop hosted at the Centro de Ciencias de Benasque Pedro Pascual (CCBPP), and the Institute for Quantum Information and Matter (IQIM) at Caltech. We thank the Simons Institute, the CCBPP, and Caltech IQIM for their hospitality.

Z.H. is supported by the MIT Department of Mathematics and the NSF Graduate Research Fellowship Program under Grant No. 2141064. QTN acknowledges support through the NSF Award No. 2238836 and IBM PhD fellowship. C.A.P. is currently a Simons-CIQC postdoctoral fellow at the Simons Institute for the Theory of Computing, supported by NSF QLCI Grant 2016245. C.A.P. acknowledges funding from the U.S. Department of Energy Office of Science, DE-SC0020290. The Institute for Quantum Information and Matter is an NSF Physics Frontiers Center.

Contents

1	Introduction	1
1.1	Reader's guide	3
1.2	Prior Works	5
1.3	Future Directions	5
1.4	Acknowledgments	6
2	Weight enumerator formalism	7
3	Circuit formalism	10
3.1	Networks	10
3.2	Types	12
4	Fault-tolerant gadgets	18
4.1	Gadgets	22

4.2	Decoupling of faults	25
4.3	Level reduction	26
4.4	Miscellaneous tools	29
5	Circuit correctness	33
5.1	Adversarial noise	34
5.2	Local stochastic noise	34
5.3	Coherent noise	35
6	Application: qLDPC codes and transversal gates	38
6.1	Correctable sets	38
6.2	Spacetime code	44
6.3	Error correction gadget	50
6.4	Constant depth gadgets	55
7	qLDPC Threshold Theorems	60
7.1	Surface codes	60
7.1.1	State injection	63
7.1.2	Magic state distillation	65
7.1.3	Assembly	69
7.2	Fault-tolerant quantum output	72
7.3	Constant space overhead fault-tolerance	74
	Nomenclature	80

2 Weight enumerator formalism

We now introduce some definitions that will allow us to work with sets of faulty locations in a composable way that we call the “weight enumerator formalism”. An earlier version of the weight enumerator formalism appeared in [NP24].

We begin by defining sets that we will want the support of errors (or faults) to avoid. Roughly, whenever the errors do not contain any of these sets, we will be able to conclude some property irrespective of what the error is. One (basic) example of this is for a length- n repetition code, all subsets of weight $\lfloor \frac{d-1}{2} \rfloor + 1$.

Definition 2.1 (Avoiding sets). For a set Ω and a family of subsets $\mathcal{F} \subseteq P(\Omega)$, referred to as **the bad sets**, a subset $X \subseteq \Omega$ is said to be \mathcal{F} -avoiding if it does not contain an element of \mathcal{F} . That is, $X \subseteq \Omega$ is \mathcal{F} -avoiding if and only if

$$\forall F \in \mathcal{F}, F \not\subseteq X. \quad (2.1)$$

Definition 2.2 (Set lower bound). For a set Ω and two families of subsets $\mathcal{F}_1, \mathcal{F}_2 \subseteq P(\Omega)$, \mathcal{F}_1 is said to be a **set lower bound** of \mathcal{F}_2 if for every $F_2 \in \mathcal{F}_2$, there exists $F_1 \in \mathcal{F}_1$ such that $F_1 \subseteq F_2$. We denote this relation as $\mathcal{F}_1 \preceq \mathcal{F}_2$.

As it turns out, it is frequently the case that the only data required about the family of sets (for an end-user) is how many sets of each weight there are. Thus, we can associated these families with polynomials that we refer to as *weight enumerator polynomials* in analogy to the weight enumerator polynomials used in coding theory.

Definition 2.3 (Weight enumerators). For a finite set Ω and a family of subsets $\mathcal{F} \subseteq P(\Omega)$, it will be convenient to associate a polynomial $\mathcal{W}(\mathcal{F}; x)$ to \mathcal{F} defined as the sum

$$\mathcal{W}(\mathcal{F}; x) = \sum_{w=0}^{\infty} A_w x^w, \quad (2.2)$$

where $A_w = |\{f \in \mathcal{F} \mid |f| = w\}|$ is the number of elements of \mathcal{F} of weight w .

Example 2.4 (Local stochastic noise). To see that this data is useful, suppose $x \in \mathbb{F}_2^n$ is a bit string with entries 1 with probability p and 0 with probability $1 - p$. Then, for any family $\mathcal{F} \subseteq P([n])$ the probability that x fails to be \mathcal{F} -avoiding is upper bounded by the evaluation of the corresponding weight enumerator:

$$\Pr(x \text{ is not } \mathcal{F}\text{-avoiding}) \leq \sum_{f \in \mathcal{F}} \Pr(f \subseteq x) \leq \mathcal{W}(\mathcal{F}; p) \quad (2.3)$$

More generally, this bound holds whenever x is distributed according to a so-called *local stochastic* distributions. Local stochastic distributions obey the following property.

$$\text{for all } S \subseteq [n] \quad \Pr(S \subseteq x) \leq p^{|S|} \quad (2.4)$$

Once one associates polynomials to these families of bad sets, it is natural to ask whether the polynomial ring operations have any operational meaning. For many noise distributions, the error support is not independent on disjoint sets. However, we can define certain sum and product operations that correspond to union-bounds and independence in the independent-noise case.

Definition 2.5 (Ring of bad sets). For a set Ω with the decomposition $\Omega_1 \cup \Omega_2 = \Omega$ and families of subsets $\mathcal{F}_1 \subseteq P(\Omega_1)$ and $\mathcal{F}_2 \subseteq P(\Omega_2)$, we define two operations between \mathcal{F}_1 and \mathcal{F}_2 to arrive at a subset $\mathcal{F} \subseteq \Omega$. Let ι_1 (ι_2) be the canonical inclusion map from Ω_1 (Ω_2) to Ω .

The first operation is a sort of addition operation.

$$\mathcal{F}_1 \boxplus \mathcal{F}_2 := \iota_1(\mathcal{F}_1) \cup \iota_2(\mathcal{F}_2), \quad (2.5)$$

where by $\iota_1(\mathcal{F}_1)$, we mean the application of ι_1 to each element of the family \mathcal{F}_1 .

When Ω_1 and Ω_2 are disjoint, that is $\Omega = \Omega_1 \sqcup \Omega_2$, we define a multiplication operation.

$$\mathcal{F}_1 \otimes \mathcal{F}_2 := \{\iota_1(f_1) \cup \iota_2(f_2) \mid f_1 \in \mathcal{F}_1, f_2 \in \mathcal{F}_2\}. \quad (2.6)$$

The weight enumerators of the sums and products are sums and products of the corresponding weight enumerators.

Proposition 2.6. It holds that

$$\mathcal{W}(\mathcal{F}_1 \boxplus \mathcal{F}_2; x) = \mathcal{W}(\mathcal{F}_1; x) + \mathcal{W}(\mathcal{F}_2; x), \quad (2.7)$$

$$\mathcal{W}(\mathcal{F}_1 \otimes \mathcal{F}_2; x) = \mathcal{W}(\mathcal{F}_1; x) \mathcal{W}(\mathcal{F}_2; x). \quad (2.8)$$

Proof. A weight w element of $\mathcal{F}_1 \boxplus \mathcal{F}_2$ is a weight w element of \mathcal{F}_1 or of \mathcal{F}_2 , so the coefficients of the same degree add. To prove the second property, note that any weight w element of $\mathcal{F}_1 \otimes \mathcal{F}_2$ must be the disjoint union of a weight w_1 element of \mathcal{F}_1 and a weight w_2 element of \mathcal{F}_2 where $w = w_1 + w_2$. \square

The final operation we will need is a sort of composition. This should be thought of capturing the bad errors of concatenated codes. This operation will appear when we analyze the recursive use of gadgets (Theorem 4.11). Informally, given a fault-tolerant circuit C that fails with probability $P_{L,1}(p)$ and fault-tolerant gadgets that fail with probability $P_{L,2}(p)$, the circuit given by replacing every gate in C with a gate gadget of the second fault-tolerance scheme fails with probability $P_{L,1}(P_{L,2}(p))$.

Definition 2.7 (Composition). For a proposition Q and a set X , let $\mathbb{I}_Q[X]$ denote X when Q holds and \emptyset when $\neg Q$ holds.

Fix a set Ω and $\mathcal{F} \subseteq P(\Omega)$. For ease of notation, identify $\Omega \simeq [n]$. Also, consider sets $\{\omega_i\}_{i \in \Omega}$ and families of sets $\{\mathcal{S}_i \subseteq P(\omega_i)\}_{i \in \Omega}$ that are indexed by elements of Ω . We define an operation $\mathcal{F} \bullet \{\mathcal{S}_i\}_i \subseteq P(\bigsqcup_{i \in \Omega} \omega_i)$ where

$$\mathcal{F} \bullet \{\mathcal{S}_i\}_i := \boxplus_{f \in \mathcal{F}} (\mathbb{I}_{n \in f}[\mathcal{S}_n] \otimes \mathbb{I}_{n-1 \in f}[\mathcal{S}_{n-1}] \otimes \cdots \otimes \mathbb{I}_{1 \in f}[\mathcal{S}_1]). \quad (2.9)$$

Note that if $\mathcal{S}_i = \mathcal{S} \subseteq P(\omega)$ for some set ω , then $\mathcal{F} \bullet \{\mathcal{S}_i\}_i \subseteq P(\Omega \times \omega)$ corresponds to all sets

which for some element $f \in \mathcal{F}$ are elements of \mathcal{S} on each row of $\Omega \times \omega$ where f is non-trivial. We will use the notation $\mathcal{F} \bullet \mathcal{S}$ for this special case.

For a number $n \in \mathbb{N}$, and a family of sets \mathcal{S} , we use the notation $\mathcal{S}^{\bullet n}$ to denote the n -fold \bullet operation with $\mathcal{S}^{\bullet 0}$ defined to be $\{\{\bullet\}\}$ where $\{\bullet\}$ is a singleton set. The element will be clear from context.

Proposition 2.8 (Composition of weight enumerators). Using the variables as defined in Definition 2.7, when there exists a polynomial $p(x)$ such that on some interval $x \in I \subseteq \mathbb{R}_+$ for all $i \in \Omega$, $\mathcal{W}(\mathcal{S}_i; x) \leq p(x)$,

$$\mathcal{W}(\mathcal{F} \bullet \{\mathcal{S}_i\}_i; x) \leq \mathcal{W}(\mathcal{F}; p(x)) \quad (2.10)$$

for $x \in I$.

Proof. $\mathcal{F} \bullet \{\mathcal{S}_i\}_i$ is constructed as a sum over a product for each $f \in \mathcal{F}$. Using the weight enumerator sum rule, the weight enumerator is the sum of weight enumerators for each product. Each product has $|f|$ terms and the corresponding weight enumerator can be evaluated using the weight enumerator product rule. Applying the restriction $x \in I$ and the upper bound $p(x)$, each term in the sum has upper bound $p(x)^{|f|}$. For each $d \in \mathbb{N}$, the number of terms in the sum proportional to $p(x)^d$ correspond to the number of elements of \mathcal{F} of weight d . \square

Since avoiding families of bad sets imposes properties on a set, it will be convenient to define a partial order of bad sets.

Definition 2.9 (Partial order of bad sets). For a set Ω and two family of subsets $\mathcal{F}_1, \mathcal{F}_2 \subseteq P(\Omega)$, we say that $\mathcal{F}_1 \preceq \mathcal{F}_2$ if every element of \mathcal{F}_2 is a superset of some element of \mathcal{F}_1 .

$$\forall T \in \mathcal{F}_2, \quad \exists S \in \mathcal{F}_1, \quad S \subseteq T. \quad (2.11)$$

It is straightforward to see that, for a subset $X \subseteq \Omega$, if X is \mathcal{F}_1 -avoiding and $\mathcal{F}_1 \preceq \mathcal{F}_2$, then X is also \mathcal{F}_2 -avoiding.

3 Circuit formalism

In order to work with our fault-tolerant circuits and gadgets in a black-box manner, we need to model a large degree of data that describes them. This will allow us to rewrite larger fault-tolerant circuits that are built out of smaller fault-tolerant gadgets on a formal level.

3.1 Networks

We start by defining a quantum-classical circuit as a network. In what follows, we will write definitions that are heavily inspired from tensor networks. However, no knowledge of tensor networks is required.

We will use the notion of a directed graph $G = (V, E)$ with multi-edges.⁸ For an edge $e \in E$, we will say that its source is $\text{src}(e) \in V$ and its destination is $\text{dest}(e) \in V$. For a vertex $v \in V$, we use $\text{out}(v)$ to denote the number of edges $e \in E$ with source $v = \text{src}(e)$. Likewise, we use $\text{in}(v)$ to denote the number of edges $e \in E$ with destination $v = \text{dest}(e)$.

For two vertices $u, v \in V$, we say that u is an ancestor of v if there exists a sequence of vertices $w_1, w_2, w_3, \dots, w_m \in V$ that form a (directed) path from u to v . That is, there is a sequence of edges $(u, w_1), (w_1, w_2), \dots, (w_{m-1}, w_m), (w_m, v)$ in G .

Definition 3.1 (Network). A **network** is a labeled directed multi-graph (V, E, conn) described by edge labels $\text{conn}: E \rightarrow \mathbb{N} \times \mathbb{N}$, referred to as the *connections* of the network. When there exists an edge e with source v and destination u and connection $(i, j) = \text{conn}(e)$, the edge e is said to connect output i of v to input j of u . The following conditions ensure that this assignment is consistent.

- Every edge $e \in E$ is labeled by valid input and output indices.

$$\text{conn}(e) \in [\text{out}(\text{src}(e))] \times [\text{in}(\text{dest}(e))]. \quad (3.1)$$

- For every vertex $v \in V$ and every output index $i \in [\text{out}(v)]$, there is a unique edge $e \in E$ with source $v = \text{src}(e)$ and label $\text{conn}(e) = (i, j)$ for some arbitrary j .
- For every vertex $u \in V$ and every input index $j \in [\text{in}(u)]$, there is a unique edge $e \in E$ with destination $u = \text{dest}(e)$ and label $\text{conn}(e) = (i, j)$ for some arbitrary i .

The vertex set may additionally include two special vertices \perp and \top which are referred to as the **input and output vertices** (IO vertices), respectively. We require that \perp has no inputs and \top has no outputs, and refer to the set $V \setminus \{\perp, \top\}$ as the **non-IO vertices** of the network.

A network should be thought of as a directed graph between vertices where each edge connects a particular output of a vertex to a particular input of some other vertex. Note that this is very nearly the same data that defines a tensor network: In a tensor network, the connections denote contractions of tensor legs labeled “locally” by indices. The IO vertices will eventually be placeholders for quantum or classical input and outputs.

Our networks will represent circuits, whose operations (e.g., gates) correspond to the vertices, so they will need a notion of time.

Definition 3.2 (Foliated network). A foliated network $G = (V, E, \text{conn})$ of depth D is a network with a partition of vertices, denote the “time” of a vertex, $\mathsf{T}: V \rightarrow [D]$, such that every edge $e \in E$ has source and destinations at consecutive times.

$$\mathsf{T}(\text{src}(e)) + 1 = \mathsf{T}(\text{dest}(e)) \quad (3.2)$$

⁸In a multi-graph, there may be multiple edges between a pair of vertices. Thus, the edge set is $E \subseteq V \times V \times \mathbb{N}$.

The foliation will be kept implicit until needed. For a time $t \in [D]$, the set of vertices at time t , referred to as a **timestep**, is $\mathsf{T}^{-1}(t) \subseteq V$.

Note that in a foliated network, cycles are not permitted and all outputs of a vertex are connected to inputs of vertices at the following time. One can straightforwardly see that a foliated network is a **directed acyclic graph**: For any vertex $v \in V$, every ancestor $u \in V$ of v satisfies $\mathsf{T}(u) < \mathsf{T}(v)$. Thus $\mathsf{T}(\cdot)$ induces a partial order on the set of vertices. This partial order can be extended to a (possibly non-unique) total order, known as a **topological sort**, such that if $u, v \in V$ satisfies $u \leq v$, then v is not an ancestor of u . For convenience, we will refer to an arbitrary (but fixed) topological sort $(v_1, \dots, v_{|V|})$ as the ordered vertices of G and require $v_1 = \perp$ and $v_{|V|} = \top$.

3.2 Types

Each connection in a network must carry additional information. We refer to this as the “type” of a connection in analogy to types in computer programming languages. A type will specify the Hilbert space associated with a connection, as well as whether the Hilbert space is of classical or quantum nature.

Preservation of types during rewriting operations will ensure that each rewrite is valid as well as completely specify any ambiguity.

Definition 3.3 (Types). Fix a Hilbert space \mathcal{H} of n qubits and $\mathsf{cq} \in \{\mathsf{Q}, \mathsf{C}\}$ (for quantum or classical). The corresponding **type** \mathbf{t} is the tuple $(\mathcal{H}, \mathsf{cq})$. We say that such a type is quantum or classical, respectively. We use $\mathcal{H}_{\mathbf{t}}$ to denote the Hilbert space \mathcal{H} associated with the type \mathbf{t} . The set of all types is denoted \mathcal{T} , and, for a finite sequence of types $\mathbf{t}_{\bullet} = (\mathbf{t}_1, \dots, \mathbf{t}_N)$, we use $\mathcal{H}_{\mathbf{t}_{\bullet}}$ to denote $\mathcal{H}_{\mathbf{t}_1} \otimes \dots \otimes \mathcal{H}_{\mathbf{t}_N}$.

To each Hilbert space we will implicitly associate a complete set of orthogonal states that we call **computational basis states**.

We will want to describe computations with a classical component. However, we need to model the state of such computations quantum-mechanically. These are the classical-quantum states.

Definition 3.4 (Classical-quantum states). Let \mathbf{t}_{\bullet} be a sequence of types, and decompose $\mathcal{H}_{\mathbf{t}_{\bullet}}$ into two subsystems CQ where the subsystems of classical type are in C and the subsystems of quantum type are in Q . A pure state $|\psi\rangle \in \mathcal{H}_{\mathbf{t}_{\bullet}}$ is said to satisfy the type \mathbf{t}_{\bullet} if C is in a classical basis state. That is, for each classical subsystem $i \in C$, there is a computational basis state $|x_i\rangle \in \mathcal{H}_{t_i}$ such that

$$\mathrm{tr}_Q |\psi\rangle\langle\psi| = \otimes_{i \in C} |x_i\rangle\langle x_i| \quad (3.3)$$

In other words, the classical subsystem is in a well defined computational basis state. A mixed state ρ is said to satisfy the type \mathbf{t}_{\bullet} if it is a convex combination of pure states satisfying \mathbf{t}_{\bullet} . We use the notation $\mathsf{D}(\mathcal{H}_{\mathbf{t}_{\bullet}})$ or simply $\mathsf{D}(\mathbf{t}_{\bullet})$ to refer to the set of all mixed states of type \mathbf{t}_{\bullet} .

Note that these states are separable between the classical and quantum subsystems.

We now introduce operators and superoperators. For a complete discussion see [KSV02].

Definition 3.5 (Operators and Superoperators). Let \mathcal{H} and \mathcal{H}' be two finite-dimensional Hilbert spaces. We use $\mathcal{L}(\mathcal{H}', \mathcal{H})$ to refer to the space of linear operators (i.e. matrices) mapping from \mathcal{H} to \mathcal{H}' . We define the shorthand $\mathcal{L}(\mathcal{H}) \equiv \mathcal{L}(\mathcal{H}, \mathcal{H})$. The subset of positive semi-definite (PSD) trace-1 operators (density matrices) on \mathcal{H} is denoted $\mathcal{D}(\mathcal{H})$.

We define the space of superoperators $\mathcal{L}(\mathcal{H}', \mathcal{H})$ to be the set of linear maps on operators that maps from $\mathcal{L}(\mathcal{H})$ to $\mathcal{L}(\mathcal{H}')$. We say that a superoperator $Q \in \mathcal{L}(\mathcal{H}', \mathcal{H})$ is completely positive and trace-preserving (CPTP) if

- For all operators $O \in \mathcal{L}(\mathcal{H})$, the trace of O is preserved in the sense that $\text{tr}(O) = (\text{tr} \circ Q)(O)$.
- For all Hilbert spaces \mathcal{H}_E , Q extended by the identity on the auxillary Hilbert space \mathcal{H}_E maps PSD operators to PSD operators. That is, for all positive semi-definite operators $O \in \mathcal{L}(\mathcal{H} \otimes \mathcal{H}_E)$, $(Q \otimes \text{Id}_{\mathcal{H}_E})(O)$ is positive semi-definite where $\text{Id}_{\mathcal{H}_E}$ is the identity superoperator.

Definition 3.6 (Operator Types). For an operator $O \in \mathcal{L}(\mathcal{H}', \mathcal{H})$ and two sequences of types $\mathbf{t}_{\bullet}^{\text{in}} = (\mathbf{t}_1^{\text{in}}, \dots, \mathbf{t}_N^{\text{in}})$ and $\mathbf{t}_{\bullet}^{\text{out}} = (\mathbf{t}_1^{\text{out}}, \dots, \mathbf{t}_M^{\text{out}})$, O is said to have input type $\mathbf{t}_{\bullet}^{\text{in}}$ and output type $\mathbf{t}_{\bullet}^{\text{out}}$ if

- $\mathcal{H} = \mathcal{H}_{\mathbf{t}_{\bullet}^{\text{in}}}$ and $\mathcal{H}' = \mathcal{H}_{\mathbf{t}_{\bullet}^{\text{out}}}$
- Every pure state $|\psi\rangle \in \mathcal{H}_{\mathbf{t}_{\bullet}^{\text{in}}}$ of type $\mathbf{t}_{\bullet}^{\text{in}}$ is mapped to a (not necessarily normalized) state $|\phi\rangle = O|\psi\rangle \in \mathcal{H}_{\mathbf{t}_{\bullet}^{\text{out}}}$ of type $\mathbf{t}_{\bullet}^{\text{out}}$.

The set of operators of input type $\mathbf{t}_{\bullet}^{\text{in}}$ and output type $\mathbf{t}_{\bullet}^{\text{out}}$ will be written as $\mathcal{L}(\mathbf{t}_{\bullet}^{\text{out}}, \mathbf{t}_{\bullet}^{\text{in}})$.

Definition 3.7 (Gate). For two sequences of types $\mathbf{t}_{\bullet}^{\text{in}} = (\mathbf{t}_1^{\text{in}}, \dots, \mathbf{t}_N^{\text{in}})$ and $\mathbf{t}_{\bullet}^{\text{out}} = (\mathbf{t}_1^{\text{out}}, \dots, \mathbf{t}_M^{\text{out}})$ and an superoperator \mathcal{G} , \mathcal{G} is said to be a **gate** with **input type** $\mathbf{t}_{\bullet}^{\text{in}}$ and **output type** $\mathbf{t}_{\bullet}^{\text{out}}$ if it has a Kraus representation $\{K_{\mu}\}_{\mu}$ where each Kraus operator K_{μ} has the corresponding input type and output type $K_{\mu} \in \mathcal{L}(\mathbf{t}_{\bullet}^{\text{out}}, \mathbf{t}_{\bullet}^{\text{in}})$. For short, we say that the gate has type $(\mathbf{t}_{\bullet}^{\text{in}}, \mathbf{t}_{\bullet}^{\text{out}})$. We use $\mathcal{L}(\mathbf{t}_{\bullet}^{\text{out}}, \mathbf{t}_{\bullet}^{\text{in}})$ to denote the set of all such superoperators.

We call \mathcal{G} a **physical gate** if it is additionally completely-positive and trace-preserving (CPTP).

Remark 3.8. We will use “non-physical” gates to describe general and often adversarial models of faults in a circuit, and later prove thresholds against such faults.

We are now ready to state the central definition of our framework, a classical-quantum circuit (“quantum circuit” or just “circuit” for short).

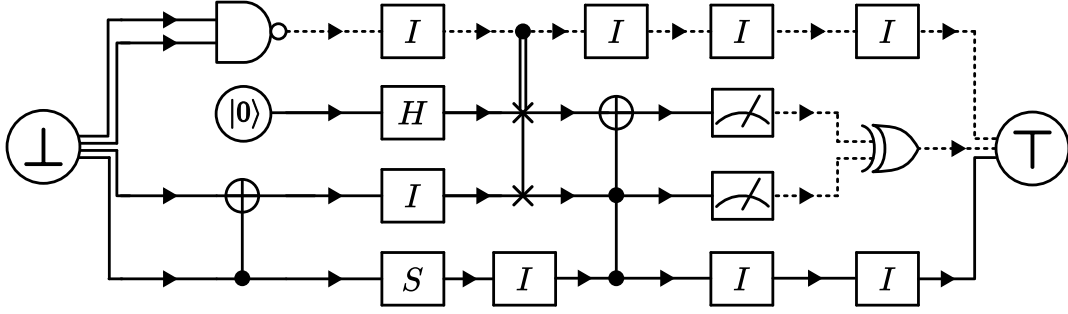


Figure 1: Illustration of a classical-quantum circuit. Taking two qubits and two bits as input and outputting one qubit and two bits. Types of edges and connections are annotated in red.

Definition 3.9 (Classical-quantum circuit). Let (V, E, conn) be a foliated network on the vertices V which we refer to as **locations**. Let $\text{type}: E \rightarrow \mathcal{T}$ be an assignment of a type to each edge. For each non-IO vertex $v \in V \setminus \{\perp, \top\}$, we assign a classical-quantum gate $\text{gate}(v)$ (Definition 3.7) such that $\text{gate}(v)$ satisfies the type of the corresponding input and output edges.^a For convenience, we define $\text{gate}(\perp)$ and $\text{gate}(\top)$ to be the identity – these vertices correspond to inputs and outputs of the circuit. The data $C = (V, E, \text{conn}, \text{type}, \text{gate})$ is said to be a classical-quantum circuit.

^aFormally, in the sequence of input types, $\text{t}_i^{\text{in}} = \text{type}(e)$ when $\text{dest}(e) = v$ and $\text{conn}(e) = (i, \cdot)$. Likewise, in the sequence of output types, $\text{t}_j^{\text{out}} = \text{type}(e)$ when $\text{src}(e) = v$ and $\text{conn}(e) = (\cdot, j)$.

Recall that the depth of a network was defined in Definition 3.2.

Definition 3.10 (Circuit width). For a classical quantum circuit $C = (V, E, \text{conn}, \text{type}, \text{gate})$, we say that a vertex is a **quantum location** if any edge e incident to it is of quantum type. Likewise, a vertex is a **classical location** if all edges e incident to it are of classical type.

The quantum width W_t of a timestep t of a circuit is the sum of the maximum over quantum inputs and outputs of each vertex. We introduce the following abuse of notation, for a vertex $v \in V$ and an index $i \in \text{in}(v)$, (v, i) uniquely specifies the edge e with source $\text{src}(e) = v$ and $\text{conn}(e) = (i, \cdot)$. Here, we will label edges by such pairs and an analogous definitions defined for a vertex $v \in V$ and an index $j \in \text{out}(v)$, (j, v) .

$$W_t = \sum_{v \in \mathcal{T}^{-1}(t)} \max(|\{i \in \text{in}(v) \mid \text{type}((v, i)) = (\cdot, \text{Q})\}|, |\{j \in \text{out}(v) \mid \text{type}((j, v)) = (\cdot, \text{Q})\}|) \quad (3.4)$$

The quantum width of a circuit W is the maximum quantum width in any timestep, $W = \max_{t \in [D]} W_t$. The classical width of a circuit (W_C) is defined analogously with Q replaced by C. We use the term “total width” to refer to the sum $W + W_C$ of quantum and classical widths. In what follows, we will simply refer to the quantum width as the width.

Note that a classical-quantum circuit is *not* a channel – many distinct circuits implement the

same channel. However, these circuits may have very different fault-tolerance properties. We now describe the channel implemented by the circuit.

Definition 3.11 (Circuit implementation). To every classical-quantum circuit C specified by the tuple of data $(V, E, \text{conn}, \text{type}, \text{gate})$, we can define a superoperator in the following way: Let $(v_1, \dots, v_{|V|})$ be the ordered locations (vertices) of C . Then, we define $\text{map}[C]$ to be the superoperator

$$\text{map}[C] = \text{gate}(v_{|m|}) \circ \text{gate}(v_{m-1}) \circ \dots \circ \text{gate}(v_2) \circ \text{gate}(v_1) \quad (3.5)$$

Where, for each $t \in [m]$, $\text{gate}(v_t)$ is suitably extended by a tensor product with the identity in the canonical way, and for each edge $(u, v) \in E$ with $(i, j) \in \text{conn}$, the j -th tensor factor in the input Hilbert space of $\text{gate}(v)$ is identified with the i -th tensor factor in the output Hilbert space of $\text{gate}(u)$. The ordering of tensor factors for the input $\text{gate}(\perp)$ and output $\text{gate}(\top)$ is the same as their output or input, respectively.

In order to work with sets of gates (e.g. gadgets), we will also introduce vertex contractions. Informally, a circuit contraction is simply a circuit where all the gates in a subset are grouped together into a single big gate.

Definition 3.12 (Circuit Contraction). For a classical-quantum circuit $C = (V, E, \text{conn}, \text{type}, \text{gate})$ with foliation $\top : V \rightarrow [D]$, we can **contract** vertices together to form a new circuit. More precisely, let V_κ be a set of vertices and let $\kappa : V \rightarrow V_\kappa$ be a surjective map. κ induces vertex contractions of C : for every vertex $u \in V_\kappa$, we contract the vertices $\kappa^{-1}(u)$ which are mapped to u to form a new network (V_κ, E_κ) . Note that $|E| = |E_\kappa|$. We say that the contraction is **valid** if this new network is acyclic. Vertex contractions induce new labelings $\text{conn}_\kappa, \text{type}_\kappa, \text{gate}_\kappa$, and therefore a new circuit $C_\kappa = (V_\kappa, E_\kappa, \text{conn}_\kappa, \text{type}_\kappa, \text{gate}_\kappa)$. See Figure 2 for an illustration.

We say that a contraction κ is **depth-preserving** if the contracted circuit C_κ has a foliation $\top_\kappa : V_\kappa \rightarrow [D]$ of the same depth D such that every vertex $v \in V$ is contracted with other vertices in the same timestep, namely,

$$\top_\kappa(\kappa(v)) = \top(v). \quad (3.6)$$

More generally, we say that a contraction κ is **foliation-preserving** if the contracted circuit C_κ has a foliation $\top_\kappa : V_\kappa \rightarrow [D_\kappa]$ such that for all timesteps $t \in [D]$, there is a new timestep $t_\kappa \in [D_\kappa]$ where all vertices in V at timestep t are contracted into. Precisely,

$$\forall t \in [D], \exists t_\kappa \in [D_\kappa] \quad \kappa(\top^{-1}(t)) \subseteq \top_\kappa^{-1}(t_\kappa). \quad (3.7)$$

In this work, we require that all contractions are valid.

The notion of circuit contraction is employed to define faults below. On a high level, the reason for this is that, in our noise model, the faulty locations of the circuit may be replaced by an arbitrary (global) superoperator. In order to capture faults that perform operations between parts of the circuit, we allow the fault to act on a contracted circuit.

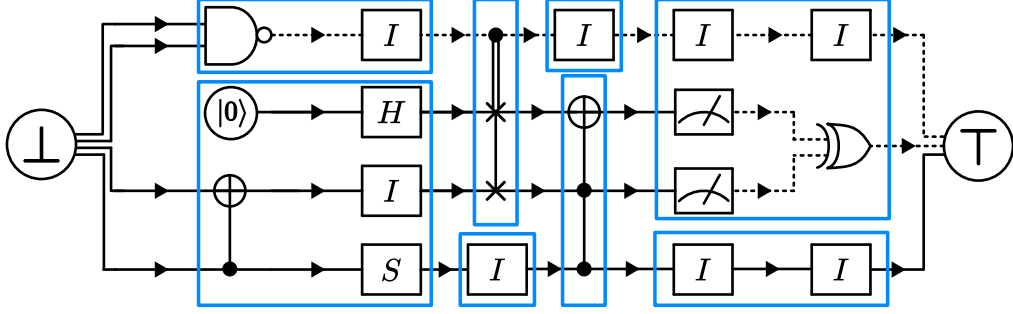


Figure 2: Illustration of a circuit contraction. Note that the indexing of inputs and outputs is modified.

Definition 3.13 (Faults). Fix a classical-quantum circuit $C = (V, E, \text{conn}, \text{type}, \text{gate})$ and ordered locations $(v_1, \dots, v_{|V|})$. A **fault** $\mathbf{f} = (\kappa, \widetilde{\text{gate}})$ consists of a depth-preserving contraction map $\kappa: V \rightarrow V_\kappa$ and a (faulty) gate map $\widetilde{\text{gate}}$ with domain V_κ such that the contracted circuit $C_\kappa = (V_\kappa, E_\kappa, \text{conn}_\kappa, \text{type}_\kappa, \text{gate}_\kappa)$ is a valid classical-quantum circuit when the induced gate map gate_κ is replaced by the faulty gate map $\widetilde{\text{gate}}$. We say that the support of $\widetilde{\text{gate}}$, known as the **fault path**, is the set of vertices in V that are contracted non-trivially. Precisely,

$$\text{supp}(\widetilde{\text{gate}}) = \{v \in V \mid \widetilde{\text{gate}}(\kappa(v)) \neq \text{gate}(v)\}. \quad (3.8)$$

From now on, we use \mathbf{f} to refer to a set of data $(\kappa, \widetilde{\text{gate}})$ compatible with C and $C[\mathbf{f}]$ to refer to the faulty circuit $(V_\kappa, E_\kappa, \text{conn}_\kappa, \text{type}_\kappa, \widetilde{\text{gate}})$. We write $\text{supp}(\mathbf{f})$ as an alias for $\text{supp}(\widetilde{\text{gate}})$.

In what follows, we will mostly concern with faults supported only on quantum locations. This restriction is not necessary as classical fault-tolerance can be constructed and analyzed in our framework as well.

It is common to first prove fault-tolerance against insertions of Pauli errors into the circuit and then reduce the general case to the Pauli error case. Thus we will define a notion of a *Pauli fault*.

Definition 3.14 (Pauli superoperator). A superoperator \mathcal{E} is said to be a **Pauli superoperator** if, for some Pauli operators P, P' and scalar $c \in \mathbb{C}$, \mathcal{E} can be written as

$$\mathcal{E}(\rho) = cP\rho P' \quad (3.9)$$

If $P = P'$ then \mathcal{E} is said to be a **diagonal Pauli superoperator**.

As we allow P, P' to differ, Definition 3.14 will naturally capture how general errors “decohere” to Pauli errors (diagonal Pauli superoperators). In general, the strategy to prove fault tolerance will be to decompose general faults into (non-diagonal) *Pauli faults* which apply Pauli superoperators (see Lemma 4.12). For example, consider the following channel that applies an S gate with probability

p .

$$\rho \mapsto (1 - p)\rho + pS\rho S^\dagger \quad (3.10)$$

Defining the coefficient $\theta = \frac{e^{i\pi/4}}{\sqrt{2}}$, we can decompose S as $S = \theta I + \theta^* Z$ and write the channel as a sum of Pauli superoperators

$$\rho \mapsto \left(1 - \frac{p}{2}\right) |\rho| + \frac{p}{2} Z\rho Z + \frac{ip}{2} |\rho Z - \frac{ip}{2} Z\rho| . \quad (3.11)$$

Definition 3.15 (Pauli fault). Fix a classical-quantum circuit $C = (V, E, \text{conn}, \text{type}, \text{gate})$, and a fault $\mathbf{f} = (\kappa: V \rightarrow V_\kappa, \widetilde{\text{gate}})$. The fault \mathbf{f} is said to be a **Pauli fault** if the faulty gate at every contracted vertex $v \in V_\kappa$ can be written as the original gate with Pauli superoperators occurring before and after. That is, for some Pauli superoperators $\mathcal{P}_v, \mathcal{P}'_v$

$$\widetilde{\text{gate}}(v) = \mathcal{P}'_v \circ \text{gate}_\kappa(v) \circ \mathcal{P}_v . \quad (3.12)$$

If this property holds for diagonal Pauli superoperators \mathcal{P}_v and \mathcal{P}'_v , then \mathbf{f} is said to be a **diagonal Pauli fault**.

While allowing Pauli superoperators to act both before and after an ideal gate may seem somewhat strange, it captures many faults that a less general definition (such as limiting Pauli superoperators to only act after an ideal gate) cannot. Consider, for example, a fault that replaces a reset gate by the identity gate. By utilizing extra ancilla (e.g. by swapping qubits to the ancilla, apply any noise superoperator, and swapping back), we can decompose the map of a circuit with an arbitrary fault into a sum of maps of circuits with this ancilla and Pauli faults (see Lemma 4.12). Such ancilla will be part of an *environment circuit*, see Definition 3.17.

Remark 3.16 (Pauli fault). It is a crucial property that for Pauli faults, the contraction can always be taken to be trivial. More precisely, since all Pauli superoperators can be written as the tensor product of single-qubit Pauli superoperators, there always exists an equivalent fault on the uncontracted vertex set. This property will allow us to analyze gadgets in isolation despite faults inserting superoperators with support acting between gadgets. The presence of the ideal gate allows the behavior of the faulty circuit to be compared to the ideal one.

As we will see in Theorem 4.11, it is unfortunately the case that physical level diagonal Pauli faults do not imply that “logical level” faults are diagonal or even Pauli.^a

^aFor example, the failure of a magic state consumption gadget caused by a measurement error.

In order to fully specify the operation of the circuit with noise, we need a model of the environment. In what follows, the environment should be thought of as a book-keeping method for tracking an unspecified second quantum circuit that is essentially unrestricted.

Definition 3.17 (Circuit with environment). For a classical-quantum circuit C specified by the data $(V, E, \text{conn}, \text{type}, \text{gate})$, the circuit C **with environment** refers to an (arbitrary) extension $C_{\text{env}} = (V \sqcup V_{\Omega}, E \sqcup E_{\Omega}, \text{conn}', \text{type}', \text{gate}')$ of the original circuit C where the sub-circuit induced by the nodes V of C_{env} is the original circuit C . C is called the **computational circuit** of C_{env} , and the sub-circuit C_{Ω} induced by the nodes V_{Ω} is called the **environment circuit** (of C_{env}).

Frequently, we will assume that the environment circuit is **closed**, which means it has no input or output edges. In other words, the environment circuit introduces its own workspace and traces out its output at the end.

For a circuit with environment, its width (depth) refers to the width (depth) of the computational circuit.

The apparent imprecision of the environment circuit is deliberate. The fault-tolerance statements we prove will hold for arbitrary environment circuits. Note that as stated, there is no interaction between the computational and environment circuits because the vertex and edge sets are disjoint. The environment circuit and computational circuits may only interact via faults which contracts vertices between the two (otherwise independent) networks.

Remark 3.18 (Adversarial faults). A fault should be thought of as replacing the gates at locations in the fault path with *arbitrary* superoperators. These superoperators, which may be supported on the environment, are not necessarily independent between locations. In other words, they may entangle the qubits of distinct locations of the original circuit. This models an arbitrarily powerful adversary with a memory that is permitted access to the part of the computational state supported in the fault every timestep.

4 Fault-tolerant gadgets

For a fault-tolerant circuit, the connections in the circuit carry quantum information encoded in quantum error correcting codes. Therefore, a collection of connections could have a **code type**, which specifies the encoding and protection of the carried information. Such code types will be convenient for working with gadgets since a general fault-tolerance scheme may employ many different codes distinct from the code protecting the bulk of the computation in different places (for example code switching or preparing resource states). This feature will allow our statements to be agnostic to the choice of code used. To define code types, we will need to first characterize physical errors on quantum states.

Our notion of a damaged state is partly inspired by [Kit97] and [ABO97].

Definition 4.1 (Deviation). For a set of qubits A , a family of bad sets $\mathcal{B} \subseteq P(A)$, and two states $\sigma, \tilde{\sigma}$ on A , $\tilde{\sigma}$ is said to be **\mathcal{B} -deviated** from σ if there exists a \mathcal{B} -avoiding set $S \subseteq A$ and a (not necessarily CPTP) superoperator \mathcal{E}_S supported only on S such that $\tilde{\sigma} = (\mathcal{I}_{A \setminus S} \otimes \mathcal{E}_S)(\sigma)$.

We denote this relationship by

$$\sigma \lesssim_{\mathcal{B}} \tilde{\sigma}. \quad (4.1)$$

For a subspace \mathcal{Q} of the Hilbert space, a state $\tilde{\sigma}$ is said to be \mathcal{B} -deviated from \mathcal{Q} if there exists a (possibly mixed) state σ in \mathcal{Q} such that $\tilde{\sigma}$ is \mathcal{B} -deviated from σ . We use the less general term “Pauli \mathcal{B} -deviated” (“diagonal Pauli \mathcal{B} -deviated”) when the superoperator \mathcal{E}_S is a Pauli superoperator (diagonal Pauli superoperator).

In general, our circuits may have data encoded in different quantum codes. Formally, we attach this data to groups of qubits using *code types*. The code types will track the code (including a choice of logical basis) and any demands that will be made on damaged code states.

Definition 4.2 (Code Type). For a n qubit Hilbert space, let $r = n - k$, a quantum code \mathcal{Q} with k logical qubits is specified by a **decoding unitary** $U : \mathcal{H}^n \rightarrow \mathcal{H}^k \otimes \mathcal{H}^r$ and a family of **bad error supports** $\mathcal{B} \subseteq P([n])$ such that

- The codespace \mathcal{Q} is specified by U^\dagger , the **encoding unitary**, acting on the k qubits logical information and the trivial syndrome,

$$\mathcal{Q} = \{U^\dagger |\psi\rangle |0^r\rangle \mid |\psi\rangle \in \mathcal{H}^k\}. \quad (4.2)$$

We define the **reversible encoding channel** as $\text{renc}(\rho) = U^\dagger \rho U$ and the **reversible decoding channel** as its inverse $\text{rdec}(\sigma) = U \sigma U^\dagger$.

- For any logical state $\rho \in \mathcal{D}(\mathcal{H}^k)$, for any state $\tilde{\sigma}$ that is \mathcal{B} -deviated from an encoded logical state $\sigma = U^\dagger(\rho \otimes |0^r\rangle\langle 0^r|)U$, the decoded state^a is proportional to the logical state.^b

$$\text{tr}_{\mathcal{H}^r}(\text{rdec}(\tilde{\sigma})) \propto \rho \quad (4.3)$$

We define the **irreversible encoding channel** as $\text{enc}(\rho) = \text{renc}(\rho \otimes |0^r\rangle\langle 0^r|)$, and the **irreversible decoding channel** as $\text{dec}(\sigma) = \text{tr}_{\mathcal{H}^r}(\text{rdec}(\sigma))$.

A **code type** of \mathcal{Q} is the tuple of data $\text{ctype}_{\mathcal{Q}} = (U, \mathcal{B})$. The **logical type** of $\text{ctype}_{\mathcal{Q}}$ is the type corresponding to an unencoded register $(\mathcal{H}^k, \text{cq})$. Likewise the **syndrome type** of $\text{ctype}_{\mathcal{Q}}$ is the type corresponding to the syndrome needed to make the encoding map unitary $(\mathcal{H}^r, \text{cq})$.

For convenience, we define a **trivial code type** $\text{ctype}_I = (I, \{\{1\}, \dots, \{n\}\})$ to indicate that the qubits (or bits) are unencoded and unprotected. Similarly, we can trivially extend a code type to an arbitrary number of unencoded qubits (or bits) by extending the unitary U with identity operators.

^aThe term “state” is an abuse of terminology, since in general, the result is not completely positive or trace-1.

^bIn particular, this implies that errors on \mathcal{B} -avoiding sets are correctable and that the proportionality constant may only depend on the error.

Remark 4.3. Typically, in quantum error correction, we think of an encoding unitary as Clifford. However, in this case, the constraint that rdec can successfully recover states that are \mathcal{B} -deviated from the codespace means that rdec and its inverse, renc is quite complicated in general.^a Since rdec is a proof tool, this does not present any complications.

^aThink of it as the purification of a channel that measures the syndrome and corrects the state.

For correctable states, the application of the decoder decouples the logical subsystem from the syndrome subsystem. We will later use this property for showing that correctly operating gadgets decouple the computation from the action of the fault.

Proposition 4.4 (Decoupling of Errors). Let \mathcal{Q} be a code on n qubits with code type (U, \mathcal{B}) . For every superoperator \mathcal{E} supported on a subset of qubits $B \subseteq [n]$ that is \mathcal{B} -avoiding, there exists an operator $v_{\mathcal{E}} \in \mathcal{L}(\mathcal{H}^r)$ such that for all $\rho \in \mathcal{D}(\mathcal{H}^k)$, the decoded logical subsystem is unentangled from the syndrome subsystem

$$\text{rdec} \circ \mathcal{E} \circ \text{enc}(\rho) = \rho \otimes v_{\mathcal{E}} . \quad (4.4)$$

Proof. Let E be a Pauli operator whose support is \mathcal{B} -avoiding. Consider a logical state $|\psi\rangle \in \mathcal{H}^k$ and the decoded state $|\psi_E\rangle = UEU^\dagger(|\psi\rangle|0^r\rangle)$. From definition (Definition 4.2) decoder map satisfies (Eq. (4.3)) that $\text{tr}_{\mathcal{H}^r}(|\psi_E\rangle\langle\psi_E|) \propto |\psi\rangle\langle\psi|$.⁹ Since the result of the partial trace is a pure state, this implies that $|\psi_E\rangle$ is unentangled between the logical and syndrome subsystems. That is, we can write $|\psi_E\rangle = |\psi\rangle|s_{\psi,E}\rangle$ for some pure state $|s_{\psi,E}\rangle \in \mathcal{H}^r$ of the syndrome subsystem.

We now use linearity to show that $|s_{\psi,E}\rangle$ is independent of ψ . Consider a superposition of any two logical states $|\psi\rangle, |\phi\rangle \in \mathcal{H}^k$ written as $|\varphi\rangle = \alpha|\psi\rangle + \beta|\phi\rangle$. Applying the previous argument to $|\varphi\rangle$, $|\phi\rangle$ and $|\psi\rangle$, we have that

$$UEU^\dagger(|\varphi\rangle|0^r\rangle) = \alpha|\psi\rangle|s_{\psi,E}\rangle + \beta|\phi\rangle|s_{\phi,E}\rangle \quad (4.5)$$

$$UEU^\dagger(|\varphi\rangle|0^r\rangle) = (\alpha|\psi\rangle + \beta|\phi\rangle)|s_{\varphi,E}\rangle . \quad (4.6)$$

It must therefore be the case that $|s_{\psi,E}\rangle = |s_{\phi,E}\rangle = |s_{\varphi,E}\rangle$. In other words, the syndrome state $|s_E\rangle$ is independent of the logical state and only depends on the error E .

We now consider the case of a more general error superoperator \mathcal{E} (not necessarily CPTP). \mathcal{E} can be written as a linear combination of Pauli superoperators by expanding the Kraus operators in a basis of Pauli matrices. Therefore, we can write $\mathcal{E}(\sigma) = \sum_{\mu,\nu} \alpha_{\mu,\nu} K_\mu \sigma K'_\nu$ for Pauli operators $\{K_\mu\}_\mu$ supported on B and complex coefficients $\{\alpha_{\mu,\nu}\}_{\mu,\nu}$. Then, we can apply the previous argument to

⁹In fact, we have equality since all superoperators in the expression are CPTP.

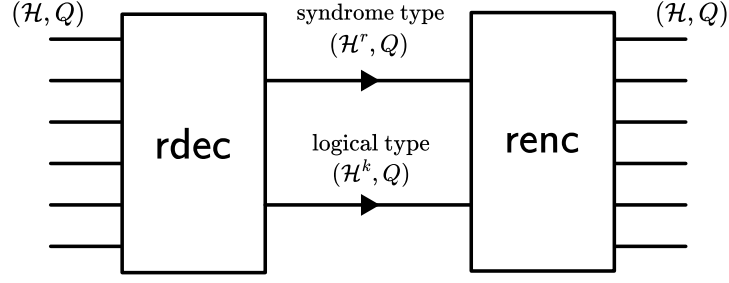


Figure 3: Reversible decoder and encoders of a quantum code.

an arbitrary pure state ψ ,

$$\text{rdec} \circ \mathcal{E} \circ \text{enc}(\psi) = \sum_{\mu, \nu} \alpha_{\mu, \nu} U K_{\mu} U^{\dagger} (\psi \otimes |0^r\rangle\langle 0^r|) U K'_{\nu} U^{\dagger} \quad (4.7)$$

$$= \sum_{\mu, \nu} \alpha_{\mu, \nu} \psi \otimes |s_{\mu}\rangle\langle s_{\nu}| \quad (4.8)$$

$$= \psi \otimes \left(\sum_{\mu, \nu} \alpha_{\mu, \nu} |s_{\mu}\rangle\langle s_{\nu}| \right) \quad (4.9)$$

for syndrome states $|s_{\mu}\rangle, |s_{\nu}\rangle$ independent of ρ . The same conclusion holds for mixed states ρ by considering a purification. \square

The code types will need to be assigned to sets of edges in a circuit that share the same source and destination. To do this, we introduce a notion of a “bundle” of edges.

Definition 4.5 (Bundled Circuit). For a classical-quantum circuit $C = (V, E, \text{conn}, \text{type}, \text{gate})$, if there are multiple edges with the same source and destination vertices (which is common in contracted circuits), we may **bundle** these edges together, and label them with code specifications. Specifically, a bundling is a surjective function $\beta : E \rightarrow E_{\beta}$ from the edges of C to a set E_{β} called the *bundles of C* such that all edges in a bundle $b \in E_{\beta}$ share the same source, destination, and type. That is, for $e_1, e_2 \in \beta^{-1}(b)$

$$\text{src}(e_1) = \text{src}(e_2), \quad \text{dest}(e_1) = \text{dest}(e_2), \quad \text{type}(e_1) = \text{type}(e_2) \quad (4.10)$$

We can now assign code types to the bundles of a circuit.

Definition 4.6 (Code Specification). For a circuit $C = (V, E, \text{conn}, \text{type}, \text{gate})$ and a bundling $\beta : E \rightarrow E_{\beta}$, we define a **code specification** to be a labeling **spec** which assigns a code type to every edge bundle $b \in E_{\beta}$ that is compatible in the sense that the edges of the edge bundle are in bijection with the qubits of the code type. To reduce notation, this bijection will be implicit and clear from context.

In order to concisely talk about input and output behavior of circuits with respect to inputs and outputs satisfying the code types, let us now introduce notation for the various data induced by the code specification.

Definition 4.7 (Circuit Input/Output). Let $C = (V, E, \text{conn}, \text{type}, \text{gate})$ be a classical-quantum circuit, and fix a bundling $\beta : E \rightarrow E_\beta$ and specification spec . In order to state various input and output properties of the circuit, we will introduce the following notations which are induced from the specification data in the obvious way. By input (output) code types, we mean the input (output) code types associated with each input (output) edge bundle. Likewise, we introduce the terms input/output logical (syndrome) types associated with each edge bundle.

Suppose the circuit C has ℓ input and m output edge bundles, respectively. Let $(\mathcal{B}_1^{\text{in}}, \dots, \mathcal{B}_\ell^{\text{in}})$ correspond to the bad error supports of the input code types. We define the **input bad error supports** \mathcal{B}^{in} of C as the sum of the bad error supports of the input code type.^a We define the **output bad error supports** \mathcal{B}^{out} analogously with the bad error supports of the output code types $(\mathcal{B}_1^{\text{out}}, \dots, \mathcal{B}_m^{\text{out}})$.

$$\mathcal{B}^{\text{in}} = \mathcal{B}_1^{\text{in}} \boxplus \dots \boxplus \mathcal{B}_\ell^{\text{in}} \quad (4.11)$$

$$\mathcal{B}^{\text{out}} = \mathcal{B}_1^{\text{out}} \boxplus \dots \boxplus \mathcal{B}_m^{\text{out}} \quad (4.12)$$

Similarly, let $(\text{renc}_1^{\text{in}}, \dots, \text{renc}_\ell^{\text{in}})$ correspond to the reversible encoders of the input code types. This induces an **input reversible encoder** renc^{in} for the whole circuit given by applying each reversible encoder to each edge bundle. Again, we define an analogous output reversible encoder renc^{out} using the reversible encoder of the output code types $(\text{renc}_1^{\text{out}}, \dots, \text{renc}_m^{\text{out}})$.

$$\text{renc}^{\text{in}} = \text{renc}_1^{\text{in}} \otimes \dots \otimes \text{renc}_\ell^{\text{in}} \quad (4.13)$$

$$\text{renc}^{\text{out}} = \text{renc}_1^{\text{out}} \otimes \dots \otimes \text{renc}_m^{\text{out}} \quad (4.14)$$

Decoders and their irreversible counterparts are similarly defined in the obvious way.

^aWe leave the bijection between the input edges of C and the qubits of each bundle and code type implicit.

4.1 Gadgets

We are now ready to state our definition of a fault-tolerant gadget. Under some circumstances, most commonly concatenating distance-3 codes [AGP05], a circuit may satisfy multiple gadget definition for various combinations of parameters. E.g. stronger conditions on the fault may imply stronger conditions on the output error or weaker conditions on the input error.

Our definition of gadget most closely aligns with [Kit97] and [CFG24] although we desire that our gadgets *exactly* simulate the operation when the state is good.

Definition 4.8 (Fault-tolerant Gadget). Let $C = (V, E, \text{conn}, \text{type}, \text{gate})$ be a classical-quantum circuit with input and output types $\mathbf{t}_\bullet^{\text{in}}, \mathbf{t}_\bullet^{\text{out}}$. Fix a bundling $\beta : E \rightarrow E_\beta$ and associated code specification spec . Let $\mathcal{F} \subseteq P(V)$ be a collection of bad fault paths.

Let C_{env} be the computational circuit C with an arbitrary environment circuit C_{Ω} (recall Definition 3.17) with input and output types $\mathbf{t}_{\Omega}^{\text{in}}, \mathbf{t}_{\Omega}^{\text{out}}$. We use the circuit input/output definitions (input/output bad error supports \mathcal{B}^{in} and \mathcal{B}^{out} , etc.) from Definition 4.7 extended to the environment circuit in the trivial way by assigning trivial code types to all edges of the environment circuit.

Let \mathcal{G} be a gate with input and output types $\mathbf{t}_{\mathcal{G}}^{\text{in}}, \mathbf{t}_{\mathcal{G}}^{\text{out}}$. We say that C is a **fault-tolerant gadget** which simulates \mathcal{G} with respect to the data $(\beta, \text{spec}, \mathcal{F})$ if the input and output logical types of C (under spec) match the input/output types of \mathcal{G} and the following conditions hold. For any \mathcal{F} -avoiding (non-diagonal) *Pauli* fault \mathbf{f} supported only on the computational circuit,^a there exists superoperators $\tilde{\mathcal{G}}_{\mathbf{f}} \in \mathcal{L}(\mathbf{t}_{\bullet}^{\text{out}}, \mathbf{t}_{\bullet}^{\text{in}})$ and $\mathcal{R}_{\mathbf{f}} \in \mathcal{L}(\mathbf{t}_{\bullet}^{\text{in}}, \mathbf{t}_{\bullet}^{\text{in}})$ such that the circuit map can be decomposed as^b

$$\text{map}[C_{\text{env}}[\mathbf{f}]] = \left(\tilde{\mathcal{G}}_{\mathbf{f}} \otimes \text{map}[C_{\Omega}] \right) \circ (\mathcal{R}_{\mathbf{f}} \otimes I_{\Omega}). \quad (4.15)$$

These superoperators satisfy the following properties.

Friendly For an *arbitrary* state $\sigma \in \mathcal{L}(\mathbf{t}_{\bullet}^{\text{in}} \otimes \mathbf{t}_{\Omega}^{\text{in}})$, there exists a logical state $\rho \in \mathcal{D}(\mathbf{t}_L^{\text{in}} \otimes \mathbf{t}_{\Omega}^{\text{in}})$ such that

$$\text{enc}^{\text{in}}(\rho) \lesssim_{\mathcal{B}^{\text{in}}} (\mathcal{R}_{\mathbf{f}} \otimes I_{\Omega})(\sigma). \quad (4.16)$$

Moreover, $\mathcal{R}_{\mathbf{f}}$ acts as logical identity on “good” input states that are \mathcal{B}^{in} -deviated from the codespace. That is, for some logical state $\rho \in \mathcal{D}(\mathbf{t}_L^{\text{in}} \otimes \mathbf{t}_{\Omega}^{\text{in}})$,

$$\text{enc}^{\text{in}}(\rho) \lesssim_{\mathcal{B}^{\text{in}}} \sigma \implies (\mathcal{R}_{\mathbf{f}} \otimes I_{\Omega})(\sigma) \lesssim_{\mathcal{B}^{\text{in}}} \sigma. \quad (4.17)$$

We refer to the superoperator $\mathcal{R}_{\mathbf{f}}$ as a **filter**.^c Furthermore, $\mathcal{R}_{\mathbf{f}}$ factorizes into operators acting on each input edge bundle.

$$\mathcal{R}_{\mathbf{f}} = \mathcal{R}_1 \otimes \cdots \otimes \mathcal{R}_{\ell}. \quad (4.18)$$

Simulation For a “good” input state σ with $\text{enc}^{\text{in}}(\rho) \lesssim_{\mathcal{B}^{\text{in}}} \sigma$, the output is \mathcal{B}^{out} -deviated from the encoding of the logical operation \mathcal{G} applied to the logical state ρ .

$$\text{enc}^{\text{out}} \circ (\mathcal{G} \otimes \text{map}[C_{\Omega}])(\rho) \lesssim_{\mathcal{B}^{\text{out}}} \left(\tilde{\mathcal{G}}_{\mathbf{f}} \otimes \text{map}[C_{\Omega}] \right)(\sigma). \quad (4.19)$$

^aNote that this assumption is reasonable since we will apply Lemma 4.12 to the entire circuit being analyzed, which turns an arbitrary fault into Pauli faults.

^b I_{Ω} is the identity acting on the environment subsystem.

^cThis serves the same role as filters in [AGP05].

In the above definition, the friendly property can be dropped¹⁰ in many contexts where recursive simulation is not required. Friendly gadgets corresponds roughly to a “strong” simulation of [Kit97].

With this definition, the key technical work in proving fault-tolerance becomes proving that various

¹⁰In which case one can set $\mathcal{R}_{\mathbf{f}}$ to be identity.

circuits satisfy this gadget definition with the desired code types. Later in Section 6, we will show that the standard error correction circuit for a qLDPC code, which repeatedly measure stabilizers for $O(d)$ rounds, is a friendly and fault-tolerant gadget which simulates identity. We then compose this gadget with other circuits that implement logical operations, such as transversal gates.

We prove the following composition result Proposition 4.9 which allows gadgets to be assembled into new gadgets. This lemma will be used in many locations in Section 7 without reference.

Proposition 4.9 (Composition of gadgets). Suppose C_1, C_2 are fault-tolerant gadgets for gates $\mathcal{G}_1, \mathcal{G}_2$ with respect to the data $(\kappa_1, \text{spec}_1, \mathcal{F}_1)$ and $(\kappa_2, \text{spec}_2, \mathcal{F}_2)$.

Parallel Composition $C_1 \otimes C_2$ is a fault-tolerant gadget for $\mathcal{G}_1 \otimes \mathcal{G}_2$, with respect to the data $(\kappa_1 \sqcup \kappa_2, \text{spec}_1 \sqcup \text{spec}_2, \mathcal{F}_1 \boxplus \mathcal{F}_2)$, where $\kappa_1 \sqcup \kappa_2$ is the disjoint union of the two bundling and $\text{spec}_1 \sqcup \text{spec}_2$ is the disjoint union of the two specifications. Additionally, if C_1 and C_2 are both friendly, then $C_1 \otimes C_2$ is friendly.

Sequential Composition Suppose there is a bijection between the output bundled edges of C_1 and the input bundled edges of C_2 such that $\text{spec}_1, \text{spec}_2$ assigns the same code type to edges paired in bijection. Define $C_2 \circ C_1$ and $\mathcal{G}_2 \circ \mathcal{G}_1$ by joining the bundled edges according to the bijection. Then $C_2 \circ C_1$ is a fault-tolerant gadget for $\mathcal{G}_2 \circ \mathcal{G}_1$ with respect to the data $(\kappa_1 \cup \kappa_2, \text{spec}_1 \cup \text{spec}_2, \mathcal{F}_1 \boxplus \mathcal{F}_2)$, where the bundling $\kappa_1 \cup \kappa_2$ and specification $\text{spec}_1 \cup \text{spec}_2$ are the (overlapping) union of the bundlings and specifications induced by the bijection. Additionally, if C_1 is friendly, then $C_2 \circ C_1$ is friendly.

Proof. For parallel composition, recall that in the definition of a gadget the properties hold with respect to an arbitrary environment circuit. For a Pauli fault \mathbf{f} supported on the locations of $C_1 \otimes C_2$, it can be decomposed as¹¹ $\mathbf{f} = \mathbf{f}_1 \otimes \mathbf{f}_2$ where \mathbf{f}_1 is supported on the locations of C_1 and \mathbf{f}_2 is supported on the locations of C_2 . Then

$$\text{map}[C_1 \otimes C_2[\mathbf{f}]] = \text{map}[C_1[\mathbf{f}_1]] \otimes \text{map}[C_2[\mathbf{f}_2]] \quad (4.20)$$

and our claim follows by applying Eq. (4.15) independently to both circuits, since \mathbf{f}_1 is \mathcal{F}_1 -avoiding and \mathbf{f}_2 is \mathcal{F}_2 -avoiding.

For sequential composition, again consider a Pauli fault \mathbf{f} and its decomposition $\mathbf{f} = \mathbf{f}_1 \otimes \mathbf{f}_2$.

$$\text{map}[C_2 \circ C_1[\mathbf{f}]] = \text{map}[C_2[\mathbf{f}_2]] \circ \text{map}[C_1[\mathbf{f}_1]] \quad (4.21)$$

Let C_{env} be circuit with computational circuit $C_2 \circ C_1$ and an arbitrary environment circuit C_Ω . We can write

$$\text{map}[C_{\text{env}}] = \text{map}[C_2 \circ C_1] \otimes \text{map}[C_\Omega] \quad (4.22)$$

$$= \text{map}[C_2 \otimes I_\Omega] \circ \text{map}[C_1 \otimes C_\Omega]. \quad (4.23)$$

$$\text{map}[C_{\text{env}}[\mathbf{f}]] = \text{map}[C_2 \otimes I_\Omega[\mathbf{f}_2]] \circ \text{map}[C_1 \otimes C_\Omega[\mathbf{f}_1]]. \quad (4.24)$$

¹¹In particular, there is a fault \mathbf{f}' that does not contract any locations that is equivalent to \mathbf{f} in the sense that $\text{map}[C_1 \otimes C_2[\mathbf{f}]] = \text{map}[C_1 \otimes C_2[\mathbf{f}']]$.

Apply Eq. (4.15) to $C_2 \otimes I_\Omega$ and $C_1 \otimes C_\Omega$, we see that

$$\text{map}[C_2 \otimes I_\Omega[\mathbf{f}_2]] = (\tilde{\mathcal{G}}_{2,\mathbf{f}_2} \otimes I_\Omega) \circ (\mathcal{R}_{\mathbf{f}_2} \otimes I_\Omega), \quad (4.25)$$

$$\text{map}[C_1 \otimes C_\Omega[\mathbf{f}_1]] = (\tilde{\mathcal{G}}_{1,\mathbf{f}_1} \otimes \text{map}[C_\Omega]) \circ (\mathcal{R}_{\mathbf{f}_1} \otimes I_\Omega). \quad (4.26)$$

Here $\mathcal{R}_{\mathbf{f}_1}, \mathcal{R}_{\mathbf{f}_2}$ will be identities if their respective gadgets are not friendly. Putting things together, we see that

$$\text{map}[C_{\text{env}}[\mathbf{f}]] = (\tilde{\mathcal{G}}_{2,\mathbf{f}_2} \otimes I_\Omega) \circ (\mathcal{R}_{\mathbf{f}_2} \otimes I_\Omega) \circ (\tilde{\mathcal{G}}_{1,\mathbf{f}_1} \otimes \text{map}[C_\Omega]) \circ (\mathcal{R}_{\mathbf{f}_1} \otimes I_\Omega) \quad (4.27)$$

$$= ((\tilde{\mathcal{G}}_{2,\mathbf{f}_2} \circ \mathcal{R}_{\mathbf{f}_2} \circ \tilde{\mathcal{G}}_{1,\mathbf{f}_1}) \otimes \text{map}[C_\Omega]) \circ (\mathcal{R}_{\mathbf{f}_1} \otimes I_\Omega). \quad (4.28)$$

Note that the superoperator $\tilde{\mathcal{G}}_{2,\mathbf{f}_2} \circ \mathcal{R}_{\mathbf{f}_2} \circ \tilde{\mathcal{G}}_{1,\mathbf{f}_1}$ satisfies the simulation property Eq. (4.19) for $\mathcal{G}_2 \circ \mathcal{G}_1$ and our conclusion follows. \square

4.2 Decoupling of faults

When a gadget is faulty, anything can go wrong. In particular, the logical output state may depend on the input error in a highly non-trivial way. Informally, we think of noisily encoded states as being reversibly decoded to a state supported on two subsystems: the “logical” subsystem and the “syndrome” subsystem. For non-faulty gadgets, the syndrome subsystem remains decoupled from the logical subsystem much like the environment circuit. However, when a gadget fails, the fault is permitted to access the syndrome subsystem (since, on the encoded level, the resulting state is not independent of the syndrome).¹²

This decoupling will be critical to the proof of Theorem 4.11.

Lemma 4.10 (Decoupling of Faults). We use the variables defined in the definition of gadgets Definition 4.8. To emphasize the irrelevance of the environment circuit C_Ω , let $\text{renc}_C^{\text{out}}$ and $\text{rdec}_C^{\text{in}}$ refer to the reversible encoder and decoder of the computational circuit C (which is our fault-tolerant gadget). Let $\mathbf{t}_{\text{syn}}^{\text{in}}, \mathbf{t}_{\text{syn}}^{\text{out}}$ denote the input and output syndrome type (see Definition 4.2) of C . For any \mathcal{F} -avoiding Pauli fault \mathbf{f} on C_{env} supported only on the computational circuit, there exists a superoperator $\mathcal{N}_{\mathcal{G},\mathbf{f}} \in \mathcal{L}(\mathbf{t}_{\text{syn}}^{\text{out}}, \mathbf{t}_{\text{syn}}^{\text{in}})$ such that we can write the following decomposition.

$$\text{map}[C_{\text{env}}[\mathbf{f}]] = \left(\text{renc}_C^{\text{out}} \circ [\mathcal{G} \otimes \mathcal{N}_{\mathcal{G},\mathbf{f}}] \circ \text{rdec}_C^{\text{in}} \circ \mathcal{R}_{\mathbf{f}} \right) \otimes \text{map}[C_\Omega]. \quad (4.29)$$

We have drawn the corresponding circuit in Fig. 4. Additionally, there exists a superoperator $\mathcal{N}_{\mathcal{R},\mathbf{f}} \in \mathcal{L}(\mathbf{t}_{\text{syn}}^{\text{in}}, \mathbf{t}_{\text{syn}}^{\text{in}})$ such that for all good input states σ , we have

$$\mathcal{R}_{\mathbf{f}}(\sigma) = \text{renc}_C^{\text{in}} \circ (I_{\mathbf{t}_{\text{syn}}^{\text{in}}} \otimes \mathcal{N}_{\mathcal{R},\mathbf{f}}) \circ \text{rdec}_C^{\text{in}}(\sigma). \quad (4.30)$$

Proof. From equation (4.15), we have that

$$\text{map}[C_{\text{env}}[\mathbf{f}]] = \left(\tilde{\mathcal{G}}_{\mathbf{f}} \otimes \text{map}[C_\Omega] \right) \circ (\mathcal{R}_{\mathbf{f}} \otimes I_\Omega). \quad (4.31)$$

¹²Recall that faults may act on a contracted circuit.

Consider an arbitrary input state $\sigma_0 \in \mathsf{L}(\mathfrak{t}_{\bullet}^{\text{in}} \otimes \mathfrak{t}_{\Omega}^{\text{in}})$. By the friendly property of the gadget C , we have that $\sigma_1 = \mathcal{R}_{\mathbf{f}} \otimes I_{\Omega}(\sigma_0)$ is a good input state, which means there exists a logical state $\rho \in \mathsf{D}(\mathfrak{t}_L^{\text{in}} \otimes \mathfrak{t}_{\Omega}^{\text{in}})$ such that

$$\text{enc}^{\text{in}}(\rho) \lesssim_{\mathcal{B}^{\text{in}}} \sigma_1 \quad (4.32)$$

It suffices for us to show that there exists a superoperator $\mathcal{N}_{\mathcal{G},\mathbf{f}}$ such that for any such σ_1 ,

$$\left(\tilde{\mathcal{G}}_{\mathbf{f}} \otimes \text{map}[C_{\Omega}] \right) (\sigma_1) = \left(\left(\text{renc}_C^{\text{out}} \circ [\mathcal{G} \otimes \mathcal{N}_{\mathcal{G},\mathbf{f}}] \circ \text{rdec}_C^{\text{in}} \right) \otimes \text{map}[C_{\Omega}] \right) (\sigma_1). \quad (4.33)$$

Since the input and output code types on the environment circuit C_{Ω} is trivial, the encoders and decoders act trivially on the environmental inputs and outputs. Therefore, the above equation is equivalent to

$$\left(\left(\text{rdec}_C^{\text{out}} \circ \tilde{\mathcal{G}}_{\mathbf{f}} \circ \text{renc}_C^{\text{in}} \right) \otimes \text{map}[C_{\Omega}] \right) (\text{rdec}^{\text{in}}(\sigma_1)) = ([\mathcal{G} \otimes \mathcal{N}_{\mathcal{G},\mathbf{f}}] \otimes \text{map}[C_{\Omega}]) (\text{rdec}^{\text{in}}(\sigma_1)). \quad (4.34)$$

From Proposition 4.4, we see that $\text{rdec}^{\text{in}}(\sigma_1) = \rho \otimes v$ for some (not necessarily physical) state v satisfying the input syndrome type of C . Let M^{in} be the set of (not necessarily physical) syndrome states $v \in \mathsf{L}(\mathfrak{t}_{\text{syn}}^{\text{in}})$ such that

$$\text{enc}^{\text{in}}(\rho) \lesssim_{\mathcal{B}^{\text{in}}} \text{renc}^{\text{in}}(\rho \otimes v). \quad (4.35)$$

Similarly define M^{out} . Note that $M^{\text{in}}, M^{\text{out}}$ may not be linear subspaces of $\mathsf{L}(\mathfrak{t}_{\text{syn}}^{\text{in}})$ and $\mathsf{L}(\mathfrak{t}_{\text{syn}}^{\text{out}})$, respectively. However, the compositions of maps in Eq. (4.34) are linear. Therefore, let M be a basis of $\text{span}(M^{\text{in}})$. From the simulation property of the gadget C , we see that for any $v \in M$, there exists $v_{\mathbf{f}} \in M^{\text{out}}$ such that for all states ρ of the input logical type of C , we have

$$\left(\left(\text{rdec}_C^{\text{out}} \circ \tilde{\mathcal{G}}_{\mathbf{f}} \circ \text{renc}_C^{\text{in}} \right) \otimes \text{map}[C_{\Omega}] \right) (\rho \otimes v) = (\mathcal{G} \otimes \text{map}[C_{\Omega}]) (\rho) \otimes v_{\mathbf{f}}. \quad (4.36)$$

Define $\mathcal{N}_{\mathcal{G},\mathbf{f}} : \mathsf{L}(\mathfrak{t}_{\text{syn}}^{\text{in}}) \rightarrow \mathsf{L}(\mathfrak{t}_{\text{syn}}^{\text{out}})$ by $\mathcal{N}_{\mathcal{G},\mathbf{f}}(v) = v_{\mathbf{f}}$. Then $\mathcal{N}_{\mathcal{G},\mathbf{f}}$ is a valid linear map defined on $\text{span}(M^{\text{in}})$. Extend $\mathcal{N}_{\mathcal{G},\mathbf{f}}$ linearly and arbitrarily onto the rest of the domain $\mathsf{L}(\mathfrak{t}_{\text{syn}}^{\text{in}})$. We have that for all good input states σ_1 , Eq. (4.34) holds as desired. Applying the same arguments to $\mathcal{R}_{\mathbf{f}}$, we can construct $\mathcal{N}_{\mathcal{R},\mathbf{f}}$ for Eq. (4.30). \square

Later on, in the proof of level reduction, the syndrome subsystem will be moved to be part of the environment circuit.

4.3 Level reduction

We are now ready to prove an analog of “level reduction” from [AGP05] which represents the most challenging form of composition. Roughly, this allows reasoning about the simulated circuit whenever gadgets in the fault-tolerant circuit fail. The proof will crucially use the friendliness of the gadgets and the environment circuit.

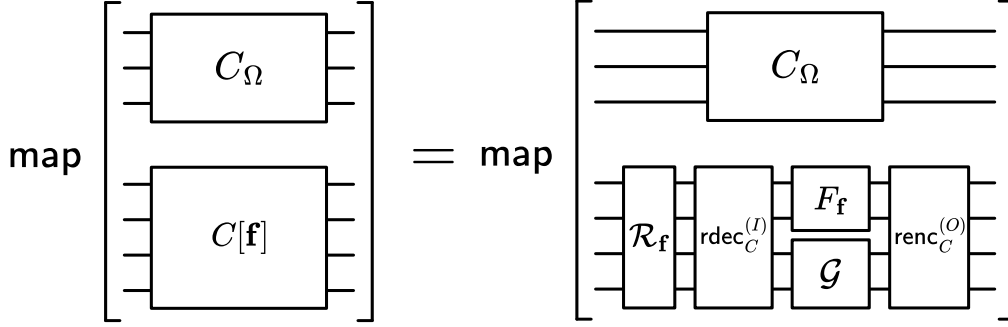


Figure 4: Pictorial representation of the decoupling lemma (Lemma 4.10). Whenever C accepts or outputs multiple wire bundles, $\mathcal{N}_{\mathcal{G},\mathbf{f}}$ acts on all syndrome subsystems and \mathcal{G} acts on all logical subsystems.

Theorem 4.11 (Level reduction). Fix a circuit $C = (V, E, \text{conn}, \text{type}, \text{gate})$ (potentially with environment) and a contraction map $\kappa: V \rightarrow V_\kappa$ such that every location in $W = V_\kappa$ of the contracted circuit $C_\kappa = (W, E_\kappa, \text{conn}_\kappa, \text{type}_\kappa, \text{gate}_\kappa)$ corresponds to a gadget $w \in W$. Assign a bundling $\beta: E \rightarrow E_\beta$ to the edges of C_κ compatible with the bundling of each gadget and a code specification spec such that each gadget $w \in W$ simulates the gate \mathcal{G}_w with input and output code types^a given by the global spec spec and bad Pauli faults $\mathcal{F}_w \subseteq P(\kappa^{-1}(w))$. We further require that: 1) The input and output types of the circuit (with respect to spec) are trivial (see Definition 4.2). 2) All gadgets $w \in W$ connected to the output have no bad fault paths $\mathcal{F}_w = \{\}$ e.g. are classical decoders (classical locations do not have faults).^b

Consider also the logical circuit $C_L = (W, E_L, \text{conn}_L, \text{type}_L, \text{gate}_L)$ where edge bundle $e_\beta \in E_\beta$ of C_κ is replaced by an edge of the corresponding logical type (given by spec) and, for every gadget $w \in W$ of C_κ , the gate $\text{gate}_\kappa(w)$ is replaced by the gate that the gadget simulates $\text{gate}_L(w) := \mathcal{G}_w$.

Now fix a Pauli fault \mathbf{f} supported on C and a family of bad fault paths $\mathcal{W} \subseteq P(W)$. If \mathbf{f} is $\mathcal{W} \bullet \{\mathcal{F}_w\}_{w \in W}$ -avoiding, then, for some \mathcal{W} -avoiding fault^c \mathbf{f}' supported on $C_{L,\Omega}$ (C_L with environment) we have that

$$\text{map}[C[\mathbf{f}]] = \text{map}[C_{L,\Omega}[\mathbf{f}']] \quad (4.37)$$

^aActually, they need only be compatible with the code type in the sense that the code type assigned to the wire bundle may be a stronger set of bad error supports.

^bThis is possible for circuits with classical input and output in the model of computation where classical circuits do not have faults. More generally, the corresponding statement requires encoders on the input and decoders on the output.

^c \mathbf{f}' is not Pauli in general.

Proof. Let $U \subseteq W$ be the set of “good” gadgets where, for each gadget $w \in U$, the fault \mathbf{f} is \mathcal{F}_w -avoiding on the locations of the gadget $\kappa^{-1}(w)$. We will construct $C_{L,\Omega}[\mathbf{f}']$ by a sequence of circuit rewrites of $C_\kappa[\mathbf{f}]$.

The first step will be to rewrite the gates of $C_\kappa[\mathbf{f}]$ into a new gate map $\text{gate}_\kappa^{(1)}$. First note that, because \mathbf{f} is Pauli, we can take the vertex contraction to be trivial. Thus, in what follows, C may have an environment circuit, but we do not need to explicitly work with it. Let $\text{gate}_\kappa(w)$ be the gate map of $C_\kappa[\mathbf{f}]$.

For each gadget $w \in W$, if the gadget is good $w \in U$, then the support of the fault \mathbf{f} is \mathcal{F}_w -avoiding and can we apply the decomposition (Eq. (4.29) and Eq. (4.18)) of the decoupling lemma Lemma 4.10. For input edge bundles (b_1, \dots, b_{ℓ_w}) of w , the faulty gadget map has the decomposition

$$\widetilde{\text{gate}}_\kappa(w) = \text{renc}_w^{\text{out}} \circ [\mathcal{G}_w \otimes \mathcal{N}_{\mathcal{G}, \mathbf{f}, w}] \circ \text{rdec}_w^{\text{in}} \circ (\mathcal{R}_1^{\text{in}} \otimes \dots \otimes \mathcal{R}_{\ell_w}^{\text{in}}). \quad (4.38)$$

We will “push” the superoperators \mathcal{R}_i to the preceding gadget. More precisely, consider a gadget $w \in W$ with m_w output bundles (b_1, \dots, b_{m_w}) . For $i \in [m_w]$, let $u_i \in W$ be the gadget associated with $\text{dest}(b_i)$. If u_i is good $u_i \in U$, then define $\mathcal{T}_{w,i}^{\text{out}}$ to be the corresponding input bundle filter in the decomposition Eq. (4.38) at u_i . Otherwise, we take $\mathcal{T}_{w,i}^{\text{out}}$ to be the identity. We can now write the modified gate map $\text{gate}_\kappa^{(1)}$.

$$\text{gate}_\kappa^{(1)}(w) = \begin{cases} (\mathcal{T}_{w,1}^{\text{out}} \otimes \dots \otimes \mathcal{T}_{w,m_w}^{\text{out}}) \circ \widetilde{\text{gate}}_\kappa(w) & w \notin U \\ (\mathcal{T}_{w,1}^{\text{out}} \otimes \dots \otimes \mathcal{T}_{w,m_w}^{\text{out}}) \circ \text{renc}_w^{\text{out}} \circ [\mathcal{G}_w \otimes \mathcal{N}_{\mathcal{G}, \mathbf{f}, w}] \circ \text{rdec}_w^{\text{in}} & w \in U \end{cases} \quad (4.39)$$

By construction, the contracted circuit $C_\kappa^{(1)} = (W, E_\kappa, \text{conn}_\kappa, \text{type}_\kappa, \text{gate}_\kappa^{(1)})$ with the modified gate map is equivalent to $C_\kappa[\mathbf{f}]$:

$$\text{map}[C_\kappa^{(1)}] = \text{map}[C_\kappa[\mathbf{f}]]. \quad (4.40)$$

In the next step, we will strip off the filter superoperators where we are able to. For each gadget $w \in W$, let $\mathcal{B}_w^{\text{in}}$ and $\mathcal{B}_w^{\text{out}}$ be the input and output bad error supports of the gadget w (induced by spec). Consider the action of $\text{map}[C_\kappa]$ on an arbitrary input state ρ_0 . Let ρ_t , $t \in [|W|]$ be the state after applying the gate maps of gadgets in $(1, \dots, t)$ (in topological order) to ρ_0 .

For each $t \in [|W|]$, the direct ancestors u of w_t have the filter superoperators $(\mathcal{T}_{u,1}^{\text{out}} \otimes \dots \otimes \mathcal{T}_{u,m_u}^{\text{out}})$, so there is a logical state σ_{t-1} such that ρ_{t-1} is $\mathcal{B}_{w_t}^{\text{in}}$ -deviated from $\text{enc}_{w_t}^{\text{in}}(\sigma_{t-1})$ ¹³. Thus, if $w_t \in U$ is good, using the simulation definition (Definition 4.8), it follows that ρ_t is $\mathcal{B}_{w_t}^{\text{out}}$ -deviated from $\text{enc}_{w_t}^{\text{out}} \circ (\mathcal{G}_{w_t} \otimes \mathcal{N}_{\mathcal{G}, \mathbf{f}, w_t})(\sigma_{t-1})$. From Eq. (4.30) of Lemma 4.10, we see that on such a good logical state, the filter can be decomposed as

$$\forall i \in [m_{w_t}], \mathcal{T}_{w_t,i}^{\text{out}} = \text{renc}_{w_t}^{\text{out}} \circ [I \otimes \mathcal{N}_{\mathcal{T}, \mathbf{f}, w_t, i}] \circ \text{rdec}_{w_t}^{\text{out}}. \quad (4.41)$$

In other words, the filter superoperators at w_t (if present) act as identity on the logical state. We can therefore absorb the filter superoperators into the fault superoperator supported on the syndrome system at each good gadget. Denote

$$\mathcal{N}_{\mathcal{T}, \mathbf{f}, w_t} := \bigotimes_{i=1}^{m_{w_t}} \mathcal{N}_{\mathcal{T}, \mathbf{f}, w_t, i} \quad \mathcal{N}_{\mathbf{f}, w_t} := \mathcal{N}_{\mathcal{T}, \mathbf{f}, w_t} \circ \mathcal{N}_{\mathcal{G}, \mathbf{f}, w_t} \quad (4.42)$$

¹³Here the input encoder of w_t is defined by spec extended by identity for the subsystems not in the support of $\text{gate}_\kappa(w_t)$

we can define another modified gate map

$$\text{gate}_\kappa^{(2)}(w) = \begin{cases} (\mathcal{T}_{w,1}^{\text{out}} \otimes \dots \otimes \mathcal{T}_{w,m_w}^{\text{out}}) \circ \widetilde{\text{gate}}_\kappa(w) & w \notin U \\ \text{renc}_w^{\text{out}} \circ [\mathcal{G}_w \otimes \mathcal{N}_{\mathbf{f},w}] \circ \text{rdec}_w^{\text{in}} & w \in U \end{cases} \quad (4.43)$$

By induction on the circuit locations in reverse topological order, and noting that the input code type is trivial so the base case is satisfied, we see that this rewrite of gate map results in an equivalent circuit $C_\kappa^{(2)} = (W, E_\kappa, \text{conn}_\kappa, \text{type}_\kappa, \text{gate}_\kappa^{(2)})$.

$$\text{map}[C_\kappa^{(2)}] = \text{map}[C_\kappa^{(1)}] \quad (4.44)$$

We will now unencode every edge bundle of $C_\kappa^{(2)}$ by inserting a reversible encoder/decoder pair for each edge bundle b . This replaces the edges of the edge bundle by two edges, one with the logical type of b and one with the syndrome type of b . The reversible encoders and decoders cancel the reversible decoders and encoders of the good gadgets.

$$\text{gate}_\kappa^{(3)}(w) = \begin{cases} \text{rdec}_w^{\text{out}} \circ (\mathcal{T}_{w,1}^{\text{out}} \otimes \dots \otimes \mathcal{T}_{w,m_w}^{\text{out}}) \circ \widetilde{\text{gate}}_\kappa(w) \circ \text{renc}_w^{\text{in}} & w \notin U \\ \mathcal{G}_w \otimes \mathcal{N}_{\mathbf{f},w} & w \in U \end{cases} \quad (4.45)$$

We have only inserted pairs of unitaries and their inverses.

$$\text{map}[C_\kappa^{(3)}] = \text{map}[C_\kappa^{(2)}] \quad (4.46)$$

Finally, we can construct $C_{L,\varepsilon}[\mathbf{f}']$ by “splitting” all good gadgets of $C_\kappa^{(3)}$: Every good gadget w has a gate $\text{gate}_\kappa^{(3)}(w) = \mathcal{G}_w \otimes \mathcal{N}_{\mathbf{f},w}$ that factorizes into a superoperator supported only on the logical subsystem \mathcal{G}_w and a superoperator supported only on the syndrome subsystem $\mathcal{N}_{\mathbf{f},w}$.

We now equip C_L with an environment circuit C_Ω with network matching that of C_L . That is, for every vertex w of C_L , we add a vertex $w^{(s)}$ to the environment circuit. The edge types are the syndrome types of each bundle of C_κ (which are in bijection with edges of C_L) and the gate map is arbitrary. The fault \mathbf{f}' contracts $w^{(s)}$ with w whenever $w \notin U$ is not good. More precisely, let $\widetilde{\text{gate}}_L$ be the gate map of $C_{L,\varepsilon}[\mathbf{f}']$.

$$w \in U \quad \widetilde{\text{gate}}_L(w) = \mathcal{G}_w \quad (4.47)$$

$$\widetilde{\text{gate}}_L(w^{(s)}) = \mathcal{N}_{\mathbf{f},w} \quad (4.48)$$

$$w \notin U \quad \widetilde{\text{gate}}_L(\{w, w^{(s)}\}) = \text{rdec}_w^{\text{out}} \circ (\mathcal{T}_{w,1}^{\text{out}} \otimes \dots \otimes \mathcal{T}_{w,m_w}^{\text{out}}) \circ \widetilde{\text{gate}}_\kappa(w) \circ \text{renc}_w^{\text{in}} \quad (4.49)$$

Where we use the abuse of notation $\{w, w^{(s)}\}$ to refer to the vertex under the vertex contraction of the fault. This is a rearrangement of $C_\kappa^{(3)}$, so

$$\text{map}[C_{L,\varepsilon}[\mathbf{f}']] = \text{map}[C_\kappa^{(3)}] = \text{map}[C_\kappa^{(2)}] = \text{map}[C_\kappa^{(1)}] = \text{map}[C_\kappa[\mathbf{f}]] . \quad \square$$

4.4 Miscellaneous tools

We now include two miscellaneous tools that are generally useful. The first is the decomposition of faults into Pauli faults.

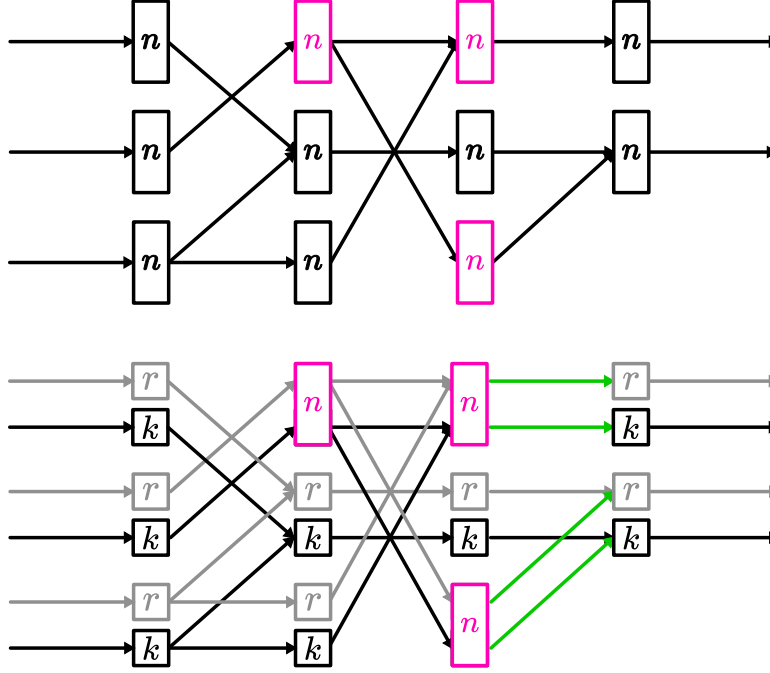


Figure 5: Pictorial transformation from $C_\kappa[\mathbf{f}]$ to $C_{L,\Omega}[\mathbf{f}']$. The set of faulty gadgets U is highlighted magenta. Every vertex (not in U) in the transformed circuit (lower) is paired with an environment vertex. For clarity, connections between environment and computational vertex pairs are drawn as doubled arrows. Green highlighted connections correspond to “pushing” a filter map \mathcal{R}_i to part of the fault on the previous vertex (see item 1 of case 1).

Lemma 4.12 (Decomposition of faults). For any circuit C and fault \mathbf{f} , there exists a new circuit with environment C_{env} and set of Pauli faults $\{\mathbf{f}'_i\}_i$ such that:

- C_{env} has computational circuit C and a closed environment circuit C_Ω (see Definition 3.17).
- The map of $C[\mathbf{f}]$ is a linear combination of maps of C_{env} subject to the faults \mathbf{f}'_i .

$$\text{map}[C[\mathbf{f}]] = \sum_i \text{map}[C_{\text{env}}[\mathbf{f}'_i]] \quad (4.50)$$

Note that since C_Ω is closed, every $\text{map}[C_{\text{env}}[\mathbf{f}'_i]]$ is a superoperator with the same input and output type as C and $C[\mathbf{f}]$.

- Furthermore, for each i , the support of each fault \mathbf{f}'_i on the computational circuit is a subset of the support of the original fault \mathbf{f} .

Proof. Note that general fault could replace gates in a circuit, while Pauli faults (Definition 3.14) are inserted before and after the intended gates. Therefore, the main argument for this proposition is to construct an environment circuit to help rewrite general faults on the computational circuit into Pauli faults.

Denote the circuit as $C = (V, E, \text{conn}, \text{type}, \text{gate})$, and let the fault be $\mathbf{f} = (\kappa, \text{gate}_\kappa)$, where $\kappa : V \rightarrow V_\kappa$ is a contraction map and $\text{gate}_\kappa(v)$ for $v \in V_\kappa$ are faulty gates. We construct C_{env} as follows. Create two independent copies of C , namely C_1, C_2 with corresponding data. Apply the fault \mathbf{f} to C_2 , by taking $\kappa_2 : V_2 \rightarrow V_{2,\kappa}$ and $\text{gate}_{2,\kappa}$ to be a labeling on $V_{2,\kappa}$. Then C_{env} is our circuit with environment, where C_1 is the computational circuit and $C_{2,\kappa} = (V_{2,\kappa}, E_{2,\kappa}, \text{conn}_{2,\kappa}, \text{type}_{2,\kappa}, \text{gate}_{2,\kappa})$ is the environment circuit.

To satisfy Eq. (4.50), we add, as a new fault \mathbf{f}' , a collection of SWAP gates between C_1 and $C_{2,\kappa}$, and later decompose these SWAP gates into Pauli faults. We define the new fault $\mathbf{f}' = (\tau, \text{gate}_\tau)$ as follows. For a vertex $v \in V$, denote its copies in V_1, V_2 as $v[1], v[2]$, respectively. Precisely, for every $v \in V$ such that $\text{gate}_{2,\kappa}(\kappa(v[2])) \neq \text{gate}_2(v[2])$, the set of vertices contracted to $\kappa(v[2])$ is $\kappa^{-1}(\kappa(v[2]))$, and the corresponding vertices in V_1 is

$$T_v := \{u[1] : u[2] \in \kappa^{-1}(\kappa(v[2]))\}. \quad (4.51)$$

Define

$$\text{in}(T_v) = \bigcup_{u[1] \in T_v} \text{in}(u[1]), \quad (4.52)$$

and similarly define $\text{out}(T_v)$. There is a natural bijection $\phi_{\text{in}} : \text{in}(T_v) \rightarrow \text{in}(\kappa(v[2]))$, induced by the bijection between T_v and the vertices contracted to $\kappa(v[2])$. Similarly there is a natural bijection $\phi_{\text{out}} : \text{out}(T_v) \rightarrow \text{out}(\kappa(v[2]))$. We define τ to contract $\kappa(v[2])$ and T_v , and define the gate on this contracted vertex to be

$$\text{gate}_\tau(\tau(T_v \cup \{\kappa(v[2])\})) = \quad (4.53)$$

$$\left(\bigotimes_{e \in \text{out}(T_v)} \text{SWAP}[e, \phi_{\text{out}}(e)] \right) \circ \left(\text{gate}_{2,\kappa}(\kappa(v[2])) \otimes \bigotimes_{u[1] \in T_v} \text{gate}_1(u[1]) \right) \circ \left(\bigotimes_{e \in \text{in}(T_v)} \text{SWAP}[e, \phi_{\text{in}}(e)] \right) \quad (4.54)$$

This gate is conceptually simple: we **SWAP** all input edges of vertices in T_v with input edges of $\kappa(v[2])$, perform the same gates as in C_{env} (which performs a faulty gate on $\kappa(v[2])$ as specified by \mathbf{f} , and performs the ideal circuit gates as in C), and then **SWAP** the output edges again. This completes our description of $\mathbf{f}' = (\tau, \text{gate}_\tau)$.

Now observe that the computational circuit of C_{env} is a copy of C , while the environment circuit C_Ω is a copy of $C[\mathbf{f}]$. We finish the construction of C_{env} by initializing¹⁴ all input edges of C_Ω to $|0\rangle$ and tracing out all output edges of C_Ω , thereby closing C_Ω . By adding the new fault \mathbf{f}' , we have **SWAPed** the faulty gates from the environment circuit into our computational circuit. This gives us

$$\text{map}[C[\mathbf{f}]] = \text{map}[C_{\text{env}}[\mathbf{f}']]. \quad (4.55)$$

We observe that the fault path of \mathbf{f}' on the computational circuit of C_{env} is in bijection with the fault path of \mathbf{f} on C . Moreover, the fault \mathbf{f}' conjugates the existing gates in C_{env} by **SWAP** gates, which can be decomposed into non-diagonal Pauli faults. This proves our claim. \square

We will also use the following lemma, which is a statement of the standard fact that stabilizer measurements decoheres general error into Pauli errors. A proof was given in [NP24], we include it here for completeness.

Definition 4.13 (Recoverable Set). For a stabilizer code \mathcal{Q} on n physical qubits, a subset of qubits $S \subseteq [n]$ is *recoverable* if there are no logical operator of \mathcal{Q} which is fully supported in S .

Lemma 4.14 (Decoherence of errors [NP24]). For a n -qubit stabilizer code \mathcal{Q} , let \mathcal{M} be the channel that measures the r stabilizer checks of \mathcal{Q} and outputs the measurement outcomes. For a superoperator \mathcal{E}_B supported on a recoverable subset of qubits $B \subseteq [n]$ and a codestate ρ of \mathcal{Q} , the application of the noise superoperator followed by measurement of the checks can be written as

$$\mathcal{M} \circ \mathcal{E}_B(\rho) = \sum_{s \in \mathbb{F}_2^r} \alpha_s |s\rangle\langle s| \otimes E_s \rho E_s, \quad (4.56)$$

where each E_s is a Pauli operator supported on B and the α_s are complex coefficients. When \mathcal{E}_B is a physical noise channel, they satisfy $\sum_s \alpha_s = 1$. In other words, measurement of the checks collapses the error into a single Pauli error *as long as we remember the measurement outcome*.

Proof. Let $\{S_i\}_{i \in [r]}$ be a basis of the stabilizer group of \mathcal{Q} . For a syndrome $s \in \mathbb{F}_2^r$, we define the projector to the syndrome space corresponding to s as $\Pi_s = \prod_r (\frac{1}{2}I + (-1)^{s_i} S_i)$. Then the measurement channel \mathcal{M} can be expressed as

$$\mathcal{M}(\rho) = \sum_{s \in \mathbb{F}_2^r} |s\rangle\langle s| \otimes \Pi_s \rho \Pi_s. \quad (4.57)$$

¹⁴This choice of initialization is completely arbitrary.

Consider the Kraus decomposition of \mathcal{E}_B in terms of Pauli operators $\{K_\mu\}_\mu, \{K'_\nu\}_\nu$ supported on B and complex coefficients $\{\alpha_{\mu,\nu}\}_{\mu,\nu}$.

$$\mathcal{M} \circ \mathcal{E}_B(\rho) = \sum_{s \in \mathbb{F}_2^r} |s\rangle\langle s| \otimes \Pi_s \left(\sum_{\mu,\nu} \alpha_{\mu,\nu} K_\mu \rho K'_\nu \right) \Pi_s \quad (4.58)$$

$$= \sum_{s \in \mathbb{F}_2^r} |s\rangle\langle s| \otimes \left(\sum_{\mu,\nu} \alpha_{\mu,\nu} \Pi_s K_\mu \rho K'_\nu \Pi_s \right) \quad (4.59)$$

$$= \sum_{s \in \mathbb{F}_2^r} |s\rangle\langle s| \otimes \left(\sum_{\mu,\nu} \alpha_{\mu,\nu} \Pi_s K_\mu \Pi_0 \rho \Pi_0 K'_\nu \Pi_s \right) \quad (4.60)$$

We see that the projectors Π_s annihilate all terms except those where K_μ, K'_ν both have syndrome s . Moreover, since B is a recoverable set, any two Pauli operators supported on B which has the same syndrome are equivalent up to stabilizers. Therefore, for every $s \in \mathbb{F}_2^r$, there exists Pauli operator E_s supported on B which is equivalent up to stabilizers to all K_μ, K'_ν with syndrome s . In other words, $\Pi_s E_s \Pi_0 = \Pi_s K_\mu \Pi_0$ for such μ, ν . Let α_s be the sum of all $\alpha_{\mu,\nu}$ where K_μ, K'_ν have syndrome s . We can write

$$\mathcal{M} \circ \mathcal{E}_B(\rho) = \sum_{s \in \mathbb{F}_2^r} |s\rangle\langle s| \otimes \left(\sum_{\mu,\nu} \alpha_{\mu,\nu} \Pi_s K_\mu \Pi_0 \rho \Pi_0 K'_\nu \Pi_s \right) \quad (4.61)$$

$$= \sum_{s \in \mathbb{F}_2^r} \alpha_s |s\rangle\langle s| \otimes \Pi_s E_s \Pi_0 \rho \Pi_0 E_s \Pi_s \quad (4.62)$$

$$= \sum_{s \in \mathbb{F}_2^r} \alpha_s |s\rangle\langle s| \otimes E_s \rho E_s. \quad \square$$

5 Circuit correctness

Here we prove that under various noise models, the total variation distance between the output distribution of the simulated circuit and that of the fault-tolerant circuit can be bounded in terms of the evaluation of the weight enumerator for various noise models.

In what follows, let C be a classical-quantum circuit with classical input and classical output. Note that $\text{map}[C]$ is a map from bitstrings to distributions.

Let $C_{\text{FT}} = (V, E, \text{conn}, \text{type}, \text{gate})$ be a gadget for C with trivial input and output types and bad fault paths \mathcal{F} (e.g. obtained from Theorem 4.11 with $\mathcal{W} = \{\{v\} \mid v \in V\}$). In what follows, in order to obtain standard threshold results, think of C_{FT} as being an element of a family indexed by some element n such that below some critical value $x \in [0, \epsilon_*)$, the evaluation of the weight enumerator can be upper bounded e.g. $\mathcal{W}(\mathcal{F}; x) \leq e^{-f(n)}$ for some positive function f . Thus, a “universal” threshold theorem simply provides such a family of mappings from circuits to gadgets with upper bounds on the weight enumerators.

5.1 Adversarial noise

Our first noise model is simply an adversarial one. In this model, an arbitrarily powerful adversary is permitted an arbitrarily large quantum memory and access to qubits supported on a set of locations of the circuit $S \subseteq V$ that is \mathcal{F} -avoiding. The adversary is permitted to replace each fault location with an arbitrary channel of the same type. This is far stronger than the capability to insert Pauli operators: They may, for example, “steal” qubits and insert them into the circuit at a later time.

Let $C_{\text{FT},\Omega}[\mathbf{f}]$ be the execution of the circuit C_{FT} in the presence of the adversary. Their interaction is captured by the introduction of an environment circuit and an \mathcal{F} -avoiding \mathbf{f} .

Proposition 5.1. Let $C_{\text{FT},\Omega}[\mathbf{f}]$ be the execution of C_{FT} in the presence of an arbitrary, closed environment circuit C_Ω and a \mathcal{F} -avoiding physical fault \mathbf{f} , such that $\text{map}[C_{\text{FT},\Omega}[\mathbf{f}]]$ is a physical CPTP channel. It holds that

$$\text{map}[C_{\text{FT},\Omega}[\mathbf{f}]] = \text{map}[C]. \quad (5.1)$$

Proof. We first apply Lemma 4.12 to write the map of the original circuit in terms of a sum over executions of circuit C_{FT} with a modified environment subject to Pauli faults $\{\mathbf{f}'_i\}_i$.

$$\text{map}[C_{\text{FT},\Omega}[\mathbf{f}]] = \sum_i \text{map}[C_{\text{FT},\Omega'}[\mathbf{f}'_i]]. \quad (5.2)$$

The Pauli faults are also \mathcal{F} -avoiding, so we can apply the definition of a fault-tolerant gadget (simulation property, Definition 4.8). For some complex coefficients $\{c_i\}_i$ we can write the following.

$$\sum_i \text{map}[C_{\text{FT},\Omega'}[\mathbf{f}'_i]] = \sum_i c_i \text{map}[C] = \text{map}[C]. \quad (5.3)$$

Where the last equality holds because $\text{map}[C_{\text{FT},\Omega}[\mathbf{f}]]$ is a physical (CPTP) channel (which is the case, for example, when each (faulty) gate in $C_{\text{FT},\Omega}[\mathbf{f}]$ is physical). \square

5.2 Local stochastic noise

Our next noise model is related to the adversarial stochastic noise mode of [AGP05]. It is the adversarial noise model of the previous section, but the support of the adversary is permitted to be a random variable that obeys a “locally stochastic” property.

Proposition 5.2. Let $C_{\text{FT},\Omega}[\mathbf{f}]$ be the execution of C_{FT} in the presence of an arbitrary, closed environment circuit and a physical fault \mathbf{f} distributed according to some distribution M such that, for all subsets $S \subseteq V$ of locations of the computational circuit, $\Pr_M(S \subseteq \text{supp } \mathbf{f}) \leq \epsilon^{|S|}$. Then, for any input bitstring x , the total variation distance of the output distributions (equivalently, trace distance of the corresponding density operators) is at most $\epsilon_L = \mathcal{W}(\mathcal{F}; \epsilon)$. That is, denoting the

probability density of \mathbf{f} by P_M ,

$$\frac{1}{2} \operatorname{tr} \left| \operatorname{map}[C] (|x\rangle\langle x|) - \left(\sum_{\mathbf{f}} P_M(\mathbf{f}) \operatorname{map}[C_{\text{FT},\Omega}[\mathbf{f}]] (|x\rangle\langle x|) \right) \right| \leq \mathcal{W}(\mathcal{F}; \epsilon). \quad (5.4)$$

Proof. We first compute the probability that the support of \mathbf{f} is not \mathcal{F} -avoiding.

$$\Pr_M(\operatorname{supp} \mathbf{f} \text{ is not } \mathcal{F}\text{-avoiding}) \leq \sum_{S \in \mathcal{F}} \Pr_M(S \subseteq \operatorname{supp} \mathbf{f}) \leq \sum_{S \in \mathcal{F}} \epsilon^{|S|} = \mathcal{W}(\mathcal{F}; \epsilon). \quad (5.5)$$

Let \bar{E} be this subset of events. Whenever the fault is not in the set, the two density matrices are equal by Proposition 5.1. Otherwise, the difference may be at most 2 by normalization.

$$\frac{1}{2} \operatorname{tr} \left| \operatorname{map}[C] (|x\rangle\langle x|) - \left(\sum_{\mathbf{f}} P_M(\mathbf{f}) \operatorname{map}[C_{\text{FT},\Omega}[\mathbf{f}]] (|x\rangle\langle x|) \right) \right| \quad (5.6)$$

$$\leq \frac{1}{2} \sum_{\mathbf{f}} P_M(\mathbf{f}) \operatorname{tr} |\operatorname{map}[C] (|x\rangle\langle x|) - \operatorname{map}[C_{\text{FT},\Omega}[\mathbf{f}]] (|x\rangle\langle x|)| \quad (5.7)$$

$$= \frac{1}{2} \sum_{\mathbf{f} \in \bar{E}} P_M(\mathbf{f}) \operatorname{tr} |\operatorname{map}[C] (|x\rangle\langle x|) - \operatorname{map}[C_{\text{FT},\Omega}[\mathbf{f}]] (|x\rangle\langle x|)| \quad (5.8)$$

$$\leq \sum_{\mathbf{f} \in \bar{E}} P_M(\mathbf{f}) \leq \mathcal{W}(\mathcal{F}; \epsilon). \quad \square$$

In the above proposition, the statement holds for any input string x because we have assumed classical gates in C_{FT} to be noiseless. This assumption can be dropped if we incorporate classical fault-tolerance constructions.

5.3 Coherent noise

We can also bound against coherent noise in much the same way using a standard technique [ABO97; AGP05]. However, a more careful bound on the contribution from locations outside of the fault support is required.

Proposition 5.3. Let $C_{\text{FT}}[\mathbf{f}]$ be the execution of C_{FT} subject to a fault \mathbf{f} that does not contract any vertices and applies the correct gate of C_{FT} followed by an operation with diamond distance $\leq \epsilon$ from the identity. Additionally, suppose that

1. C_{FT} is a composition of gadgets,

$$\operatorname{map}[C_{\text{FT}}] = \operatorname{map}[C_m] \circ \operatorname{map}[C_{m-1}] \circ \cdots \circ \operatorname{map}[C_1], \quad (5.9)$$

and each gadget C_i has bad fault paths \mathcal{F}_i such that $\mathcal{F} = \boxplus_{i=1}^m \mathcal{F}_i$;

2. Let V_i denote the set of locations of C_i . There exists Γ such that for each $i \in [m]$, for every $F_i \in \mathcal{F}_i$, it holds that $|V_i| \leq \Gamma |F_i|$.

Set $\eta = \max_i \mathcal{W}(\mathcal{F}_i; 2(1+\epsilon)^{\Gamma-1}\epsilon)$. Then, for any input bitstring x , the trace distance of the corresponding density operators is

$$\frac{1}{2} \text{tr} |\text{map}[C](|x\rangle\langle x|) - \text{map}[C_{\text{FT}}[\mathbf{f}]](|x\rangle\langle x|)| \leq m(1+\eta)^m \eta. \quad (5.10)$$

Proof. Since the faults do not contract any locations and act on each gate independently, we can decompose the fault \mathbf{f} into $\mathbf{f}_1, \dots, \mathbf{f}_m$ supported on the locations in C_1, \dots, C_m .

$$\text{map}[C_{\text{FT}}[\mathbf{f}]] = \text{map}[C_m[\mathbf{f}_m]] \circ \dots \circ \text{map}[C_1[\mathbf{f}_1]] \quad (5.11)$$

We first analyze each individual gadget $C_i[\mathbf{f}_i]$. Let $\widetilde{\text{gate}}_i$ be the gate map of \mathbf{f}_i . For each $v \in V_i$, we can decompose

$$\widetilde{\text{gate}}_i(v) = (1-\epsilon)\text{gate}_i(v) + 2\epsilon\mathcal{N}_i(v), \quad \text{where } \|\mathcal{N}_i(v)\|_{\diamond} \leq 1. \quad (5.12)$$

For a subset $S \subseteq V_i$, let $\widetilde{\text{gate}}_{i,S}$ denote the gate map that takes the difference of the noisy and perfect gate whenever $v \in S$.

$$v \in V_i \quad \widetilde{\text{gate}}_{i,S}(v) = \begin{cases} 2\epsilon\mathcal{N}_i(v) & v \in S \\ (1-\epsilon)\text{gate}_i(v) & v \notin S \end{cases}. \quad (5.13)$$

Let $\tilde{\mathbf{f}}_{i,S}$ denote a fault with this gate map. Using the sub-multiplicativity of the diamond norm, we note that

$$\|\text{map}[C_i[\tilde{\mathbf{f}}_{i,S}]]\|_{\diamond} \leq (2\epsilon)^{|S|}(1-\epsilon)^{|V_i|-|S|} \quad (5.14)$$

since $\text{map}[C_i[\tilde{\mathbf{f}}_{i,S}]]$ contains $|S|$ operators each with diamond norm at most 2ϵ and the remaining operators each have diamond norm $1-\epsilon$. For simplicity of notation, let us call a set S good if it is \mathcal{F}_i -avoiding and bad otherwise. We can rewrite the execution of the noisy circuit in terms of these faults:

$$\text{map}[C_i[\mathbf{f}_i]] = \sum_{S \subseteq V_i} \text{map}[C_i[\tilde{\mathbf{f}}_{i,S}]] \quad (5.15)$$

$$= \sum_{\text{good } S \subseteq V_i} \text{map}[C_i[\tilde{\mathbf{f}}_{i,S}]] + \sum_{\text{bad } S \subseteq V_i} \text{map}[C_i[\tilde{\mathbf{f}}_{i,S}]]. \quad (5.16)$$

Let us denote

$$\mathcal{C}_{i,+} = \sum_{\text{good } S \subseteq V_i} \text{map}[C_i[\tilde{\mathbf{f}}_{i,S}]], \quad \mathcal{C}_{i,-} = \sum_{\text{bad } S \subseteq V_i} \text{map}[C_i[\tilde{\mathbf{f}}_{i,S}]]. \quad (5.17)$$

We can bound the total contribution of the ‘bad’ fault paths within a single gadget in terms of

diamond norm.

$$\|\mathcal{C}_{i,-}\|_{\diamond} = \left\| \sum_{\text{bad } S \subseteq V_i} \text{map}[C_i[\tilde{\mathbf{f}}_{i,S}]] \right\|_{\diamond} \quad (5.18)$$

$$\leq \sum_{F \in \mathcal{F}_i} \sum_{F \subseteq S \subseteq V_i} (2\epsilon)^{|F|} (2\epsilon)^{|S|-|F|} (1-\epsilon)^{|V_i|-|S|} \quad (5.19)$$

$$= \sum_{F \in \mathcal{F}_i} (2\epsilon)^{|F|} (1+\epsilon)^{|V_i|-|F|} \quad (5.20)$$

$$\leq \sum_{F \in \mathcal{F}_i} (2(1+\epsilon)^{\Gamma-1}\epsilon)^{|F|} \quad (5.21)$$

$$= \mathcal{W}(\mathcal{F}_i; 2(1+\epsilon)^{\Gamma-1}\epsilon). \quad (5.22)$$

Set $\eta = \max_i \mathcal{W}(\mathcal{F}_i; 2(1+\epsilon)^{\Gamma-1}\epsilon)$. By the triangular inequality of diamond norm, we see that $\|\mathcal{C}_{i,+}\|_{\diamond} \leq 1 + \eta$ for all i . Using these upper bounds, we can bound the total contribution of the bad fault paths on the whole circuit.

$$\text{map}[C_{\text{FT}}[\mathbf{f}]] = \text{map}[C_m[\mathbf{f}_m]] \circ \dots \circ \text{map}[C_1[\mathbf{f}_1]] \quad (5.23)$$

$$= (\mathcal{C}_{m,+} + \mathcal{C}_{m,-}) \circ \dots \circ (\mathcal{C}_{1,+} + \mathcal{C}_{1,-}). \quad (5.24)$$

Denote

$$\mathcal{C}_+ = \mathcal{C}_{m,+} \circ \mathcal{C}_{m-1,+} \circ \dots \circ \mathcal{C}_{1,+}, \quad \mathcal{C}_- = \text{map}[C_{\text{FT}}[\mathbf{f}]] - \mathcal{C}_+. \quad (5.25)$$

\mathcal{C}_- has a decomposition based on the last gadget that failed, as follows.

$$\mathcal{C}_- = \sum_{i=1}^m \mathcal{C}_{m,+} \circ \mathcal{C}_{m-1,+} \circ \dots \circ \mathcal{C}_{i,-} \circ (\mathcal{C}_{i-1,+} + \mathcal{C}_{i-1,-}) \circ \dots \circ (\mathcal{C}_{1,+} + \mathcal{C}_{1,-}). \quad (5.26)$$

This decomposition enables us to bound the diamond norm of \mathcal{C}_- . By the sub-multiplicativity of the diamond norm, and the fact that $\mathcal{C}_{i,+} + \mathcal{C}_{i,-}$ has diamond norm 1 for all i ,

$$\|\mathcal{C}_-\|_{\diamond} \leq \sum_{i=1}^m \|\mathcal{C}_{m,+} \circ \mathcal{C}_{m-1,+} \circ \dots \circ \mathcal{C}_{i,-} \circ (\mathcal{C}_{i-1,+} + \mathcal{C}_{i-1,-}) \circ \dots \circ (\mathcal{C}_{1,+} + \mathcal{C}_{1,-})\|_{\diamond} \quad (5.27)$$

$$\leq \sum_{i=1}^m (1+\eta)^{m-i} \eta \quad (5.28)$$

$$\leq m(1+\eta)^m \eta. \quad (5.29)$$

On the other hand, since \mathcal{C}_+ is the sum over noisy executions of C_{FT} with \mathcal{F} -avoiding faults, it holds by the definition of gadget that

$$\mathcal{C}_+ = \alpha \cdot \text{map}[C]. \quad (5.30)$$

for some $\alpha \in \mathbb{C}$. Now fix an input state $|x\rangle\langle x|$, we write

$$\text{tr} |\text{map}[C] (|x\rangle\langle x|) - \text{map}[C_{\text{FT}}[\mathbf{f}]](|x\rangle\langle x|)| \quad (5.31)$$

$$= \text{tr} |\text{map}[C] (|x\rangle\langle x|) - (\mathcal{C}_+ + \mathcal{C}_-)(|x\rangle\langle x|)| \quad (5.32)$$

$$\leq \text{tr} |\text{map}[C] (|x\rangle\langle x|) - \alpha \cdot \text{map}[C] (|x\rangle\langle x|)| + \text{tr} |\mathcal{C}_- |x\rangle\langle x|| \quad (5.33)$$

$$\leq |1 - \alpha| + \|\mathcal{C}_-\|_\diamond \quad (5.34)$$

$$\leq |1 - \alpha| + m(1 + \eta)^m \eta, \quad (5.35)$$

where we used here that the trace norm is bounded by the diamond norm. To bound $|1 - \alpha|$, note that

$$1 = \text{tr}(\text{map}[C_{\text{FT}}[\mathbf{f}]](|x\rangle\langle x|)) = \text{tr}(C_+ |x\rangle\langle x|) + \text{tr}(C_- |x\rangle\langle x|) = \alpha + \text{tr}(C_- |x\rangle\langle x|). \quad (5.36)$$

Therefore we can bound

$$|1 - \alpha| = |\text{tr}(C_- |x\rangle\langle x|)| \leq \text{tr} \|C_- |x\rangle\langle x|\| \leq \|C_- |x\rangle\langle x|\|_\diamond = m(1 + \eta)^m \eta. \quad (5.37)$$

We conclude that

$$\frac{1}{2} \text{tr} |\text{map}[C] (|x\rangle\langle x|) - \text{map}[C_{\text{FT}}[\mathbf{f}]](|x\rangle\langle x|)| = m(1 + \eta)^m \eta. \quad \square$$

6 Application: qLDPC codes and transversal gates

We show that a standard construction of error correction gadgets (originating from [Den+02; Got14]) for quantum-low density parity check (qLDPC) codes satisfies our gadget definition.

6.1 Correctable sets

A CSS code defined by the check matrices (H_X, H_Z) is said to be Δ -qLDPC if every row and column of H_X and H_Z has weight at most Δ .

For an error rate p (IID) on n qubits, on average, errors have weight around np . Thus, naively one might conclude that linear distance scaling is a necessary condition for a code family to achieve a threshold. This turns out to be too strong: It suffices to be able to correct “most” errors of weight np . It turns out that the qLDPC property imposes strong constraints¹⁵ on the low weight logical operators of the code, so that it is rare for a random error to have large overlap with a logical operator. [KP13] showed that all qLDPC code families with polynomial distance scaling possess a threshold – below a critical error rate, recovery from depolarizing noise fails with exponentially small (in the code distance) probability when decoding with a minimum weight decoder.

We begin by defining a minimum-weight decoder for a classical linear code as well as a minimum-weight decoder in the X/Z basis which we refer to as the CSS minimum-weight decoder.

¹⁵More specifically, the qLDPC property induces a sort of “geometry” on the space of qubits, and logical operators must be the union of large connected components.

Definition 6.1 (Minimum-weight decoder). For a classical code defined by a $m \times n$ parity check matrix H , given a syndrome $s \in \mathbb{F}_2^m$ in the column space of H , a minimum weight decoder outputs a minimum Hamming weight vector in the set of valid corrections, $\{x \in \mathbb{F}_2^n : Hx = s\}$. For a CSS code defined by (H_X, H_Z) , a CSS minimum weight decoder $D: \mathbb{F}_2^{n-k} \mapsto \{I, X, Y, Z\}^n$ consists of two minimum weight decoders, D_X, D_Z , for matrices H_X, H_Z . For a syndrome $s = (s_X, s_Z)$, where s_X, s_Z are the X and Z syndromes respectively,

$$D(s) = X^{D_Z(s_Z)} \cdot Z^{D_X(s_X)}. \quad (6.1)$$

Here for a vector $v \in \mathbb{F}_2^n$, we use X^v to denote the Pauli operator $X^{v_1} \otimes \dots \otimes X^{v_n}$, where $X^1 = X$ and $X^0 = I$. Z^v is defined similarly.

We now need a parameterized notion of bad sets (from [Got14]) of the code. These are the sets that will trick our decoder into applying the wrong correction.

Definition 6.2 (Adjacency graph). For a check matrix $H \in \mathbb{F}_2^{r \times n}$, the adjacency graph of H , denoted $G_{\text{adj}}[H]$ is a graph on the vertices $[n]$, where two vertices $i \neq j \in [n]$ are adjacent if there exists a row k which is non-zero on columns i and j . I.e. if bit vertices i and j are connected by a constraint vertex k in the Tanner graph associated with H .

We will now associate length- n bitstrings with subsets of $[n]$.

Proposition 6.3 ([KP13]). For a pair of check matrices (H_X, H_Z) defining a CSS code of distance d , any non-trivial X logical operator with support x must induce a subgraph of $G_{\text{adj}}[H_Z]$ with at least one connected component of size at least d .

Definition 6.4 (Clustering sets). For a graph $G = (V, E)$ the (d, t) -clustering sets $\mathcal{CG}_{(d,t)} \subseteq P(V)$ of G are the subsets $W \subseteq U \subseteq V$ of V of size $|W| = t$ contained in a subset U of size $|U| = d$ that induces a connected subgraph of G .

$$\mathcal{CG}_{(d,d)} = \{U \subseteq V \mid |U| = d \text{ and } U \text{ induces a connected subgraph of } G\} \quad (6.2)$$

$$t < d \quad \mathcal{CG}_{(d,t)} = \bigcup_{U \in \mathcal{CG}_{(d,d)}} \{W \subseteq U \mid |W| = t\} \quad (6.3)$$

The t parameter of these clusters is the appropriate analog of the weight of an error for qLDPC code families with sublinear distance: The error cannot be too high weight in any one region. As we will see, this parameter will behave much like Hamming weight in that we can recovery a sort of sub-additivity, see Proposition 6.11.

We give upper bounds on the weight enumerator of the clustering sets by bounding the number of bad sets analogously to [Got14]. We first need the following lemma from [AGP07].¹⁶

¹⁶We actually require the stronger variant proven (but not stated) in [AGP07]. See Lemma 2 of [Got14].

Lemma 6.5 (Cluster counting [AGP07] Lemma 5). Let $G = (V, E)$ be a graph with maximum degree Δ . Fix a subset $S \subseteq V$ of vertices. Then, consider subsets $C \subseteq V$ that is a union of connected sets $C = \cup_i C_i$ in G where each set C_i contains at least one element of S , $C_i \cap S \neq \emptyset$. We say that C is a **cluster cover** of S , and denote the collection of cluster covers of size m to be $\mathcal{CO}(m, S)$. The number of such subsets of size m , $M(m, S)$ satisfies the upper bound

$$M(m, S) \leq e^{|S|-1} (\Delta e)^{m-|S|} \quad (6.4)$$

This allows us to upper bound the weight enumerator of this family.

Lemma 6.6. For a maximum degree- Δ graph $G = (V, E)$ with the (d, t) -clustering sets $\mathcal{CG}_{(d,t)} \subseteq P(V)$, we have that ($x \in [0, 1]$)

$$\mathcal{W}(\mathcal{CG}_{(d,t)}; x) \leq \left(\binom{d}{t-1} (\Delta e)^{d-1} |V| \right) x^t \quad (6.5)$$

Proof. We compute the number of elements of $\mathcal{CG}_{(d,t)}$ by taking a union bound over the following decomposition:

$$\mathcal{CG}_{(d,t)} \subseteq \bigcup_{v \in V} \bigcup_{\substack{U \subseteq V \\ |U|=d \\ U \text{ is connected in } G}} \{W \subseteq U \mid |W| = t \text{ and } v \subseteq W\} \quad (6.6)$$

Lemma 6.5 upper bounds the number of connected (in G) sets $U \subseteq V$ of size $|U| = d$ containing a particular vertex $v \in V$ as $(\Delta e)^{d-1}$. There are $\binom{d}{t-1}$ subsets of U that contain v . By construction, each element of $\mathcal{CG}_{(d,t)}$ has weight t . The result follows. \square

We are now ready to define our code types. From now on, fix a Δ -qLDPC code \mathcal{Q} with parameters $\llbracket n, k, d \rrbracket$ defined by check matrices (H_X, H_Z) and an arbitrary choice of irreversible encoding map implementable by a Clifford circuit M acting on a logical state $D(\mathcal{H}^k)$ and an $n - k$ ancilla register initialized to $|0\rangle^{\otimes(n-k)}$. Call this the **computational code**. M satisfies the property that for some choice of $n - k$ generators of the stabilizer $\{s_i\}_{i=1}^{n-k} \subseteq \{I, X, Y, Z\}^{n-k}$, any eigenstate in eigenspace $s \in \mathbb{F}_2^{n-k}$ of the stabilizer generators (the **syndrome space** s) can be written as $M(|\psi\rangle \otimes |s\rangle)$ for some state $|\psi\rangle \in \mathcal{H}^k$.

Definition 6.7 (Purified decoder). Let $\sigma: \{I, X, Y, Z\}^n \mapsto \mathbb{F}_2^{n-k}$ be the syndrome map with respect to the generating set of the stabilizer $\{s_i\}_{i=1}^{n-k} \subseteq \{I, X, Y, Z\}^n$ of \mathcal{Q} and $D: \mathbb{F}_2^{n-k} \mapsto \{I, X, Y, Z\}^n$ be a decoding algorithm outputting a correction such that $\sigma \circ D(s) = s$. We can define the purification U_D of D as the unitary U that measures the syndrome s , applies the corresponding correction $D(s)$, and places the value of the syndrome in the ancilla register after applying the unencoding circuit M^\dagger . Define the syndrome space projector $\Pi_s = M(I \otimes |s\rangle\langle s|) M^\dagger$. Then U_D can be written in terms of a sum over syndrome spaces and the corresponding correc-

tions.

$$U_D = \sum_{s \in \mathbb{F}_2^{n-k}} (I \otimes |s\rangle\langle 0|) M^\dagger D(s) \Pi_s \quad (6.7)$$

Definition 6.8 (Code types for computational code). Let $\mathcal{CG}_{(m,t)}^{(X)}$ and $\mathcal{CG}_{(m,t)}^{(Z)}$ be the (m, t) -clustering sets in $G_{\text{adj}}[H_X]$ and $G_{\text{adj}}[H_Z]$, respectively. Let $c \in [0, 1]$. All the bad error supports will be stated in terms of c . We define the bad error supports to be all sufficiently large subsets of connected subsets of size at least d .

$$\mathcal{B}_c^{(X)} = \boxplus_{m=d}^n \mathcal{CG}_{(m, \lceil c \cdot m/2 \rceil)}^{(X)} \quad (6.8)$$

$$\mathcal{B}_c^{(Z)} = \boxplus_{m=d}^n \mathcal{CG}_{(m, \lceil c \cdot m/2 \rceil)}^{(Z)} \quad (6.9)$$

We define $\mathcal{B}_c \equiv \mathcal{B}_c^{(X)} \boxplus \mathcal{B}_c^{(Z)}$.

The reversible decoding unitary $U_{\mathcal{Q}}$ is the purification of a CSS minimum-weight decoder (see Definition 6.1) for \mathcal{Q} , as defined in Eq. (6.7). The code type CompCode_c is defined to be the pair of bad sets and the purified minimum weight decoder $(U_{\mathcal{Q}}, \mathcal{B}_c \equiv \mathcal{B}_c^{(X)} \boxplus \mathcal{B}_c^{(Z)})$.

Note that $U_{\mathcal{Q}}$ inverts M on the trivial syndrome space and can be easily confirmed to be unitary.

$$\forall |\psi\rangle \in \mathcal{H}^k \quad |\psi\rangle = (U_{\mathcal{Q}} M) |\psi\rangle |0\rangle^{\otimes n-k} \quad (6.10)$$

We now show that CompCode_c is a valid code type Definition 4.2.

Proposition 6.9 (Correctable errors [KP13; Got14]). For $c \in [0, 1]$, let $\text{CompCode}_c = (U_{\mathcal{Q}}, \mathcal{B}_c)$. Fix a logical state $\rho \in \mathcal{D}(\mathcal{H}^k)$ and a state $\sigma \in \mathcal{L}(\mathcal{H}^n)$ such that σ is \mathcal{B}_c -deviated from the encoding of ρ : $\text{enc}(\rho) \lesssim_{\mathcal{B}_c} \sigma$.

Then, the decoding of σ is proportional to ρ .

$$\text{dec}(\sigma) \propto \rho \quad (6.11)$$

Proof. Let \mathcal{E} be a superoperator whose support $S \subseteq [n]$ is \mathcal{B}_c -avoiding. First, we write the Kraus decomposition of the superoperator in terms of Pauli operators $\{K_\mu\}_\mu, \{K'_\nu\}_\nu$ supported on $S \subseteq [n]$ and complex coefficients $\{\alpha_{\mu,\nu}\}_{\mu,\nu}$

$$\mathcal{E} \circ \text{enc}(\rho) = \sum_{\mu,\nu} \alpha_{\mu,\nu} K_\mu M(\rho \otimes |0\rangle\langle 0|) M^\dagger K'_\nu \quad (6.12)$$

Using this decomposition and the definition of U_Q (Eq. (6.7)), we consider the action of dec on σ .

$$\text{dec}(\sigma) = \text{tr}_{\mathcal{H}^r} \left(U_Q(\mathcal{E} \circ \text{enc}(\rho)) U_Q^\dagger \right) \quad (6.13)$$

$$= \text{tr}_{\mathcal{H}^r} \left(\sum_{s,t \in \mathbb{F}_2^{n-k}} (I \otimes |s\rangle\langle 0|) M^\dagger D(s) \Pi_s \left(\sum_{\mu,\nu} \alpha_{\mu,\nu} K_\mu M(\rho \otimes |0\rangle\langle 0|) M^\dagger K'_\nu \right) \Pi_t D(t) M(I \otimes |0\rangle\langle t|) \right) \quad (6.14)$$

$$= \text{tr}_{\mathcal{H}^r} \left(\sum_{s,t \in \mathbb{F}_2^{n-k}} \left(\sum_{\mu,\nu} \alpha_{\mu,\nu} (I \otimes |s\rangle\langle 0|) M^\dagger D(s) \Pi_s K_\mu M(\rho \otimes |0\rangle\langle 0|) M^\dagger K'_\nu \Pi_t D(t) M(I \otimes |0\rangle\langle t|) \right) \right) \quad (6.15)$$

Let s_μ, s_ν denote the syndrome of K_μ, K'_ν . The projectors Π_s and Π_t annihilate all terms except for those where $s_\mu = s$ and $s_\nu = t$. Suppose $D(s)K_\mu$ and $D(t)K'_\nu$ are both stabilizers, we can write

$$D(s) \Pi_s K_\mu M(\rho \otimes |0\rangle\langle 0|) M^\dagger K'_\nu \Pi_t D(t) = M(\rho \otimes |0\rangle\langle 0|) M^\dagger. \quad (6.16)$$

We can then simplify the equation for $\text{dec}(\sigma)$.

$$\text{dec}(\sigma) = \text{tr}_{\mathcal{H}^r} \left(\sum_{s,t \in \mathbb{F}_2^{n-k}} \left(\sum_{\mu,\nu: s_\mu=s, s_\nu=t} \alpha_{\mu,\nu} (I \otimes |s\rangle\langle 0|) M^\dagger M(\rho \otimes |0\rangle\langle 0|) M^\dagger M(I \otimes |0\rangle\langle t|) \right) \right) \quad (6.17)$$

$$= \text{tr}_{\mathcal{H}^r} \left(\sum_{s,t \in \mathbb{F}_2^{n-k}} \left(\sum_{\mu,\nu: s_\mu=s, s_\nu=t} \alpha_{\mu,\nu} \rho \otimes |s\rangle\langle t| \right) \right). \quad (6.18)$$

The partial trace annihilates all terms where $s \neq t$. Let α_s be the sum of all $\alpha_{\mu,\nu}$ where $s_\mu = s_\nu = s$. Then the above simplifies to

$$\text{dec}(\sigma) = \sum_{s \in \mathbb{F}_2^{n-k}} \alpha_s \rho. \quad (6.19)$$

It suffices for us to show that for an arbitrary Pauli error E with $\mathcal{B}_c^{(X)} \boxplus \mathcal{B}_c^{(Z)}$ -avoiding support, the minimum weight decoder will correct E up to a stabilizer. Decompose the Pauli error $E = pE_X E_Z$ into X and Z parts $E_X \in \{X, I\}^n$ and $E_Z \in \{Z, I\}^n$ up to a scalar $p \in \{\pm 1, \pm i\}$. Consider the correction of E_X , the case of E_Z is identical with the roles of Z and X interchanged.

Since E has \mathcal{B}_c -avoiding support, E_X must have $\mathcal{B}_c^{(Z)} = \boxplus_{m=d}^n \mathcal{CG}_{(m, \lceil c \cdot m/2 \rceil)}^{(Z)}$ -avoiding support. The minimum weight decoder applies a minimum weight correction $C_X \in \{I, X\}^n$ such that $C_X E_X$ has trivial syndrome. We will show that $C_X E_X$ is in the stabilizer of the code, following [KP13; Got14].

We first establish some properties of the corrected error $C_X E_X$. $\text{supp}(C_X E_X)$ induces a subgraph of the adjacency graph $G_{\text{adj}}[H_Z]$ that has connected cluster L_1, \dots, L_ℓ . From the definition of the adjacency graph, each connected cluster must have trivial syndrome (commute with the stabilizer of the code) if $C_X E_X$ does. Thus, the restriction of $C_X E_X$ to any connected cluster L' of size

strictly less than the code distance d must be in the stabilizer of the code. It also must be the case that on L' , the error is supported on at least half of the set, $|L' \cap \text{supp } E_X| \geq |L'|/2$. Otherwise, $E_X|_{L'}$ would be a lower weight correction than $C_X|_{L'}$ for $E_X|_{L'}$.

Now suppose, for a contradiction, that $C_X E_X$ is not a stabilizer of the code. Then, at least one connected cluster L of $\text{supp}(C_X E_X)$ in $G_{\text{adj}}[H_Z]$ must have size $m \geq d$. Since L satisfies $|L \cap \text{supp } E_X| \geq |L|/2$, $\text{supp } E_X$ contains a subset of L of size at least $m/2$. So this subset is a superset of an element of $\mathcal{CG}_{(m, \lceil c \cdot m/2 \rceil)}^{(Z)}$, contradicting the assumption that E was \mathcal{B}_c -avoiding.

An identical argument holds for $C_Z E_Z$, so $C_Z E_Z C_X E_X$ is in the stabilizer of the code and E is corrected by the CSS minimum-weight decoder. This completes our proof. \square

We will also need to be able to analyze the structure of unions of sets that avoid the clustering sets.

Definition 6.10 (Separable Bad Sets). For $\mathcal{B}, \mathcal{B}_1, \mathcal{B}_2 \subseteq P(\Omega)$, we say that \mathcal{B} is **separable** into $\mathcal{B}_1, \mathcal{B}_2$ if for all \mathcal{B}_1 -avoiding set X and \mathcal{B}_2 -avoiding set Y , $X \cup Y$ is \mathcal{B} -avoiding.

Proposition 6.11 (Union of avoiding sets). Let $G = (V, E)$ be a graph, and $\mathcal{CG}_{(d,t)}$ be the (d, t) -clustering sets of G . For $s, t \in \mathbb{N}$ such that $s + t \leq d$, $\mathcal{CG}_{(d,s+t)}$ is separable into $\mathcal{CG}_{(d,s)}$ and $\mathcal{CG}_{(d,t)}$.

Proof. Consider $X, Y \subseteq [V]$ such that X is $\mathcal{CG}_{(d,s)}$ -avoiding and Y is $\mathcal{CG}_{(d,t)}$ -avoiding, let $Z = X \cup Y$. Suppose, for a contradiction, that Z is not $\mathcal{CG}_{(d,s+t)}$ -avoiding. Then, there exists a subset of vertices $U \subseteq V$ of size $|U| = d$ such that U induces a connected subgraph of G and $|U \cap Z| \geq s + t$.

However, since X and Y are $\mathcal{CG}_{(d,s)}$ -avoiding and $\mathcal{CG}_{(d,t)}$ -avoiding, respectively, we have that their intersections with U must be small: $|U \cap X| < s$ and $|U \cap Y| < t$. The contradiction follows.

$$s + t \leq |U \cap Z| \leq |U \cap X| + |U \cap Y| < s + t. \quad \square$$

It is a straightforward corollary that if we combine errors (e.g. by applying a two-qubit gate transversally) then the resulting error is well controlled.

Corollary 6.12. For $c + c' \leq 1$, $\mathcal{B}_{c+c'}$ is separable into \mathcal{B}_c and $\mathcal{B}_{c'}$.

Proof. Apply Proposition 6.11 to the clustering sets in the definition of \mathcal{B}_c . \square

Concluding our definition of the computational code type, we bound the weight enumerator of \mathcal{B}_c .

Lemma 6.13.

$$\mathcal{W}(\mathcal{B}_c; x) \leq n \cdot [O_{\Delta, c}(x)]^{\frac{cd}{2}}. \quad (6.20)$$

Proof. Since H_X, H_Z has row and column weights bounded by Δ , the graphs $G_{\text{adj}}[H_X], G_{\text{adj}}[H_Z]$ has degree bounded by Δ^2 . We can bound the weight enumerator of \mathcal{B}_c with Lemma 6.6.

$$\mathcal{W}(\mathcal{B}_c^{(X)}; x) \leq \sum_{m \geq d} \mathcal{W}(\mathcal{CG}_{(m, \lceil c \cdot m/2 \rceil)}^{(X)}; x) \quad (6.21)$$

$$\leq \sum_{m \geq d} n \cdot (2e\Delta^2)^m x^{\frac{cm}{2}} \quad (6.22)$$

$$\leq n \cdot [O_{\Delta, c}(x)]^{\frac{cd}{2}} \quad (6.23)$$

Since $\mathcal{B}_c = \mathcal{B}_c^{(X)} \boxplus \mathcal{B}_c^{(Z)}$, we multiply the above upper bound by a factor of 2, which is absorbed into the constants in $O_{\Delta, c}(x)$. \square

6.2 Spacetime code

Before we are ready to construct and prove properties of the error correction gadget, we first need to construct the following classical code commonly referred to as a “spacetime code.” In what follows, we will use the CSS property to focus on correction of X errors. The case of Z errors will follow with X and Z interchanged.

Definition 6.14 (Spacetime code). Let (H_X, H_Z) be a CSS code. The spacetime code of $H_Z \in \mathbb{F}_2^{r_Z \times n}$ is a modified check matrix that corresponds to the spacetime history of repeated syndrome extraction.^a

The spacetime code check matrix $H \in \mathbb{F}_2^{(Tr_Z) \times T(n+r_Z)}$ will have T copies of H_Z along the diagonal and T blocks of r_Z extra columns corresponding to syndrome measurement errors. Each measurement error will be able to flip the syndrome of a check in two consecutive rounds.

We describe this more explicitly. In what follows, for two intervals I and J , we will use $H[I, J]$ to denote the sub-matrix with rows in I and columns in J . For $t \in [T]$, we define the data error intervals and syndrome error intervals (see Fig. 6).

$$I_t = [(t-1) \cdot r_Z + 1, t \cdot r_Z - 1] \quad (6.24)$$

$$J_t^d = [(t-1) \cdot n + 1, t \cdot n - 1] \quad (6.25)$$

$$J_t^s = [(t-1) \cdot r_Z + Tn, t \cdot r_Z + Tn - 1] \quad (6.26)$$

We can now specify the entries of H (all unspecified entries are 0).

$$t \in [T] \quad H[I_t, J_t^d] = H_Z \quad (6.27)$$

$$t \in [T] \quad H[I_t, J_t^s] = \text{Id}_{r_Z \times r_Z} \quad (6.28)$$

$$t \in [T-1] \quad H[I_t, J_{t+1}^s] = \text{Id}_{r_Z \times r_Z} \quad (6.29)$$

We will also require an “interpretation” of the results of decoding the spacetime code. Define the **flattening map** $\phi: \mathbb{F}_2^{Tn+Tr_Z} \rightarrow \mathbb{F}_2^n$ given by dropping the syndrome error blocks and summing

	J_1^d	\dots	J_T^d	J_1^s	\dots	J_T^s
I_1	H_Z			Id		
\vdots		H_Z		Id	Id	
I_T			H_Z		Id	Id

Figure 6: Spacetime code check matrix

together the data error blocks $x \mapsto \sum_{t \in [T]} x[J_t^d]$.

^aSyndrome bits of this code are sometimes called “detectors.”

The rows I_t correspond to the change in syndrome measurements in the t -th round. The columns J_t^s correspond to syndrome measurement errors affecting the syndrome at rounds $t - 1$ and t while the columns J_t^d correspond to new data errors accumulated in round t .

For our error correction gadget, we will consider the **input error**, which corresponds to bits of J_1^d , separately from the remaining error. For this purpose, we define the **bulk** of the spacetime code with bits $J_{\text{bulk}} := [T(n + r_Z)] \setminus J_1^d$ and checks $I_{\text{bulk}} = I_2 \cup \dots \cup I_T$ namely, all data and syndrome bits except for the input error bits and all rows except for the first. We denote this **bulk check matrix** H_{bulk} . Similarly, we define the **bulk flattening map** $\phi_{\text{bulk}}: \mathbb{F}_2^{(T-1)n + Tr_Z} \rightarrow \mathbb{F}_2^n$ as $\phi_{\text{bulk}}(x) = \sum_{t=2}^T x[J_t^d]$. We call the last r_Z bits J_T^s of a bitstring $\mathbb{F}_2^{Tn + Tr_Z}$ the **final round syndrome error** bits, and the first r_Z bits J_1^s the **initial round syndrome error**.

We will now prove some properties about the spacetime code: Let d be the distance of the original CSS code.

Lemma 6.15 (Final round syndrome). For a bitstring $x \in \mathbb{F}_2^{Tn + Tr_Z}$ with trivial spacetime syndrome $Hx = 0$, the syndrome of the flattened bitstring $\phi(x)$ is equal to the final round syndrome error.

$$H_Z \phi(x) = x[J_T^s] \quad (6.30)$$

Proof. x has zero syndrome with respect to the spacetime code $Hx = 0$ which implies (by definition of H) the following system of equations.

$$t \in [T - 1] \quad H_Z x[J_{t+1}^d] + x[J_t^s] + x[J_{t+1}^s] = 0 \quad (6.31)$$

$$H_Z x[J_1^d] + x[J_1^s] = 0 \quad (6.32)$$

Summing over this linear system yields the result.

$$\sum_{t \in [T]} H_Z x[J_t^d] + x[J_T^s] = H_Z \phi(x) + x[J_T^s] = 0. \quad \square$$

Lemma 6.16. Similarly, for a bitstring $x_{\text{bulk}} \in \mathbb{F}_2^{(T-1)n+Tr_Z} \cong \mathbb{F}_2^{J_{\text{bulk}}}$ with no bulk spacetime syndrome $H_{\text{bulk}}x = 0$, its flattened syndrome is the difference between the initial and final round syndrome errors.

$$H_Z \phi_{\text{bulk}}(x_{\text{bulk}}) = x_{\text{bulk}}[J_T^s] + x_{\text{bulk}}[J_1^s] \quad (6.33)$$

Proof. As in the proof of Lemma 6.15, the zero syndrome condition implies the following equation from which the result follows.

$$H_Z \phi_{\text{bulk}}(x_{\text{bulk}}) + x_{\text{bulk}}[J_T^s] + x_{\text{bulk}}[J_1^s] = 0 \quad \square$$

Lemma 6.17. For a bitstring $x \in \mathbb{F}_2^{Tn+Tr_Z}$ with trivial spacetime syndrome $Hx = 0$ and trivial final round syndrome error $x[J_T^s] = 0$, if x is the union of connected components in the adjacency graph of the spacetime code $G_{\text{adj}}[H]$ of size strictly less than the distance $< d$, then x flattens to a trivial logical operator of the CSS code, $\phi(x) \in \text{rowsp}(H_X)$.

Proof. Consider a single connected component x' of x in $G_{\text{adj}}[H]$. Since no other row of H with support on x' has support on $x - x'$, it must be that $Hx' = 0$ which implies $H_Z \phi(x') = 0$ by Lemma 6.15 and the assumption on the final round syndrome bits $x[J_T^s] = 0$. Using the assumption on the size of each connected component, the Hamming weight of the flattened bitstring is less than the distance $|\phi(x')| \leq |x'| < d$.

$\phi(x')$ is in the kernel of H_Z and has Hamming weight less than d , so by the distance of the CSS code (H_X, H_Z) , it must be in the row space of H_X , $\phi(x') \in \text{rowsp}(H_X)$. It then follows that every connected component of x flattens to an element of the row space of H_X . $\phi(x)$ is the sum over the flattening of every connected component so it is also in the row space of H_X . \square

Definition 6.18 (Extended spacetime decoding graph). To bound the residual error on the code after correction, we introduce a slight modification to $G_{\text{adj}}[H]$. We expand H by n columns and r_Z rows, which we denote J_{T+1}^d and I_{T+1} . We denote this new matrix H_{ext} with the additional entries

$$H_{\text{ext}}[I_{T+1}, J_{T+1}^d] = H_Z, \quad H_{\text{ext}}[I_{T+1}, J_T^s] = \text{Id}_{r_Z \times r_Z} . \quad (6.34)$$

The graph $G_{\text{adj}}[H_{\text{ext}}]$, compared to $G_{\text{adj}}[H]$, has one more layer of n vertices labeled by J_{T+1}^d which are connected to vertices labeled by J_T^s . We say that $G_{\text{adj}}[H_{\text{ext}}]$ is the **extended spacetime decoding graph**.

Lemma 6.19 (Residual Error [Got14]). For a bitstring $x \in \mathbb{F}_2^{Tn+Tr_Z}$ with trivial spacetime syndrome $Hx = 0$, let $y \in \mathbb{F}_2^n$ be a minimum weight vector (supported on J_{T+1}^d) such that $H_Z \phi(x) = H_Z y$. Let $S \subseteq J_{T+1}^d$ be a set. If y contains S , then there must exist a cluster cover R of S (see Lemma 6.5), $R \subseteq x \sqcup y$, such that $|R| \geq 2|S|$ and for every cluster C of R , we have

$$|x \cap C| \geq |C|/2.$$

Proof. Consider the set of vertices $x \sqcup y$. Let $x' \sqcup y'$ denote the restriction of $x \sqcup y$ to maximal connected clusters in $G_{\text{adj}}[H_{\text{ext}}]$ containing vertices in S , with $x' \subseteq x, y' \subseteq y$. Note that $S \subseteq y' \subseteq y$ and $R = x' \sqcup y'$ is a cluster cover of S in $G_{\text{adj}}[H_{\text{ext}}]$. By Lemma 6.15, for every cluster C in R , we have $H_Z(y'|_C) = H_Z\phi(x'|_C)$. Moreover, since y is the minimum weight vector with $H_Z y = H_Z\phi(x)$, we have that $(y'|_C)$ is the minimum weight vector with $H_Z(y'|_C) = H_Z\phi(x'|_C)$. Therefore

$$|x \cap C| = |x'|_C| \geq |\phi(x'|_C)| \geq |y'|_C| = |C \cap S|, \quad (6.35)$$

which implies that $2|x \cap C| \geq |x'|_C| + |y'|_C| = |C|$. Our claim follows. \square

Lemma 6.20. For a bitstring $x \in \mathbb{F}_2^{Tn+Trz}$ with trivial spacetime syndrome $Hx = 0$, let $y \in \mathbb{F}_2^n$ be a minimum weight vector (supported on J_{T+1}^d) such that $H_Z\phi(x) = H_Z y$. If $\phi(x) - y \notin \text{rowsp}(H_X)$, then there must exist a connected cluster of vertices S of size at least d in $G_{\text{adj}}[H_{\text{ext}}]$ such that $|x \cap S| \geq |S|/2$.

Proof. Since $\phi(x) - y \notin \text{rowsp}(H_X)$, it must be the case that $\phi(x) - y$ contains a connected cluster of size at least d in $G_{\text{adj}}[H_Z]$, the adjacency matrix of the base code H_Z . This implies that $x \sqcup y$ contains a connected cluster S of size at least d in $G_{\text{adj}}[H_{\text{ext}}]$. Restricting to this cluster S , we see that $H_Z y|_S = H_Z\phi(x|_S)$, and $y|_S$ is the minimum weight vector satisfying this equality. Therefore we have

$$|x \cap S| = |x|_S| \geq |\phi(x|_S)| \geq |y|_S|. \quad (6.36)$$

This implies that $|x \cap S| \geq |S|/2$, as desired. \square

We are now ready to prove when a correction procedure for the spacetime code succeeds.

Definition 6.21 (Decoding). Consider the extended spacetime decoding graph $G_{\text{adj}}[H_{\text{ext}}]$. Let $e_{\text{in}} \in \mathbb{F}_2^{|J_1^d|}$ be an input error, and $e \in \mathbb{F}_2^{|J_{\text{bulk}}|}$ be an error on the bulk. We denote $\tilde{e} = e_{\text{in}} \sqcup e$. Let $c \in \mathbb{F}_2^{|J_{\text{bulk}}|}$ be the minimum weight vector such that $H(e) = H(c)$, and let $c_{\text{in}} \in \mathbb{F}_2^n \cong \mathbb{F}_2^{|J_1^d|}$ be the minimum weight vector such that

$$H_Z(c_{\text{in}}) = H_Z(e_{\text{in}}) + (e + c)|_{J_1^s}. \quad (6.37)$$

Denote $\tilde{c} = c_{\text{in}} \sqcup c$, then \tilde{c} is the output correction of this minimum-weight decoding procedure, with $H(\tilde{e} + \tilde{c}) = 0$. Let $y \in \mathbb{F}_2^{|J_{T+1}^d|}$ be the minimum weight vector that satisfies

$$H_Z\phi(\tilde{e} + \tilde{c}) = H_Z y, \quad (6.38)$$

we say that y is the **residue error** on the code block after decoding.

Lemma 6.22 (Successful Decoding). Recall the code type and bad error supports defined in Definition 6.8. Following the definitions and notations in Definition 6.21, let $\mathcal{B}^{(R)} \subseteq P(J_{T+1}^d)$ be a collection of bad error supports for the residue error. There exists a family of bad fault paths $\mathcal{F}_{\text{spacetime}} \subseteq P(J_{\text{bulk}})$, which depends on $\mathcal{B}^{(R)}$, such that the following conditions hold.

- **Bounded residue error.** If e is $\mathcal{F}_{\text{spacetime}}$ -avoiding, then y is $\mathcal{B}^{(R)}$ -avoiding.
- **No logical error on good input.** If additionally e_{in} is $\mathcal{B}_{1/2}^{(Z)}$ -avoiding, then $\phi(\tilde{e} + \tilde{c}) + y \in \text{rowsp}(H_X)$.

Additionally, $\mathcal{F}_{\text{spacetime}}$ has weight enumerator

$$\mathcal{W}(\mathcal{F}_{\text{spacetime}}; x) \leq |J| \cdot [O_{\Delta}(x)]^{d/40} + \mathcal{W}(\mathcal{B}^{(R)}; O_{\Delta}(x^{1/2})) . \quad (6.39)$$

Proof. We construct $\mathcal{F}_{\text{spacetime}}$ out of three collection of bad fault paths. Let $G_{\text{adj}}[H_{\text{bulk}}]$ denote the induced subgraph of $G_{\text{adj}}[H_{\text{ext}}]$ with vertices J_{bulk} . The first collection, \mathcal{F}_1 , are error supports that have significant overlap with any connected cluster of size at least d in $G_{\text{adj}}[H_{\text{bulk}}]$. Recall that $\mathcal{CG}_{(m,t)}^{(H_{\text{bulk}})}$ are the (m, t) clustering sets in $G_{\text{adj}}[H_{\text{bulk}}]$. Define

$$\mathcal{F}_1 = \boxplus_{m \geq d} \mathcal{CG}_{(m, \lceil m/10 \rceil)}^{(H_{\text{bulk}})} . \quad (6.40)$$

Now suppose e is a \mathcal{F}_1 -avoiding fault, and let us consider the connected components of $\tilde{e} + \tilde{c}$. There are three types of components:

- Type 1. Input components** that contain vertices in J_1^d and do not contain vertices in J_T^s .
- Type 2. Internal components** that do not contain vertices in J_1^d and J_T^s .
- Type 3. Residue components** that do not contain vertices in J_1^d and contain vertices in J_T^s .

In particular, there are no components that contain vertices in both J_1^d and J_T^s . To see this, let c be the restriction of \tilde{c} onto J_{bulk} . If $\tilde{e} + \tilde{c}$ contains a connected component that traverses from J_1^d to J_T^s , then $e + c$ must contain a connected component C that traverses from J_1^s to J_T^s , which means $|C| \geq d$. Since c is the minimum weight correction to e , we must have $|e \cap C| \geq |C|/2 \geq d/2$, which is ruled out by \mathcal{F}_1 .

Observe that the same argument implies that all internal components must have size less than d . By Lemma 6.17, these components are flattened to stabilizers. They induces no logical error and no residue syndrome. We can safely ignore them for the rest of this proof.

To bound the residue error y and therefore show the first condition, we can safely ignore input components and consider residue components. Let B be a set in $\mathcal{B}^{(R)}$. From Lemma 6.19, if y contains B , then there must be a cluster cover R of B in the extended graph $G_{\text{adj}}[H_{\text{ext}}]$ such that

1. $R \subseteq (e + c) \sqcup y$, and $|R| \geq 2|B|$;
2. For every cluster C of R , we have $|(e + c) \cap C| \geq |C|/2$. Note that C must contain some vertex of B , which is in J_{T+1}^d .

Taking the union over disjoint connected clusters C , we have

$$(e + c) \cap R \geq |R|/2 . \quad (6.41)$$

Due to minimality of c , we have $|e \cap R| \geq |R|/4$. For a set $B \subseteq J_{T+1}^d$, recall that $\mathcal{CO}(r, B)$ denote the collection of cluster covers of B of size at least r in $G_{\text{adj}}[H_{\text{ext}}]$. Define

$$\mathcal{F}_2 = \boxplus_{B \subseteq J_{T+1}^d, B \in \mathcal{B}^{(R)}} \boxplus_{r \geq 2|B|} \boxplus_{R \in \mathcal{CO}(r, B)} \{T \subseteq R : |T| = |R|/4, \text{ and } T \subseteq J_{\text{bulk}}\}. \quad (6.42)$$

We see that if y is not $\mathcal{B}^{(R)}$ -avoiding, then e must contain a set from \mathcal{F}_2 . In other words, if e is $(\mathcal{F}_1 \boxplus \mathcal{F}_2)$ -avoiding, then y is $\mathcal{B}^{(R)}$ -avoiding, as desired.

Next, we want to rule out the case that $\phi(\tilde{e} + \tilde{c}) + y \notin \text{rowsp}(H_X)$. As discussed earlier, internal components do not induce logical errors when flattened. For residue components, by Lemma 6.20, there must exists a connected cluster $C \subset (e + c) \sqcup y$ of size at least d in $G_{\text{adj}}[H_{\text{ext}}]$ (C needs to contain vertices in J_{T+1}^d) such that $|(e + c) \cap C| \geq |C|/2$. We have

$$|(e + c) \cap C| \geq |C|/2 \Rightarrow |e \cap C| \geq |C|/4 \geq d/4. \quad (6.43)$$

This is forbidden by the fact that e is \mathcal{F}_1 -avoiding.¹⁷ Therefore residue components cannot induce logical errors when flattened.

Finally, we consider input components. Let \mathcal{T} be the union of all input components. Recall that in our decoding process (Definition 6.21), we first computed c as a correction to e and then computed c_{in} as a correction to e_{in} . Therefore $\mathcal{T} \cap J_{\text{bulk}} = \mathcal{T} \cap (e + c)$, which has no spacetime syndrome under H_{bulk} . By Lemma 6.16, we have

$$H_Z \phi_{\text{bulk}}(\mathcal{T} \cap (e + c)) = (e + c)|_{J_1^s}. \quad (6.44)$$

Let $f_{\text{in}} \in \mathbb{F}_2^n$ be the minimum weight vector such that $H_Z f_{\text{in}} = (e + c)|_{J_1^s}$, then c_{in} is the minimum weight vector such that $H_Z c_{\text{in}} = H_Z(e_{\text{in}} + f_{\text{in}})$. We can now repeat our arguments above, as f_{in} is canonical to y . Since e is \mathcal{F}_1 -avoiding, we have that

$$\phi_{\text{bulk}}(\mathcal{T} \cap (e + c)) + f_{\text{in}} \in \text{rowsp}(H_X). \quad (6.45)$$

Moreover, define

$$\mathcal{F}_3 = \boxplus_{B \subseteq J_{1/10}^d, B \in \mathcal{B}_{1/10}^{(Z)}} \boxplus_{r \geq 2|B|} \boxplus_{R \in \mathcal{CO}(r, B)} \{T \subseteq R : |T| = |R|/4, \text{ and } T \subseteq J_{\text{bulk}}\}. \quad (6.46)$$

If e is \mathcal{F}_3 -avoiding, we know that f_{in} is $\mathcal{B}_{1/10}^{(Z)}$ -avoiding.

Now suppose e_{in} is $\mathcal{B}_{1/2}^{(Z)}$ -avoiding. By Corollary 6.12, $e_{\text{in}} + f_{\text{in}}$ is $\mathcal{B}_{6/10}^{(Z)}$ -avoiding, which implies that $e_{\text{in}} + f_{\text{in}} + c_{\text{in}} \in \text{rowsp}(H_X)$. Therefore,

$$\phi(\mathcal{T}) = e_{\text{in}} + c_{\text{in}} + \phi_{\text{bulk}}(\mathcal{T} \cap (e + c)) = (e_{\text{in}} + c_{\text{in}} + f_{\text{in}}) + (f_{\text{in}} + \phi_{\text{bulk}}(\mathcal{T} \cap (e + c))) \in \text{rowsp}(H_X). \quad (6.47)$$

In other words, input components cannot induce logical errors.

¹⁷This is not directly implied by the definition of \mathcal{F}_1 , since it is defined over connected components in the bulk graph, while C is a connected component in the extended graph. To prove our claim, we construct a connected cluster C' in the bulk graph. Let $z \in \mathbb{F}_2^n$ be the restriction of C to J_{T+1}^d . Let z' denote the same set (equivalently vector) of vertices as z , but supported over J_T^d . Let $C' = C|_{J_{\text{bulk}}} \cup z'$ (note that we used \cup instead of \sqcup here). Then C' is a connected component in the bulk graph of size at least d , and $|e \cap C'| = |e \cap C| \geq d/4$. This cannot happen since e is \mathcal{F}_1 -avoiding.

Putting things together, we define

$$\mathcal{F}_{\text{spacetime}} = \mathcal{F}_1 \boxplus \mathcal{F}_2 \boxplus \mathcal{F}_3, \quad (6.48)$$

and we have argued that if e is $\mathcal{F}_{\text{spacetime}}$ -avoiding, then both conditions hold. It remains for us to bound the weight enumerator of $\mathcal{F}_{\text{spacetime}}$. Let Δ_H denote the maximum degree of $G_{\text{adj}}[H_{\text{ext}}]$, we know that $\Delta_H = O(\Delta)$. By the same calculation as in Lemma 6.13, we have

$$\mathcal{W}(\mathcal{F}_1; x) \leq |J| \cdot [O_\Delta(x)]^{d/10}. \quad (6.49)$$

To bound \mathcal{F}_2 , we consider bad error supports in $\mathcal{B}^{(R)}$. Invoking Lemma 6.5,

$$\mathcal{W}(\mathcal{F}_2; x) \leq \sum_{B \in \mathcal{B}^{(R)}} \sum_{r \geq 2|B|} \sum_{R \in \mathcal{CO}(r, B)} 2^{|R|} x^{|R|/4} \quad (6.50)$$

$$\leq \sum_{B \in \mathcal{B}^{(R)}} \sum_{r \geq 2|B|} (\Delta_H e)^r 2^r x^{r/4} \quad (6.51)$$

$$\leq \sum_{B \in \mathcal{B}^{(R)}} [O_\Delta(x)]^{|B|/2} \quad (6.52)$$

$$\leq \mathcal{W}(\mathcal{B}^{(R)}; O_\Delta(x^{1/2})). \quad (6.53)$$

To bound \mathcal{F}_3 , we need to consider $\mathcal{B}_{1/10}^{(Z)}$.

$$\mathcal{W}(\mathcal{F}_3; x) \leq \sum_{B \in \mathcal{B}_{1/10}^{(Z)}} \sum_{r \geq 2|B|} \sum_{R \in \mathcal{CO}(r, B)} 2^r x^{r/4} \quad (6.54)$$

$$\leq \sum_{B \in \mathcal{B}_{1/10}^{(Z)}} \sum_{r \geq 2|B|} (\Delta_H e)^r 2^r x^{r/4} \quad (6.55)$$

$$\leq \mathcal{W}(\mathcal{B}_{1/10}^{(Z)}; O_\Delta(x^{1/2})) \quad (6.56)$$

$$\leq |J| \cdot [O_\Delta(x)]^{d/40}. \quad (6.57)$$

Therefore, putting everything together we see that

$$\mathcal{W}(\mathcal{F}_{\text{spacetime}}; x) \leq |J| \cdot [O_\Delta(x)]^{d/40} + \mathcal{W}(\mathcal{B}^{(R)}; O_\Delta(x^{1/2})). \quad \square$$

6.3 Error correction gadget

Theorem 6.23 (Error correction gadget [Den+02; Got14]). We can construct a gadget $\text{Gad}_{\text{EC}} = (V, E, \text{conn}, \text{type}, \text{gate})$ of depth $O(d)$ for the computational code with the following guarantee.

1. For input code type $\text{CompCode}_{1/2} = (U, \mathcal{B}_{1/2})$ and any output code type $(U, \mathcal{B}^{(R)})$, let $\mathcal{B}_1^{(R)}, \mathcal{B}_2^{(R)}$ be two collections^a such that $\mathcal{B}^{(R)}$ is separable into $\mathcal{B}_1^{(R)}, \mathcal{B}_2^{(R)}$ as defined in Definition 6.10. There is a family of bad fault paths \mathcal{F}_{EC} such that Gad_{EC} is a fault-tolerant gadget for identity with respect to the input/output code types and \mathcal{F}_{EC} .

2. This family of bad fault paths \mathcal{F}_{EC} has weight enumerator

$$\mathcal{W}(\mathcal{F}_{\text{EC}}; x) \leq d \cdot n (O_{\Delta}(x))^{\frac{d}{80\Delta}} + \mathcal{W}(\mathcal{B}_1^{(R)}; O_{\Delta}(x^{\frac{1}{4\Delta}})) + \mathcal{W}(\mathcal{B}_2^{(R)}; O_{\Delta}(x^{\frac{1}{2\Delta}})) . \quad (6.58)$$

^aHere we separate the output error $\mathcal{B}^{(R)}$ into two collections to account for two sources of residue errors: one from the decoder output as in Lemma 6.22, and another from the last round of syndrome extraction circuit.

Construction. Syndrome extraction circuit. First, we construct a constant-depth syndrome extraction circuit for the code in the standard way. A Δ -qLDPC code defined by the check matrices (H_X, H_Z) has a depth $\Delta + 2$ syndrome extraction circuit each for the X syndrome and the Z syndrome. Let us consider the case of measuring the Z syndrome (X type operators associated with H_X). The case of the X syndrome is identical with CNOT replaced by CZ and H_X replaced by H_Z .

First, compute an edge coloring of the Tanner graph of $H_Z \in \mathbb{F}_2^{r \times n}$: Define the bipartite graph $G = ([r] \sqcup [n], E)$ where two vertices $(i, j) \in [r] \times [n]$ are adjacent if $H[i, j] = 1$. G has maximum degree Δ by the Δ -qLDPC property. The edge coloring of G is a map $\phi: E \mapsto [\Delta]$ such that no vertex has two edges of the same color incident to it. Equivalently, for each color $c \in [\Delta]$, the subset of edges $\phi^{-1}(c)$ is a matching in G . G is bipartite so such a map exists and is efficiently computable [Sch+03].

A single round of syndrome extraction is as follows:

1. Prepare r ancilla qubits in $|+\rangle$.
2. For each color $c \in [\Delta]$, perform the gate layer:
 - For each edge $(i, j) \in \phi^{-1}(c)$, apply CNOT(i, j) (controlled on the ancilla qubit i)
3. Measure all r ancilla qubits in the X basis. The i -th measurement outcome is the eigenvalue of the X operator associated with the i -th row of the check matrix H_X .

Let us refer to a single round of syndrome Z and X -type syndrome extraction as the circuit M . The full error correction gadget alternates layers of Z and X -type syndrome extraction rounds T (to be picked later) rounds.

That is, the gadget applies M for T times where each application of M should be interpreted as producing an Z and X measurement vector $m_1, \dots, m_T \in \mathbb{F}_2^{r_X + r_Z}$. ◀

Proof. From these measurements, we will compute a correction and apply it.

Decoding. Let r_X be the number of rows of H_X and r_Z be the number of rows of H_Z . The syndrome extraction circuit produces a pair of measurements $m_X \in \mathbb{F}_2^{r_X \times T}$ and $m_Z \in \mathbb{F}_2^{r_Z \times T}$. We take pairwise differences to obtain syndromes s_X and s_Z .

$$s_{X/Z}[I_1] = m_{X/Z}[I_1] \quad (6.59)$$

$$t \in [T-1], \quad s_{X/Z}[I_{t+1}] = m_{X/Z}[I_t] + m_{X/Z}[I_{t+1}] \quad (6.60)$$

In what follows, to reduce notation, we will not distinguish between diagonal Pauli superoperators and Pauli operators. Additionally, for a bit string $x \in \mathbb{F}_2^m$, we will also use x to denote the state

$|x\rangle\langle x|$ when clear from context. For now, suppose that the fault \mathbf{f} is a diagonal Pauli fault and the input state ρ_0 is \mathcal{B}_c -deviated from a state $\text{enc}(\sigma)$ by a diagonal Pauli superoperator E_0

$$\text{map}[M^{\otimes t}[\mathbf{f}]](\rho_0) = \rho_t \otimes m_X[I_1] \otimes m_Z[I_1] \otimes \cdots \otimes m_X[I_t] \otimes m_Z[I_t] \quad (6.61)$$

be the state (including measurement outputs) after the t -th round of Z and X syndrome extraction, and E_t be the diagonal Pauli superoperator supported on the code block that takes ρ_{t-1} to ρ_t , $E_t(\rho_{t-1}) = \rho_t$. Let us write ξ for the syndrome map $\mathcal{P}^n \rightarrow \mathbb{F}_2^{r_X+r_Z}$. For $t \in [T]$, define $\tilde{m}_t = m_t - \xi(E_{t-1} \dots E_0)$ to be the difference between the measured syndrome and syndrome of the state input to the t -th application of M , and decompose $E_t = E_t^{(X)} E_t^{(Z)}$ into the product of Pauli X and Pauli Z . Using that, in the absence of any further faults, the measurement output is the syndrome of the state and that ξ is a group homomorphism, $\xi(AB) = \xi(A) + \xi(B)$, we can see that the syndrome differences (Eq. (6.59)) give the syndrome of any new errors applied up to measurement errors.

$$s_{X/Z}[I_1] = \xi(E_0^{(Z/X)}) + \tilde{m}_{X/Z}[I_1] \quad (6.62)$$

$$t \in [T-1], \quad s_{X/Z}[I_{t+1}] = \xi(E_{t-1}^{(Z/X)}) + \tilde{m}_{X/Z}[I_t] + \tilde{m}_{X/Z}[I_{t+1}] \quad (6.63)$$

For convenience, we will decode these syndromes separately. We focus on decoding X errors with syndrome s_Z . The case of Z errors is identical.

We now decode the syndrome s_Z with respect to the spacetime code H of H_Z , as specified by the decoding procedure in Definition 6.21, to produce a correction $c_X \in \mathbb{F}_2^{(Tn+Tr_z)}$. We apply an X correction to the i -th qubit when the i -th bit of the flattening ϕ of the correction c_X is 1, $C_X = X^{\phi(c_X)}$.

The overall gadget Gad_{EC} is then given in figure Fig. 7. (Runtime of the decoder ignored.)

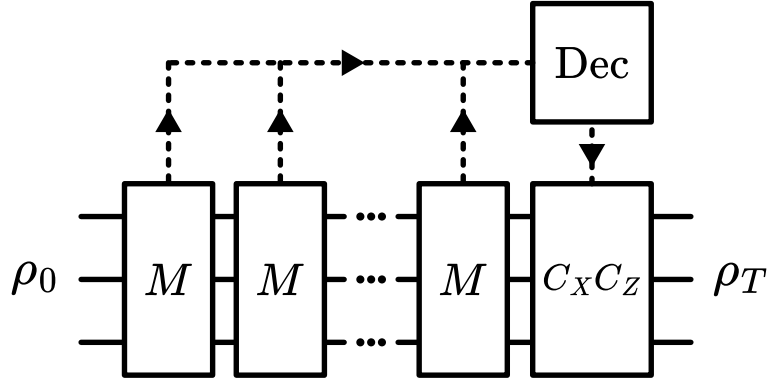


Figure 7: Error correction gadget Gad_{EC} , the circuit M is repeated T times.

Fault analysis. We would now like to analyze the residual error. We will set $T = d$. First note that there is an error $e \in \mathbb{F}_2^{(Tn+Tr_z)}$ of the spacetime code corresponding to the error that in the

circuit that yields the measured syndrome differences.

$$t \in [T], \quad E_{t-1}^{(X)} = X^{e[J_t^d]} \quad (6.64)$$

$$t \in [T], \quad \tilde{m}_Z[I_t] = e[J_t^s] \quad (6.65)$$

With the exception of the final round error, the flattening also matches the residual error: $X^{\phi(e)} = E_{T-1}^{(X)} \dots E_0^{(X)}$. We now utilize the obvious bijection between the supports of $E_{T-1}^{(X)}, \dots, E_0^{(X)}, \tilde{m}_Z$, and bits of the spacetime code $[Tn + Tr_z]$. For a bit $i \in [Tn + Tr_z]$, let $\chi_i \subseteq V$ denote the set of locations of Gad_{EC} for which a fault may flip bit i . A fault supported on a single location may flip at most $s = 2\Delta$ bits of the spacetime code: A two-qubit gate failure affects a data and ancilla qubit, and each of these has a gate shared with $\Delta - 1$ other qubits.¹⁸ Likewise, a single bit of the spacetime code may be flipped by a fault at one of at most $\Delta(2\Delta + 4)$ locations.

Define a bipartite graph $G = (V \sqcup [Tn + Tr], E)$ between the locations of the gadget and the spacetime code bits. Bit i is connected to spacetime location v if $v \in \chi_i$. G has left degree at most 2Δ and right degree at most $\Delta(2\Delta + 4)$. We can now define a family of bad fault in the gadget corresponding to the bad spacetime code errors. Let $\mathcal{F}_{\text{EC},1}^{(X)}$ be the family of bad error supports in the statement of Lemma 6.22, with the guarantee that the residue error is $\mathcal{B}_1^{(R)}$ -avoiding. For a subset of vertices $S \subseteq [Tn + Tr]$, let $N(S)$ denote the neighborhood of S in G . If a location is capable of flipping a particular spacetime code bit then the two must be adjacent in G .

$$\mathcal{F}_{\text{EC},1}^{(X)} = \boxplus_{f \in \mathcal{F}_{\text{spacetime}}^{(X)}} \{f' \subseteq N(f) \mid \forall u \in f, |N(u) \cap f'| > 0\} \quad (6.66)$$

There are at most $2^{|f|\Delta(2\Delta+4)}$ subsets of $N(f)$ and each subset must have weight at least $|f|/(2\Delta)$. Thus for $x \in [0, 1]$, we can upper bound the weight enumerator as

$$\mathcal{W}(\mathcal{F}_{\text{EC},1}^{(X)}; x) \leq \mathcal{W}(\mathcal{F}_{\text{spacetime}}^{(X)}; 2^{\Delta(2\Delta+4)} x^{\frac{1}{2\Delta}}) \quad (6.67)$$

We repeat this construction with X and Z interchanged to obtain $\mathcal{F}_{\text{EC},1}^{(Z)}$.

We also need to bound the error caused by the last application of M that propagates to E_T . We can perform an identical argument with $\mathcal{F}_{\text{spacetime}}^{(X)}$ replaced by $\mathcal{B}_2^{(R)}$ and bits of the spacetime code replaced by E_T .

$$\mathcal{W}(\mathcal{F}_{\text{EC},2}; x) \leq \mathcal{W}(\mathcal{B}_2^{(R)}; 2^{\Delta(2\Delta+4)} x^{\frac{1}{2\Delta}}) \quad (6.68)$$

Our bad fault paths are

$$\mathcal{F}_{\text{EC}} = \mathcal{F}_{\text{EC},1}^{(X)} \boxplus \mathcal{F}_{\text{EC},1}^{(Z)} \boxplus \mathcal{F}_{\text{EC},2} . \quad (6.69)$$

Now, let \mathbf{f} be a \mathcal{F}_{EC} -avoiding diagonal Pauli fault. In the execution of $M[\mathbf{f}]$ on a state ρ_0 , the spacetime code error e (Eq. (6.64)) is $\mathcal{F}_{\text{EC},1}^{(X)}$ -avoiding in the Z -type syndrome extraction and $\mathcal{F}_{\text{EC},1}^{(Z)}$ -avoiding in the X -type syndrome extraction. In the case where E_0 is $\mathcal{B}_{1/2}$ -avoiding, it follows by

¹⁸These are the well known ‘‘hook errors.’’ A more careful counting would show that the majority of the spread of the error does not intersect any of the sets of interest in so many places.

Lemma 6.22 that $C_X C_Z E_{T-1} \dots E_0$ is $\mathcal{B}_1^{(R)}$ -avoiding. Otherwise, there exists a logical operator of the code L such that $LC_X C_Z E_{T-1} \dots E_0$ is $\mathcal{B}_1^{(R)}$ -avoiding (friendly).

The final round of errors E_T has $\mathcal{B}_2^{(R)}$ -avoiding support, by the separability of $\mathcal{B}^{(R)}$ as $\mathcal{B}_1^{(R)}$ and $\mathcal{B}_2^{(R)}$, the output is $\mathcal{B}^{(R)}$ -deviated from the codespace. In the case that the input is $\mathcal{B}_{1/2}$ -deviated from the encoding of a logical state $\text{enc}(\sigma)$, then the output is $\mathcal{B}^{(R)}$ -deviated from $\text{enc}(\sigma)$.

We now drop the assumption that the faults are diagonal: For a general input error and faults (non-diagonal Pauli) Lemma 4.14 can be applied at each step, so that the errors at each step is either equivalent to a diagonal Pauli error (the left and right sides are stabilizer equivalent) up to some part in the lightcone of the fault in that round or the state is annihilated by the measurement projectors (so the deviation property trivially holds with coefficient 0).

We now plug in the bounds on the weight enumerators to obtain (recall T was picked to be d so there are at most $\propto nd$ locations in the spacetime graph.)

$$\mathcal{W}(\mathcal{F}_{\text{EC}}; x) \leq d \cdot n [O_{\Delta}(x)]^{\frac{d}{80\Delta}} + \mathcal{W}(\mathcal{B}_1^{(R)}; O_{\Delta}(x^{\frac{1}{4\Delta}})) + \mathcal{W}(\mathcal{B}_2^{(R)}; O_{\Delta}(x^{\frac{1}{2\Delta}})) . \quad (6.70)$$

Following the definition of a friendly, fault-tolerant gadget Definition 4.8, we see that this entire error correction gadget implements the filter superoperator $\mathcal{R}_{\mathbf{f}}$ and an identity logical gate, $\tilde{\mathcal{G}}_{\mathbf{f}} = \mathbf{I}$. \square

Remark 6.24 (Friendliness). Through Lemma 6.22, we have argued that the error correction gadget is friendly. I.e., when the input error is bad, the residue error of the output state is still bounded, although the logical state is erroneous. By a closer inspection of the gadget construction and decoding, we note that the EC gadget is *friendly in both the X and Z bases*. More precisely, suppose the input error is a diagonal Pauli error $P_X P_Z$. If P_X has $\mathcal{B}_{1/5}$ -avoiding support while P_Z is arbitrarily supported, the logical error L will be a Z error. Similarly, if P_Z has $\mathcal{B}_{1/5}$ -avoiding support while P_X is arbitrarily supported, the logical error L will be a X error.

In later constructions of fault-tolerance schemes, we will often construct a logical gate gadget by first applying this error correction gadget, followed by the corresponding logical operation. The constructed gadgets are therefore friendly and the error support on the input state to the logical operation can be easily bounded.

Remark 6.25 (Single-shot decoding). The error correction gadget constructed above repeatedly measure the syndrome information for $O(d)$ rounds, which protects the encoded information from measurement errors. For most codes this repetition is necessary to control the residue error distribution. There are families of qLDPC codes which can be single-shot decoded [Bom15; KV22; BKV24; FGL20; Gu+24; NP24]. More precisely, these codes have efficient decoders which only requires $O(1)$ rounds of syndrome information to compute corrections with logical error rates $e^{-O(d)}$ and controlled residue errors. Using these codes as computation code could improve the depth overhead of fault-tolerant gadgets. It is worth noting, however, that an error correction gadget constructed from a single-shot decoder may not be friendly. For instance, the decoder of [Gu+24], which is a modified version of the decoder proposed by [LZ23], assumes that the input qubit and measurement error weights are both upper bounded by cn for some constant c (the decoded code is asymptotically good). For the bad input state where this assumption is not

satisfied, the decoder makes no guarantee on the output state and residue error. Conceptually this can be seen as a *logical leakage error*, where not only is the logical information corrupted, the logical qubit itself is also lost and requires reinitialization (for instance, by a friendly gadget).

Remark 6.26 (Code switching). The error correction gadget we constructed measures the same set of stabilizers repeatedly and implements a logical identity. Alternatively, by measuring different groups of stabilizers corresponding to different codes, we can fault-tolerantly switch from one code (and code type) to another [BMD09]. Such *code-switching* protocols are highly versatile. One motivating example is to switch between a two-dimensional topological code which admits transversal Clifford gates [CS96; Ste96; BMD06] and a three-dimensional topological code which admits transversal non-Clifford gates [Bom15; KB15; PR13; KYP15], see [Bom16]. This enables one to utilize two computational codes and implement a universal gate set transversally.

Another well-studied type of code-switching is *code surgery*, which implements multi-qubit Pauli measurement on logical qubits. Such multi-qubit Pauli measurement, when supplied with magic states, can be used to implement universal quantum computation in the framework of Pauli-based computation (PBC) [BSS16]. The prototypical example of code surgery is lattice surgery [Hor+12; Fow+12] on surface codes, which can be applied to design fault-tolerant surface code architectures with two-dimensional nearest neighbor connectivity [FG18; Lit19]. These methods are later generalized to other qLDPC codes [Has17; Has21; Coh+22] under the name *qLDPC surgery*. While the first proposals of qLDPC surgery [Coh+22] incurs a heavy space overhead, recent works [Cow24; Cro+24; WY24; Ide+24; SJOY24; ZL24] presented improved proposals with significantly lower theoretical and practical overheads. These methods are subsequently improved and applied in [Cow+25; He+25; Yod+25]. In particular, the extractor architectures proposed in [He+25] enables fault-tolerant PBC on arbitrary qLDPC codes with low-overhead and fixed connectivity. It would be interesting to prove the fault-tolerance of these surgery gadgets in our weight enumerator formalism.

The aforementioned examples of code-switching can all be described using the language of subsystem codes and gauge-fixing [Vui+19]. In contrast, the dynamical codes proposed in [HH21] demonstrated that logical information can be stored and operated on in a continuous sequence of code-switching [Dav+24]. It would be interesting to describe the fault-tolerance properties of these dynamically generated logical qubits in terms of weight enumerators as well.

6.4 Constant depth gadgets

Gate gadgets for transversal gates are a short corollary.

Corollary 6.27 (Transversal gate gadget). Let \mathcal{G} be a gate on either one or two computational code blocks that can be implemented transversally. Then, there exists a gadget for \mathcal{G} with input and output code types $\text{CompCode}_{1/5}$ and a family of bad fault paths $\mathcal{F}_{\mathcal{G}}$ where

$$\mathcal{W}(\mathcal{F}_{\mathcal{G}}; x) \leq 2\mathcal{W}(\mathcal{F}_{\text{EC}}; x) + \mathcal{W}(\mathcal{B}_{1/10}; x) \leq d \cdot n \cdot [O_{\Delta}(x)]^{\frac{d}{80\Delta}}. \quad (6.71)$$

Proof. We consider the more complicated case of two code blocks.

We execute the error correction gadget Theorem 6.23 on the two inputs block, setting $\mathcal{B}^{(R)} = \mathcal{B}_{1/5}$ separable into two copies of $\mathcal{B}_{1/10}$. From Lemma 6.13, we know

$$\mathcal{W}(\mathcal{B}_{1/10}; x) \leq n \cdot [O_{\Delta}(x)]^{\frac{d}{20}}. \quad (6.72)$$

Combined with Eq. (6.58), we have an upper bound on the weight enumerator of \mathcal{F}_{EC} .

$$\mathcal{W}(\mathcal{F}_{\text{EC}}; x) \leq d \cdot n \cdot [O_{\Delta}(x)]^{\frac{d}{80\Delta}}. \quad (6.73)$$

We pick $\mathcal{F}_{\mathcal{G}}$ to be the sum of \mathcal{F}_{EC} on the two error correction gadgets and $\mathcal{B}_{1/10}$ on the transversal gates. The upper bound in our claim follows.

Assume the fault is $\mathcal{F}_{\mathcal{G}}$ -avoiding. Then, if the input state ρ is $\mathcal{B}_{1/2} \boxplus \mathcal{B}_{1/2}$ -deviated from a logical state $(\text{enc} \otimes \text{enc})(\sigma)$, then the output of the error correction gadgets is $\mathcal{B}_{1/5} \boxplus \mathcal{B}_{1/5}$ -deviated from $(\text{enc} \otimes \text{enc})(\sigma)$. The transversal gate gadget can at most create a $\mathcal{B}_{1/10}$ -avoiding error and spread a $\mathcal{B}_{1/5}$ -avoiding error to each block. It follows by Corollary 6.12 that the output state is $\mathcal{B}_{1/2} \boxplus \mathcal{B}_{1/2}$ -deviated from $(\text{enc} \otimes \text{enc}) \circ \mathcal{G}(\sigma)$.

For an arbitrary input state, the output of the error correction gadgets is still $\mathcal{B}_{1/5} \boxplus \mathcal{B}_{1/5}$ -deviated from the code space. \square

Note that we precede all transversal gates with a $O(d)$ -round error correction gadget, which incurs a notable time (equivalently depth) overhead in theoretical and practical fault-tolerance. Alternatively, we could interleave transversal gates with syndrome extraction rounds to amortize the time overhead of a full error correction gadgets across multiple logical operations. This approach forms the backbone of *transversal algorithmic fault-tolerance* as proposed and studied in [Zho+24]. It is an interesting future direction to formulate this approach in terms of gadgets and weight enumerators.

Beyond transversal gates, we further consider constant depth gates supported on one computational code block, and show the standard claim that constant depth gates are fault tolerant via a lightcone argument.

Lemma 6.28 (Constant depth gate gadget). Let \mathcal{G} be a gate on one computational code block that can be implemented by a circuit $C_{\mathcal{G}}$ of depth T . Set $c = \frac{1}{5(T+1)2^{T+1}}$. Then there exists a gadget for \mathcal{G} with input and output code type $\text{CompCode}_{1/5}$ and a family of bad fault paths $\mathcal{F}_{\mathcal{G}}$ where

$$\mathcal{W}(\mathcal{F}_{\mathcal{G}}; x) \leq dn \cdot [O_{\Delta}(x)]^{\frac{d}{80\Delta}} + n \cdot [O_{\Delta,T}(x)]^{\frac{cd}{16\Delta}} + Tn \cdot [O_{\Delta,T}(x)]^{\frac{cd}{2}}. \quad (6.74)$$

Proof. We will first apply the error correction gadget Theorem 6.23 on the code block, with $\mathcal{B}^{(R)}$ to be specified later. The residue error being $\mathcal{B}^{(R)}$ -avoiding and the fault path on $C_{\mathcal{G}}$ being $\mathcal{F}_{\mathcal{G}}$ -avoiding will guarantee that the final output error is $\mathcal{B}_{1/5}$ -avoiding.

We use a lightcone argument to define $\mathcal{B}^{(R)}$ and $\mathcal{F}_{\mathcal{G}}$. Since $C_{\mathcal{G}} = (V, E, \text{conn}, \text{type}, \text{gate})$ acts on the code qubits labelled by $[n]$ and has depth T , we partition the edges in $C_{\mathcal{G}}$ by depth as $E = \cup_{t=1}^{T+1} E_t$, where E_1, E_{T+1} are the input and output qubits respectively and each E_t is a copy of $[n]$. We

denote the qubits in E_t as e_i^t , for $i \in [n]$, with the implicit identification that $\{e_i^t\}_{t=1}^{T+1}$ denote the same physical qubit at different time. For a qubit $e_i^t \in E_t$, the *forward lightcone* of e_i^t , denoted $\mathcal{LC}(e_i^t)$, are all qubits $e' \in E_{T+1}$ such that there is a directed path from e to e' in the directed network of $C_{\mathcal{G}}$. Observe that for a qubit e_i and two time steps $t_1 < t_2$, we have $\mathcal{LC}(e_i^{t_1}) \supseteq \mathcal{LC}(e_i^{t_2})$. Note further that for all $i \in [n]$, $|\mathcal{LC}(e_i^1)| \leq 2^T$ since we assumed that every gate in $C_{\mathcal{G}}$ acts on at most two qubits.

To account for error propagation through the circuit $C_{\mathcal{G}}$, we define the *lightcone adjacency graph*, which we denote $G_{\mathcal{LC}}$. $G_{\mathcal{LC}}$ has vertices corresponding to qubits $[n]$ of the computational code. Given the stabilizer check matrices H_X, H_Z , we add an edge between any pair of two qubits which are acted on by at least one common stabilizer. In other words, $G_{\text{adj}}[H_X], G_{\text{adj}}[H_Z]$ are both subgraphs of $G_{\mathcal{LC}}$. Further, for every qubit e_i , for every qubit e_j such that $e_j^{T+1} \in \mathcal{LC}(e_i^1)$, we add an edge between e_i, e_j . This completes our definition of $G_{\mathcal{LC}}$. The maximum degree of $G_{\mathcal{LC}}$, denoted $\Delta_{\mathcal{LC}}$, satisfies $\Delta_{\mathcal{LC}} = O_{\Delta, T}(1)$.

We now bound the output error support $B^{\text{out}} \subseteq E_{T+1}$ using the support of the input error $B_1 \subseteq E_1$ and the fault path $F \subseteq V$. Recall that the vertices of $C_{\mathcal{G}}$ is partitioned into depth, $V = \cup_{t=1}^T V_t$. We similarly partition the fault path $F = \cup_{t=1}^T F_t$, where $F_t = F \cap V_t$. For each F_t , let $B_{t+1} \subseteq E_{t+1}$ denote the output qubits of vertices in F_t , and note that $|B_{t+1}| \leq 2|F_t|$. We define the lightcone of a set of qubits simply as the union of lightcones of individual qubits.

$$\mathcal{LC}(B) := \cup_{e \in B} \mathcal{LC}(e). \quad (6.75)$$

By the definition of lightcone, we note that faults on a set of qubits $B \subseteq E$ can propagate to only qubits in $\mathcal{LC}(B) \subseteq E_{T+1}$. This implies that

$$B^{\text{out}} \subseteq \mathcal{LC}\left(\bigcup_{t=1}^{T+1} B_t\right) = \bigcup_{t=1}^{T+1} \mathcal{LC}(B_t). \quad (6.76)$$

It suffices for us to bound the error supports of $\mathcal{LC}(B)$. We define bad error supports $\mathcal{B}_c^{\mathcal{LC}} \subseteq P([n])$ according to the graph $G_{\mathcal{LC}}$. Let $\mathcal{CG}_{m, m'}$ denote the (m, m') -clustering sets in $G_{\mathcal{LC}}$. Similar to Definition 6.8, we define

$$\mathcal{B}_c^{\mathcal{LC}} = \boxplus_{m=d}^n \mathcal{CG}_{m, \lceil cm/2 \rceil}. \quad (6.77)$$

From the definition of $G_{\mathcal{LC}}$, we have the following claim.

Claim 6.29. For all $t \leq T+1$, if $B \subseteq E_t$ is $\mathcal{B}_{c/2^{T+1}}^{\mathcal{LC}}$ -avoiding, then $\mathcal{LC}(B) \subseteq E_{T+1}$ is \mathcal{B}_c -avoiding.

Proof of claim. Suppose for the sake of contradiction $\mathcal{LC}(B)$ contains a set $S \subseteq E_{T+1} \cong [n]$ such that S is a subset of a connected set \bar{S} of size $m \geq d$ and $|S| \geq \lceil cm/2 \rceil$. Since \bar{S} is a connected set in $G_{\text{adj}}[H_X]$ or $G_{\text{adj}}[H_Z]$, which are both subgraphs of $G_{\mathcal{LC}}$, \bar{S} must be connected in $G_{\mathcal{LC}}$ as well. Let $N(\bar{S})$ denote the neighborhood of \bar{S} in $G_{\mathcal{LC}}$. Then

$$|B \cap (N(\bar{S}) \cup \bar{S})| \geq \frac{|S|}{2^T} \geq \frac{c|\bar{S}|}{2^{T+1}}, \quad (6.78)$$

since every lightcone has size bounded by 2^T . For convenience let $R := B \cap (N(\bar{S}) \cup S)$. Then $R \cup \bar{S}$ is a connected set of vertices in $G_{\mathcal{LC}}$ of size at least d , and

$$\frac{c|R \cup \bar{S}|}{2^{T+2}} \leq \frac{c|\bar{S}|}{2^{T+2}} + \frac{c|R|}{2^{T+2}} \leq |R|. \quad (6.79)$$

This contradicts our assumption that B is $\mathcal{B}_{c/2^{T+1}}^{\mathcal{LC}}$ -avoiding, which proves our claim. ■

Set $c = \frac{1}{5(T+1)2^{T+1}}$. Following Corollary 6.12, we see that if B_t is $\mathcal{B}_c^{\mathcal{LC}}$ -avoiding for all $t \leq T+1$, then B^{out} is $\mathcal{B}_{1/5}$ -avoiding. Therefore, we set $\mathcal{B}^{(R)} = \mathcal{B}_c^{\mathcal{LC}}$, which is separable as two copies of $\mathcal{B}_{c/2}^{\mathcal{LC}}$. The EC gadget, Theorem 6.23, gives us \mathcal{F}_{EC} with bounded weight enumerator.

$$\mathcal{W}(\mathcal{F}_{\text{EC}}; x) \leq dn \cdot [O_{\Delta}(x)]^{\frac{d}{80\Delta}} + 2\mathcal{W}\left(\mathcal{B}_{c/2}^{\mathcal{LC}}; O_{\Delta}(x^{\frac{1}{4\Delta}})\right) \quad (6.80)$$

$$\leq dn \cdot [O_{\Delta}(x)]^{\frac{d}{80\Delta}} + n \cdot [O_{\Delta,T}(x)]^{\frac{cd}{16\Delta}}. \quad (6.81)$$

We further define the bad fault paths on the circuit $C_{\mathcal{G}}$, \mathcal{F}_C , as follows. For every $t \leq T$, for $e \in E_{t+1}$, let $s(e) \subseteq V_t$ denote the locations at time t that has e as an output qubit. We define

$$\mathcal{F}_{C,t} = \boxplus_{B \in \mathcal{B}_c^{\mathcal{LC}}} \{F_t \subseteq V_t \mid \forall e \in B, |s(e) \cap F_t| > 0\}. \quad (6.82)$$

There are at most $2|B|$ locations in V_t with output qubits overlapping with B , so there are at most $2^{2|B|}$ such sets F_t . Each set must have weight at least $|B|/2$. Therefore,

$$\mathcal{W}(\mathcal{F}_{C,t}; x) \leq \mathcal{W}\left(\mathcal{B}_c^{\mathcal{LC}}; 4x^{\frac{1}{2}}\right). \quad (6.83)$$

We define \mathcal{F}_C to be the product of $\mathcal{F}_{C,t}$.

$$\mathcal{F}_C = \boxplus_{t=1}^T \mathcal{F}_{C,t} \quad (6.84)$$

Invoking Lemma 6.13, the weight enumerator can be bounded as

$$\mathcal{W}(\mathcal{F}_C; x) \leq T\mathcal{W}\left(\mathcal{B}_c^{\mathcal{LC}}; 4x^{\frac{1}{2}}\right) \quad (6.85)$$

$$\leq Tn \cdot [O_{\Delta,T}(x)]^{\frac{cd}{2}}. \quad (6.86)$$

Define $\mathcal{F}_{\mathcal{G}} = \mathcal{F}_C \boxplus \mathcal{F}_{\text{EC}}$, and the bound on its weight enumerator follows. □

Lemma 6.30 (CSS Code Initialization). Let INIT_Z be the gate that initializes k qubits to the $|0\rangle$ state. There exists a gadget for INIT_Z with trivial input code type and output code type $\text{CompCode}_{1/5}$ and a family of bad fault paths \mathcal{F} where

$$\mathcal{W}(\mathcal{F}; x) \leq d \cdot n \cdot [O_{\Delta}(x)]^{\frac{d}{80\Delta}}. \quad (6.87)$$

Proof. We use $\text{INIT}_Z^{\otimes n}$ to initialize n qubits to the $|0\rangle$ state, and then apply the error correction gadget Theorem 6.23 with $\mathcal{B}^{(R)} = \mathcal{B}_{1/5}$ separable into two copies of $\mathcal{B}_{1/10}$. Define the bad fault paths $\mathcal{F}_{\text{INIT}}$ on the locations of the transversal initialization to be $\mathcal{B}_{1/10}$. We define the overall bad fault paths \mathcal{F} to be $\mathcal{F}_{\text{INIT}} \boxplus \mathcal{F}_{\text{EC}}$. The upper bound on the weight enumerator then follows from the same analysis as in Corollary 6.27.

It remains for us to show that this gadget behaves correctly. We make use of the following fact.

Fact 6.31. For a CSS code with X parity check matrix H_X , let m be the dimension of $\text{rowsp}(H_X)$, i.e., $|\text{rowsp}(H_X)| = 2^m$. Let $|0_L\rangle^{\otimes k}$ denote the logical $|0^k\rangle$ state of the code. It holds that

$$|0\rangle^{\otimes n} = \frac{2^{m/2}}{2^n} \sum_{v \in \mathbb{F}_2^n} Z^v |0_L\rangle^{\otimes k}. \quad (6.88)$$

As a corollary, it holds that

$$|0\rangle\langle 0|^{\otimes n} = \frac{1}{2^{2n-m}} \sum_{u, v \in \mathbb{F}_2^n} Z^u \text{enc}(|0\rangle\langle 0|^{\otimes k}) Z^v. \quad (6.89)$$

Proof of claim. Recall that for a CSS code,

$$|0_L\rangle^{\otimes k} = \frac{1}{2^{m/2}} \sum_{r \in \text{rowsp}(H_X)} |r\rangle. \quad (6.90)$$

We can thereby compute

$$\frac{2^{m/2}}{2^n} \sum_{v \in \mathbb{F}_2^n} Z^v |0_L\rangle^{\otimes k} = \frac{1}{2^n} \sum_{r \in \text{rowsp}(H_X)} \left(\sum_{v \in \mathbb{F}_2^n} Z^v |r\rangle \right) \quad (6.91)$$

$$= \frac{1}{2^n} \sum_{r \in \text{rowsp}(H_X)} \left(\sum_{v \in \mathbb{F}_2^n} (-1)^{v \cdot r} |r\rangle \right) \quad (6.92)$$

For $r \neq 0^n$, the sum $\sum_{v \in \mathbb{F}_2^n} (-1)^{v \cdot r} |r\rangle$ evaluates to 0. The only term that remains is $|0\rangle^{\otimes n}$, which proves this claim. ■

Let \mathbf{f} be a \mathcal{F} -avoiding Pauli fault, which can be divided into \mathbf{f}_1 supported on locations of $\text{INIT}_Z^{\otimes n}$ and \mathbf{f}_2 supported on locations of the error correction gadget. Suppose the faulty circuit $\text{INIT}_Z^{\otimes n}[\mathbf{f}_1]$ introduces an error \mathcal{E} , whose support is $\mathcal{B}_{1/5}$ -avoiding, onto the state $|0\rangle\langle 0|^{\otimes n}$. In the error correction gadget, we repeatedly measure the stabilizers of the CSS code, which enables us to invoke a variant of the decoherence of errors lemma (Lemma 4.14) on the input state. Specifically, consider the Kraus decomposition of \mathcal{E} in terms of Pauli operators $\{K_\mu\}_\mu, \{K'_\nu\}_\nu$ and complex coefficients $\{\alpha_{\mu,\nu}\}_{\mu,\nu}$. The measurement channel can be written as

$$\mathcal{M}(\rho) = \sum_{\text{syndrome } s} |s\rangle\langle s| \otimes \Pi_s \rho \Pi_s, \quad (6.93)$$

where Π_s is the projector onto the syndrome eigenspace corresponding to s .

$$\Pi_s \mathcal{E}(|0\rangle\langle 0|^{\otimes n}) \Pi_s = \frac{1}{2^{2n-m}} \sum_{\mu, \nu} \alpha_{\mu, \nu} \Pi_s K_\mu \left(\sum_{u, v \in \mathbb{F}_2^n} Z^u \text{enc}(|0\rangle\langle 0|^{\otimes k}) Z^v \right) K'_\nu \Pi_s \quad (6.94)$$

$$= \frac{1}{2^{2n-m}} \sum_{\mu, \nu, u, v} \alpha_{\mu, \nu} \Pi_s K_\mu Z^u \text{enc}(|0\rangle\langle 0|^{\otimes k}) Z^v K'_\nu \Pi_s. \quad (6.95)$$

As in the proof of Lemma 4.14, only terms where both $K_\mu Z^u$ and $K'_\nu Z^v$ has the same syndrome s passes the projectors while the other terms are annihilated. Moreover, since we have a CSS code, the X and Z errors have independent syndromes. All X errors must come from the operators K_μ, K'_ν , whose support is $\mathcal{B}_{1/5}$ -avoiding and therefore recoverable. As in the proof of Lemma 4.14, the X terms in $K_\mu Z^u$ and $K'_\nu Z^v$ can be chosen to be the same up to stabilizers. Denote this X error $P_{X,s}$. Similarly, the Z terms can be chosen to be the same up to stabilizers since $\text{enc}(|0\rangle\langle 0|^{\otimes k})$ is stabilized by all Z logical operators. Denote this Z error $P_{Z,s}$. We see that

$$\Pi_s \mathcal{E}(|0\rangle\langle 0|^{\otimes n}) \Pi_s \propto P_{X,s} P_{Z,s} \text{enc}(|0\rangle\langle 0|^{\otimes k}) P_{X,s} P_{Z,s}. \quad (6.96)$$

Therefore, the measured state is a linear combination of $\mathcal{B}_{1/5}$ -avoiding X errors and arbitrarily supported Z errors acting on the ideal state $\text{enc}(|0\rangle\langle 0|^{\otimes k})$.

$$\mathcal{M} \circ \mathcal{E}(|0\rangle\langle 0|^{\otimes n}) = \sum_{\text{syndrome } s} |s\rangle\langle s| \otimes \left(P_{X,s} P_{Z,s} \text{enc}(|0\rangle\langle 0|^{\otimes k}) P_{X,s} P_{Z,s} \right) \quad (6.97)$$

From Remark 6.24, we know that the error correction gadget is friendly in both bases. Since X errors have bounded support, the logical error resulting from the EC gadget is a Z error, which has no impact on the state $\text{enc}(|0\rangle\langle 0|^{\otimes k})$. We conclude that our initialization gadget has the correct output state and output code type. \square

7 qLDPC Threshold Theorems

We first begin by proving a threshold theorem for the surface code roughly following the outline in [Den+02]. The scheme we construct utilizes transversal gates and magic state distillation, and does not limit the connectivity. To prove a threshold under connectivity constraints (such as a 2D lattice), one could utilize physical SWAP gates to route far-apart surface code patches near to each other (such as in [PKP23]) for transversal gates, or implement logical gates with lattice surgery (see Remark 6.26). Both of these approaches can be captured by our formalism, so it would be useful to add them.

7.1 Surface codes

We now begin a threshold proof using surface codes. Here, we will introduce surface codes as a computational code as in Section 6.

Definition 7.1 (Surface code). For $d \in \mathbb{N}$, we use $\text{SC}(d)$ to refer to a surface code of distance d encoding one logical qubit into $O(d^2)$ physical qubits. Concretely, we use the planar surface code ([BK98; FM01]) as defined in [Den+02] with parameters $\llbracket d^2 + (d-1)^2, 1, d \rrbracket$. For convenience, $\text{SC}(1)$ refers to the trivial quantum code on one qubit with identity encoding map. We follow Definition 6.8 and let $\text{SC}_c^{(d)} = (U, \mathcal{B}_c)$ denote the code types of a surface code of distance d .

Proposition 7.2. A classical linear code defined by a check matrix $H \in \mathbb{F}^{r \times n}$ with column weight at most 2 is efficiently minimum-weight decodable by the min-weight perfect matching decoder [Den+02].

Proof sketch. The proof is the standard construction of a matching decoder for the surface code. We give a brief sketch here. We introduce a graph $G_{\text{decoding}} = ([r] \cup \{b\}, E)$ with $r + 1$ vertices: One vertex for every row of H and then an additional “boundary vertex” b . For every column of H introduce an edge in G_{decoding} : If the column is weight two, connect the corresponding vertices (r_1, r_2) . If the column is weight one, connect the vertex to the boundary (r_1, b) .

Given a syndrome $s \in \text{im } H \subseteq \mathbb{F}_2^r \simeq \sigma \subseteq [r]$, add b to the set if σ is odd. Then, we compute a minimum-weight perfect matching $M \subseteq \sigma \times \sigma$ on the weighted fully connected graph $G_{\text{matching}} = (\sigma, \sigma \times \sigma, w)$ where $w(v_1, v_2)$ is the weight of the shortest path connecting v_1, v_2 in G_{decoding} . The correction is the \mathbb{F}_2 -sum of the shortest paths corresponding to each element of the matching M . \square

Corollary 7.3 (Surface code error correction gadget [Den+02; Kit03; BK98; Kit97]). There is an error correction gadget $\text{EC}^{\text{SC}(d)}$ (implementing identity) which has input code type $\text{SC}_{1/2}^{(d)}$ and output code type $\text{SC}_{1/5}^{(d)}$. Its family of bad fault paths has weight enumerator bounded as in Eq. (6.58).

Proof. Consider the spacetime code studied in Section 6.2. Its parity check matrix, illustrated in Fig. 6, has column weight bounded by 2 due to the structure of stabilizers in the surface code. Therefore, minimum weight decoding of the spacetime code is equivalent to finding a min-weight perfect matching on a graph, which can be efficiently solved. The rest of this lemma follows from Theorem 6.23. \square

Lemma 7.4 (Transversal Clifford gates). There exists a constant $\epsilon_{*, \text{TRANSVERSAL}} \in (0, 1)$ and friendly gate gadgets for the set of operations $\text{TRANSVERSAL} = \{\text{I}, \text{X}, \text{Z}, \text{CNOT}, \text{H}, \text{S}, \text{M}_\text{X}, \text{M}_\text{Z}\}$ (and their classically controlled analogs, if unitary) such that

- The gadgets for $\text{I}, \text{X}, \text{Z}, \text{CNOT}, \text{H}, \text{S}$ and their classically controlled analogs has input and output code types $\text{SC}_{1/5}^{(d)}$. The gadgets for $\text{M}_\text{X}, \text{M}_\text{Z}$ has input code type $\text{SC}_{1/5}^{(d)}$ and trivial output code type.
- Each gadget has width $O(d^2)$ and (identical) depth $O(d)$.
- For each operation, $\mathbf{g} \in \text{TRANSVERSAL}$, the family of bad fault paths of the corresponding gadget $\mathcal{F}_\mathbf{g}$ has weight enumerator bounded on $x \in [0, \epsilon_{*, \text{TRANSVERSAL}}]$ as

$$\mathcal{W}(\mathcal{F}_\mathbf{g}; x) \leq e^{-\beta d} \text{poly}(d) \quad (7.1)$$

for some constant $\beta > 0$.

Construction. The logical operations $\text{I}, \text{X}, \text{Z}$ and CNOT can all be implemented transversally on any CSS code. A depth-2 implementation of H applying $H^{\otimes n}$ and one layer of SWAP is given

in [Mou16; BB24], with the idea originating from [Den+02]. A depth-1 implementation of S applying a combination of S and CZ is given in [Mou16; BB24]. The gate gadgets for these unitary gates are their low-depth implementation preceded by the EC gadget $EC^{SC(d)}$.¹⁹ Gadgets for classically controlled analogs are constructed similarly.

For measurement M_X , we first apply the EC gadget $EC^{SC(d)}$, then transversally measure all physical qubits of $SC(d)$ in X basis to obtain a classical bitstring $b_X \in \mathbb{F}_2^n$. Let H_X denote the X stabilizer check matrix of $SC(d)$, compute $s_X = H_X b_X$, and apply a minimum weight decoder D_X to compute a correction $c_X \in \mathbb{F}_2^n$ where $H_X c_X = s_X$. Let L_X be a X logical operator of $SC(d)$, and let $\ell_X \in \mathbb{F}_2^n$ be the indicator vector of its support. Output $\ell_X \cdot (b_X + c_X)$. The gadget for M_Z can be constructed analogously.²⁰ ◀

Proof. The unitary gate gadgets for $\{I, X, Z, CNOT, H, S\}$ have bad fault paths and weight enumerator bounds given by Lemma 6.28.

For M_Z (and similarly M_X), denote the proposed gadget $MEAS_Z$. We first argue that in the absence of input error and fault, the perfect gadget $MEAS_Z$ will correctly measure the logical qubit in Z basis destructively. To see that, since surface code is a CSS code, its logical state has a standard basis.

$$\forall v \in \ker(H_Z), |v + H_X\rangle \propto \sum_{r \in \text{rowsp}(H_X)} |v + r\rangle. \quad (7.2)$$

Transversal measurement in the Z basis will therefore measure to $v + r$ for some $v \in \ker(H_Z)$ and $r \in \text{rowsp}(H_X)$. We see that $H_Z(v + r) = 0$, which means the decoder acts trivially. For a Z logical operator with indicator vector $\ell_Z \in \ker(H_X)$, we see that $\ell_Z \cdot (v + r) = \ell_Z \cdot v$. This is the correct logical measurement result due to the commutation rules of the logical operators of CSS codes.

In the presence of input error \mathcal{E}_{in} and Pauli fault \mathbf{f} , the fault \mathbf{f} is separable as \mathbf{f}_{EC} supported on the locations of the EC gadget and \mathbf{f}_Z supported on the locations of the transversal measurement. Let us consider the output state of the EC gadget,

$$\mathcal{E} \circ \text{enc}(\rho') = \text{map}[EC^{SC(d)}[\mathbf{f}_{\text{EC}}]](\mathcal{E}_{\text{in}} \circ \text{enc}(\rho)). \quad (7.3)$$

Assuming \mathcal{E}_{in} and \mathbf{f} is well-behaving as in Theorem 6.23, $\rho' = \rho$ and \mathcal{E} is $SC(d)_{1/5}$ -avoiding. Now consider the faulty transversal measurement circuit, which can be decomposed as

$$\text{map}[M_Z^{\otimes n}[\mathbf{f}_Z]] = \mathbf{f}_{Z,2} \circ M_Z^{\otimes n} \circ \mathbf{f}_{Z,1} \quad (7.4)$$

for Pauli superoperators $\mathbf{f}_{Z,1}, \mathbf{f}_{Z,2}$. Note that the faults must respect the input and output types of the affected gates. Since the measurement gate output classical bits, $\mathbf{f}_{Z,2}$ must be a diagonal Pauli fault which is a tensor product of X operators acting on a collection of bits F_2 . To analyze \mathcal{E} and $\mathbf{f}_{Z,1}$, because we measured all physical qubits individually in the Z basis, by the decoherence of errors lemma (Lemma 4.14) $\mathbf{f}_{Z,1} \circ \mathcal{E}$ decoheres into single-qubit Pauli X errors acting on a collection

¹⁹Technically, if the unitary implementation is depth-1, we add a layer of identity gates afterwards to make them depth-2 in order to satisfy the identical depth condition in lemma statement.

²⁰We pad these gadgets by identity gates (on classical registers) so that they have the same depth as the unitary gate gadgets.

of qubits $F_1 \cup E \subseteq [n]$, where F_1 is a subset of the support of $\mathbf{f}_{Z,1}$ and E is a subset of the support of \mathcal{E} .²¹

Since the support of \mathcal{E} is $\text{SC}(d)_{1/5}$ -avoiding, E is $\text{SC}(d)_{1/5}$ -avoiding. Let $\mathcal{F}_{M_Z} = \text{SC}(d)_{1/5}$ denote the bad fault paths on the transversal measurement. If \mathbf{f}_Z is \mathcal{F}_{M_Z} -avoiding, we see that $F = F_1 \cup F_2$ is $\text{SC}(d)_{1/5}$ -avoiding. Let $1_E, 1_F$ denote the indicator vectors for sets E, F , and let $e = 1_E + 1_F$. We see that the support of e is $\text{SC}(d)_{2/5}$ -avoiding, and the measurement outcome is $v + r + e$. Since we used a minimum weight decoder D_Z , the correction c_Z satisfies $c_Z + e \in \text{rowsp}(H_X)$ (Lemma 6.17). Therefore, $v + r + e + c_Z \in v + \text{rowsp}(H_X)$ and the output classical bit is again the correct logical measurement outcome. We conclude that the proposed gadget fault-tolerantly implements M_Z , with bad fault paths and weight enumerator bound given by Corollary 6.27. \square

We remark that the transversal measurement gadgets can be straightforwardly generalized to any CSS code, measuring out all logical qubits in the X or Z basis.

7.1.1 State injection

To employ magic state distillation [BK05], we need a source of noisy logical magic states. Here, we will employ a brute-force approach to construct a state-injection gadget for the surface code. We utilize the following unitary circuit for growing a surface code, which was presented in Figure 19 of [Den+02].

Fact 7.5 (Surface code growth). For $\ell \in \mathbb{N}$, there exists a depth $O(\ell)$ unitary circuit $U_{(d+\ell) \leftarrow d}$ that acts on a $\text{SC}(d)$ codeblock and $O(\ell d + \ell^2)$ qubits initialized to $|0\rangle$ and returns a $\text{SC}(d + \ell)$ codeblock. That is, as superoperators, we have that

$$U_{(d+\ell) \leftarrow d} \circ \text{enc}_{\text{SC}(d)} = \text{enc}_{\text{SC}(d+\ell)} . \quad (7.5)$$

We can use the constant-depth circuit in Fact 7.5 alternately with the error correction gadget to prepare an arbitrarily large surface code block. The main idea is that the error correction gadget has exponential error suppression in the block length, so there are relatively few ways that the earlier parts of this procedure can fail.

Lemma 7.6 (Surface code state injection gadget [Den+02]). There exists a constant $\epsilon_{*,\text{INJECT}} \in (0, 1)$ such that, given a constant-sized circuit C producing a state $|\psi\rangle$, there exists a gadget $\text{INJECT}_{|\psi\rangle}^{(d)}$ with output code type $\text{SC}_{1/5}^{(d)}$ that prepares $\text{enc}_{\text{SC}(d)}(|\psi\rangle\langle\psi|)$ and bad fault paths $\mathcal{F}_{\text{INJECT}}^d$ with weight enumerator bounded on $x \in [0, \epsilon_{*,\text{INJECT}}]$ as

$$\mathcal{W}(\mathcal{F}_{\text{INJECT}}^d; x) \leq c \cdot x \quad (7.6)$$

for some absolute constant $c > 0$. The depth is $O(d^2)$ and the width is $O(d^2)$.

²¹Technically, we apply Lemma 4.14 with the trivial quantum code whose stabilizer group consists of all n single-qubit X operators.

Construction. Let $\text{GROW}_{d' \leftarrow d}$ be the circuit that introduces the necessary ancilla qubits and applies the growth unitary $U_{d' \leftarrow d}$ from Fact 7.5. We define the $\text{INJECT}^{(d)}$ gadget inductively as

$$\text{INJECT}^{(d)} = \text{GROW}_{d \leftarrow (d-1)} \circ \text{EC}^{\text{SC}(d-1)} \circ \text{INJECT}^{(d-1)} \quad (7.7)$$

$$= \text{GROW}_{d \leftarrow (d-1)} \circ \text{EC}^{\text{SC}(d-1)} \circ \dots \circ \text{GROW}_{3 \leftarrow 2} \circ \text{EC}^{\text{SC}(2)} \circ \text{GROW}_{2 \leftarrow 1} . \quad (7.8)$$

This is a gadget for identity. The gadget that injects a given state $|\psi\rangle$ is simply the circuit C followed by injection.

$$\text{INJECT}_{|\psi\rangle}^{(d)} = \text{INJECT}^{(d)} \circ C. \quad (7.9)$$

◀

Proof. For a code distance s , we analyze the gadget $\text{GROW}_{s+1 \leftarrow s} \circ \text{EC}^{\text{SC}(s)}$. Since the growth circuit from Fact 7.5 has constant depth, by Lemma 6.28, this is a friendly gadget that implements identity with input code type $\text{SC}_{1/5}^{(s)}$ and output code type $\text{SC}_{1/5}^{(s+1)}$, with bad fault paths \mathcal{F}_s .

We now define the bad fault paths $\mathcal{F}_{\text{INJECT}}^d$. Let \bar{d} be a absolute constant to be specified later. Let V be the set of all locations in the circuit $\text{INJECT}^{(\bar{d})}$. Define \mathcal{F}_1 to be all singletons of V .

$$\mathcal{F}_1 = \{\{v\} : v \in V\}. \quad (7.10)$$

Since \bar{d} is a constant, the number of locations in V is also a constant. For $s \geq \bar{d}$, recall that \mathcal{F}_s is defined by Lemma 6.28. We define $\mathcal{F}_{\text{INJECT}}^d$ to be the sum of all of these fault paths where \mathcal{F}_s should be understood as subsets of locations of the s -th step in the growth sequence and \mathcal{F}_1 as subsets of locations of the first \bar{d} -steps in the growth sequence.

$$\mathcal{F}_{\text{INJECT}}^d = \mathcal{F}_1 \boxplus_{s=\bar{d}}^d \mathcal{F}_s. \quad (7.11)$$

Intuitively, this definition enforces that the injection gadget suffers no faults until the surface code size reaches a large enough constant, after which the exponential error suppression of the EC gadgets will dominate and ensure that the weight enumerator is sufficiently bounded.

We proceed to the analysis. From Lemma 6.28, there exists a polynomial poly and absolute constants c_1, c_2 such that the weight enumerator of \mathcal{F}_s can be bounded as follows.

$$\mathcal{W}(\mathcal{F}_s; x) \leq \text{poly}(s)(c_1 x^{1/c_2})^s. \quad (7.12)$$

We want to restrict x and s such that the following inequalities hold

$$\text{poly}(s)(c_1 x^{1/c_2})^s \leq 2^{-s} x^{\frac{s}{2c_2}} \leq 2^{-s} x. \quad (7.13)$$

Note that the first inequality is equivalent to

$$\text{poly}(s)(2c_1 x^{\frac{1}{2c_2}})^s \leq 1. \quad (7.14)$$

Let $\epsilon_{*,\text{INJECT}} = (\frac{1}{4c_1})^{2c_2}$. For all $x \in [0, \epsilon_{*,\text{INJECT}})$, it holds that $2c_1 x^{\frac{1}{2c_2}} < 1/2$. Now let \bar{d} be an integer which is at least $2c_2$, such that for all $s \geq \bar{d}$, it holds that $\text{poly}(s)2^{-s} \leq 1$. Then Eq. (7.14)

holds, which implies Eq. (7.13). The weight enumerator can then be bounded using the preceding equations.

$$\mathcal{W}(\mathcal{F}_{\text{INJECT}}^d; x) \leq \mathcal{W}(\mathcal{F}_1; x) + \sum_{s=\bar{d}}^d \mathcal{W}(\mathcal{F}_s; x) \quad (7.15)$$

$$\leq |V| \cdot x + \sum_{s=\bar{d}}^d \text{poly}(s) (c_1 x^{1/c_2})^s \quad (7.16)$$

$$\leq |V| \cdot x + \sum_{s=\bar{d}}^d 2^{-s} x \quad (7.17)$$

$$\leq (|V| + 1) \cdot x. \quad (7.18)$$

The depth of the gadget is $O(d^2)$ since every error correction gadget has depth bounded by $O(d)$ and every growth circuit has constant depth. This completes the analysis of $\text{INJECT}^{(d)}$.

For a given state $|\psi\rangle$ produced by a constant-sized circuit C , let V_C be the set of locations of C , and define \mathcal{F}_C to be all singletons of V . The collection of bad fault paths for $\text{INJECT}_{|\psi\rangle}^{(d)}$ is

$$\mathcal{F}_{\text{INJECT},|\psi\rangle}^d = \mathcal{F}_{\text{INJECT}}^d \boxplus \mathcal{F}_C, \quad (7.19)$$

and its weight enumerator is bounded by $(|V| + |V_C| + 1) \cdot x$. \square

In practice, it is better to grow the lattice all at once [Li15], but the analysis of the lattice growth leads to some involved combinatorics. Interested readers should refer to [Łod+15; SBB23].

7.1.2 Magic state distillation

We will use the magic states $|\mathsf{T}\rangle \equiv \mathsf{T} |+\rangle$.

Fact 7.7. There exists a quantum CSS code $\mathcal{Q}^{(\text{distill})}$ with parameters $\llbracket 15, 1, 3 \rrbracket$ such that the encoding map is implementable by a Clifford circuit and transversal T applies logical T . That is,

$$\text{dec}_{\mathcal{Q}^{(\text{distill})}} \mathsf{T}^{\otimes 15} \text{enc}_{\mathcal{Q}^{(\text{distill})}} = \mathsf{T}. \quad (7.20)$$

Fact 7.8 (T teleportation circuit). It is a standard result that $|\mathsf{T}\rangle$ can be consumed by a Clifford circuit containing a classically-controlled SX operation in order to implement T . For reference, see Section 10.6.2, Figure 10.25 of [NC12].

Proposition 7.9 (Single-level state distillation). Let $\mathcal{S}_{(2,15)}$ be all size-2 subsets of the set $[15]$. There exists a constant-sized Clifford circuit with measurements and classical processing acting on a 15-qubit state such that when the input is $\mathcal{S}_{(2,15)}$ -deviated from $|\mathsf{T}\rangle^{\otimes 15}$, the output is

proportional to (trivially deviated from) $|T\rangle\langle T|$.

Construction. The circuit consists of four steps.

1. PREP: Prepare the logical $|+\rangle$ state of the code $\mathcal{Q}^{(\text{distill})}$ from Fact 7.7.
2. TP: Use the teleportation circuit from Fact 7.8 to apply T to the logical $|+\rangle$ state.
3. $\text{EC}_{\text{distill}}$: Perform syndrome extraction and apply a minimum weight correction.
4. Apply $\text{dec}_{\text{distill}}$ and output the remaining qubit.

◀

Proof. We assumed that the circuit is executed without fault and the input is $\mathcal{E}_{\text{in}}(|T\rangle\langle T|^{\otimes 15})$, where \mathcal{E}_{in} is supported on a single-qubit. Consider the state

$$\sigma = \text{TP} \circ \text{PREP} \circ \mathcal{E}_{\text{in}}(|T\rangle\langle T|^{\otimes 15}) \quad (7.21)$$

The error \mathcal{E}_{in} is propagated to an error \mathcal{E} acting on one of the qubits of $\mathcal{Q}^{(\text{distill})}$, while the remaining 14 qubits have the correct T gates applied.

$$\sigma = (\mathcal{E} \otimes T^{\otimes 14}) \circ \text{enc}_{\text{distill}}(|+\rangle\langle +|) \quad (7.22)$$

$$= (\mathcal{E} \circ T^\dagger) \circ T^{\otimes 15} \circ \text{enc}_{\text{distill}}(|+\rangle\langle +|) \quad (7.23)$$

$$= (\mathcal{E} \circ T^\dagger) \circ \text{enc}_{\text{distill}}(|T\rangle\langle T|) \quad (7.24)$$

Upon application of $\text{EC}_{\text{distill}}$, by the decoherence of errors lemma (Lemma 4.14), the error superoperator $\mathcal{E} \circ T^\dagger$ decomposes into a linear combination of (diagonal) Pauli errors supported on the affected qubit, which is then corrected by the minimum weight correction.

$$\text{EC}_{\text{distill}}(\sigma) \propto \text{enc}_{\text{distill}}(|T\rangle\langle T|) \quad (7.25)$$

Therefore applying $\text{dec}_{\text{distill}}$ gives us the desired $|T\rangle\langle T|$ state. \square

Lemma 7.10 (Logical level state distillation gadget [BK05]). Let $\text{PREPARE}_{|T\rangle}$ be the superoperator that outputs a $|T\rangle$ state. There exists a constant $\epsilon_{*,\text{PREPARE}} \in (0, 1)$ such that for $d \in \mathbb{N}$, there exists a family of gadgets $\overline{\text{PREPARE}}_{|T\rangle}^{(d)}$ for $\text{PREPARE}_{|T\rangle}$ that has output code type $\text{SC}_{1/5}^{(d)}$ and a family of bad fault paths $\mathcal{F}_d^{\overline{\text{PREPARE}}_{|T\rangle}}$ satisfying the upper bound

$$\mathcal{W}\left(\mathcal{F}_d^{\overline{\text{PREPARE}}_{|T\rangle}}; x\right) = O(\text{poly}(d) \cdot e^{-\beta d}) \quad (7.26)$$

on $x \in [0, \epsilon_{*,\text{PREPARE}}]$.

$\overline{\text{PREPARE}}_{|T\rangle}^{(d)}$ has width $O(d^{2+\log_2(15)})$ and depth $O(d^2)$.

Construction. We fix a constant $\ell = \lceil \log_2 \beta d \rceil$ corresponding to the number of rounds of distillation. Run the state injection gadget $\text{INJECT}_{|T\rangle}^{(d)}$ (Lemma 7.6) for the circuit $T|+\rangle$ in parallel $N_0 = 15^\ell$ times followed by the error correction gadget²² $\text{EC}^{\text{SC}(d)}$ to obtain the state ρ_0 which is a noisy copy of $\text{enc}_{\text{SC}(d)}(|T\rangle\langle T|)^{\otimes N_0}$. Let DISTILL be the distillation circuit for $|T\rangle$ (Proposition 7.9) where every gate is replaced by $\text{SC}(d)$ transversal gate gadgets (Lemma 7.4). Perform ℓ rounds of the following procedure to obtain the sequence of states ρ_1, \dots, ρ_ℓ : In round $i \in [\ell]$, execute DISTILL $15^{\ell-i}$ times in parallel on ρ_{i-1} to obtain ρ_i . It follows that ρ_i is $N_i = 15^{\ell-i-1}$ noisy copies of $\text{enc}_{\text{SC}(d)}(|T\rangle\langle T|)$. Output ρ_ℓ . \blacktriangleleft

Proof. $\text{INJECT}_{|T\rangle}^{(d)}$ has depth $O(d^2)$ while the transversal gate gadgets have depth $O(d)$. There are $O(\ell)$ rounds of distillation, each one having $O(d)$ depth, so the overall depth is $O(d^2 + \ell \cdot d) = O(d^2)$. Overall there are at most $O(15^\ell)$ gadgets at any time, each of width $O(d^2)$, so the total width is $O(d^{2+\log_2(15)})$.

Weight enumerators Let Ω_{INJECT} be the set of $\text{INJECT}_{|T\rangle}^{(d)}$ gadgets and $\Omega_{\text{TRANSVERSAL}}$ the set of all error correction and transversal gate gadgets in the distillation steps (that is, including the error correction gadget after the distillation gadget but excluding any error correction gadgets within the injection gadgets). Let Ω_{DISTILL} be all instances of the distillation circuit DISTILL . The gadget will have two families of bad fault paths corresponding to failure of too many injection gadgets \mathcal{F}_1 and failure of a transversal gate of the distillation circuit \mathcal{F}_2 , respectively.

\mathcal{F}_2 will be the sum of the families of bad faults associated with each transversal gate gadget. In the following expression, we use \mathcal{F}_g to mean the family of bad fault paths associated with the transversal gate gadgets $g \in \Omega_{\text{TRANSVERSAL}}$.

$$\mathcal{F}_2 = \bigsqcup_{g \in \Omega_{\text{TRANSVERSAL}}} \mathcal{F}_g \quad (7.27)$$

There are $O(\ell \cdot 15^\ell)$ transversal gate gadgets, so utilizing the bound in Lemma 7.4, on values of $x \in [0, \epsilon_{*, \text{TRANSVERSAL}}]$, the weight enumerator of \mathcal{F}_2 obeys the bound

$$\mathcal{W}(\mathcal{F}_2; x) \leq O(\ell \cdot 15^\ell \text{poly}(d) e^{-\beta d}) \quad (7.28)$$

Let $\mathcal{S}_{(2,15)}$ be all size-2 subsets of $[15]$. For $i \in \{0, \dots, \ell\}$, the code blocks of ρ_i are in bijection with the $(\ell - i)$ -fold Cartesian product $[15]^{\times(\ell-i)}$. For conciseness, we define $[15]^{\times 0} \equiv \{\emptyset\}$ (singleton set) and $() \in \{\emptyset\}$ (empty tuple). The distillation circuits Ω_{DISTILL} are in bijection with $\{\emptyset\} \cup [15] \cup [15]^{\times 2} \cup \dots \cup [15]^{\times \ell-1}$ such that the distillation circuit at coordinates $(i_1, \dots, i_{\ell-i}) \in [15]^{\times(\ell-i)}$ consumes the code blocks with coordinates $(i_1, \dots, i_{\ell-i}) \times [15]$. Let $\mathcal{S} \subseteq P([15]^{\times \ell})$ be the set (recall the definition of \bullet from Definition 2.7)

$$\mathcal{S} = \underbrace{\mathcal{S}_{(2,15)} \bullet \dots \bullet \mathcal{S}_{(2,15)}}_{\ell \text{ times}}. \quad (7.29)$$

That is, for an \mathcal{S} -avoiding set $X \subseteq [15]^{\times \ell}$ and every $i \in \{0, \dots, \ell\}$, X is $(\mathcal{S}_{(2,15)})^{\bullet i}$ -avoiding on every i -rectangle $(i_1, \dots, i_{\ell-i}) \times [15]^{\times i} \subseteq [15]^{\times \ell}$ except for an $(\mathcal{S}_{(2,15)})^{\bullet(\ell-i)}$ -avoiding subset $I \subseteq [15]^{\times(\ell-i)}$ of i -rectangles.

²²The EC gadget is not necessary, but we perform it for conceptual clarity.

Let $\mathcal{F}_{\text{INJECT}}$ be the bad fault paths of the injection gadget $\text{INJECT}_{|T}^{(d)}$. We will define $\mathcal{F}_1 \subseteq \Omega_{\text{INJECT}}$ as the fault paths that lead to the failure of a set of injection gadgets that is not \mathcal{S} -avoiding.

$$\mathcal{F}_1 = \mathcal{S} \bullet \mathcal{F}_{\text{INJECT}} \quad (7.30)$$

Using the composition evaluation formula (Proposition 2.8) and the simple weight enumerator $\mathcal{W}(\mathcal{S}_{(2,15)}; x) = 105x^2$, a short calculation²³ shows that the weight enumerator of \mathcal{S} is

$$\mathcal{W}(\mathcal{S}; x) = (105)^{2^\ell - 1} x^{2^\ell} . \quad (7.31)$$

Thus, on $x \in [0, \epsilon_{*,\text{INJECT}}]$, the weight enumerator of \mathcal{F}_1 can be bounded as

$$\mathcal{W}(\mathcal{F}_1; x) \leq \mathcal{W}(\mathcal{S}; \mathcal{W}(\mathcal{F}_{\text{INJECT}}; x)) \leq (c \cdot x)^{2^\ell} \quad (7.32)$$

for some constant $c > 0$ that depends on the absolute constant in the weight enumerator upper bound of $\mathcal{F}_{\text{INJECT}}$ (Lemma 7.6). Thus, there exists a constant

$$\epsilon_{*,\text{PREPARE}} = \min \left(\epsilon_{*,\text{INJECT}}, \frac{1}{e \cdot c}, \epsilon_{*,\text{TRANSVERSAL}} \right) \quad (7.33)$$

such that on $x \in [0, \epsilon_{*,\text{PREPARE}}]$ the weight enumerator of \mathcal{F}_1 is bounded as

$$\mathcal{W}(\mathcal{F}_1; x) \leq e^{-2^\ell} . \quad (7.34)$$

The family of bad fault paths for the gadget is the sum

$$\mathcal{F}_{\text{PREPARE}_{|T}}^d = \mathcal{F}_1 \boxplus \mathcal{F}_2 . \quad (7.35)$$

Using the choice of $\ell = \lceil \log_2 \beta d \rceil$, for $x \in [0, \epsilon_{*,\text{PREPARE}}]$ the weight enumerator is bounded as

$$\mathcal{W}(\mathcal{F}_{\text{PREPARE}_{|T}}^d; x) = O \left(\text{poly}(d, \ell) 15^\ell e^{-\beta d} + e^{-2^\ell} \right) \quad (7.36)$$

$$= O(\text{poly}(d) e^{-\beta d}) . \quad (7.37)$$

Correctness It remains to show that the output satisfies the code type $\text{SC}_{1/5}^{(d)}$ when the fault path is $\mathcal{F}_{\text{PREPARE}_{|T}}^d$ -avoiding. Let \mathbf{f} be a $\mathcal{F}_{\text{PREPARE}_{|T}}^d$ -avoiding Pauli fault. Recall that ρ_0 is the state after state injection and error correction $(\text{EC}^{\text{SC}(d)} \circ \text{INJECT}_{|T}^{(d)})^{\otimes N_0}[\mathbf{f}]$.

From the construction of $\mathcal{F}_{\text{PREPARE}_{|T}}^d$, \mathbf{f} is \mathcal{F}_1 -avoiding and \mathcal{F}_2 -avoiding. It follows that there is an \mathcal{S} -avoiding set $B \subseteq \Omega_{\text{INJECT}} \equiv [15]^\ell$ of $\text{INJECT}_{|T}^{(d)}$ gadgets that have an $\mathcal{F}_{\text{INJECT}}$ -avoiding fault path. The state after the injection gadgets is therefore $\mathcal{S} \bullet \text{SC}_{1/5}^{(d)}$ -deviated from $\text{enc}_{\text{SC}(d)}(|T\rangle\langle T|^{\otimes N_0})$. Using

²³For a function $f(x) = c \cdot x^2$, the ℓ -fold composition $f^{(\ell)}(x)$ obeys the recursion $f^{(\ell)}(x) = (\sqrt{c} \cdot f^{(\ell-1)}(x))^2$ and has the closed form $f^{(\ell)}(x) = c^{2^\ell - 1} x^{2^\ell}$.

the friendliness property²⁴ of the error correction gadgets, it follows that ρ_0 is $\text{SC}_{1/5}^{(d)}$ -deviated on all blocks from a state $\text{enc}_{\text{SC}(d)}(\bar{\rho}_0)$ where $\bar{\rho}_0$ is \mathcal{S} -deviated from $|\mathbf{T}\rangle\langle\mathbf{T}|^{\otimes N_0}$.

$$\rho_0 \lesssim_{\text{SC}_{1/5}^{(d)}} \text{enc}_{\text{SC}(d)}(\bar{\rho}_0) \quad (7.38)$$

$$\bar{\rho}_0 \lesssim_{\mathcal{S}} |\mathbf{T}\rangle\langle\mathbf{T}|^{\otimes N_0} \quad (7.39)$$

Using the property that \mathcal{F}_2 is the sum of the bad fault paths for each transversal gate gadget of Ω_{DISTILL} and applying the gate the gadget definition (Definition 4.8), for each distillation level $i \in [\ell]$, ρ_i is $\text{SC}_{1/5}^{(d)}$ -deviated on each block from a state $\text{enc}_{\text{SC}(d)}(\bar{\rho}_i)$ where $\bar{\rho}_i$ is the result of applying i layers of the distillation protocol Proposition 7.9 recursively to $\bar{\rho}_0$.

We will now apply the distillation property of Proposition 7.9 inductively to establish that each $\bar{\rho}_i$ is deviated from a tensor product of $|\mathbf{T}\rangle\langle\mathbf{T}|$. Consider layer- i of executions of the distillation protocol,

$$\Omega_{\text{DISTILL}}^{(i)} \equiv [15]^{\times(\ell-i)} \subseteq \Omega_{\text{DISTILL}}. \quad (7.40)$$

$\Omega_{\text{DISTILL}}^{(i-1)}$ indexes the input qubits which are grouped together in sets labeled by indices in $\Omega_{\text{DISTILL}}^{(i)}$ (see construction). Suppose that there exists a $(\mathcal{S}_{(2,15)})^{\bullet(\ell-i)}$ -avoiding subset $I \subseteq \Omega_{\text{DISTILL}}^{(i)}$ such that for each collection of inputs $A = (j_1, \dots, j_{\ell-i}) \times [15] \in \Omega_{\text{DISTILL}}^{(i-1)}$, either $\bar{\rho}_{i-1}$ is $\mathcal{S}_2^{(15)}$ -deviated from $|\mathbf{T}\rangle\langle\mathbf{T}|^{\otimes 15}$ on A ,²⁵ or $(j_1, \dots, j_{\ell-i}) \in I$. Then, applying Proposition 7.9 to every distillation protocol execution in

$$I^c \equiv \Omega_{\text{DISTILL}}^{(i)} \setminus I \quad (7.41)$$

implies that $\bar{\rho}_i$ is $|\mathbf{T}\rangle\langle\mathbf{T}|^{\otimes N_i}$ except for a superoperator supported on I . In other words, $\bar{\rho}_i$ is $(\mathcal{S}_{(2,15)})^{\bullet(\ell-i)}$ -deviated from $|\mathbf{T}\rangle\langle\mathbf{T}|^{\otimes N_i}$. Since $\mathcal{S} \equiv (\mathcal{S}_{(2,15)})^{\bullet(\ell)}$, the base case is satisfied and $(\mathcal{S}_{(2,15)})^{\bullet 0} \equiv \{\{1\}\}$ so $\bar{\rho}_\ell$ is proportional to $|\mathbf{T}\rangle\langle\mathbf{T}|$. The output state ρ_ℓ was previously shown to be $\text{SC}_{1/5}^{(d)}$ -deviated from $\text{enc}_{\text{SC}(d)}(\bar{\rho}_\ell)$, so this completes the proof. \square

7.1.3 Assembly

Theorem 7.11 (Threshold theorem for surface code quantum computation). There exists a constant $\epsilon_* \in (0, 1)$ such that, for any Clifford+T circuit C of width W and depth D and any $\epsilon \in (0, 1)$, there exists a circuit \bar{C} that is a fault-tolerant gadget for C with bad fault paths \mathcal{F} . When C is a circuit with only classical inputs and classical outputs, \bar{C} has trivial input and output types. The type of any quantum inputs and outputs is $\text{SC}_{1/5}^{(d)}$ where $d = O(\log(V)\text{polyloglog}(V))$.

²⁴Informally, the blocks that are in the \mathcal{S} -avoiding set are arbitrarily damaged and are corrected to some arbitrary codestate.

²⁵More accurately, $\bar{\rho}_{i-1}$ is $\mathcal{S}_2^{(15)}$ -deviated from a state $|\mathbf{T}\rangle\langle\mathbf{T}|^{\otimes 15} \otimes \sigma$ where σ is an arbitrary state of the complement A^c and the deviation superoperator may act arbitrarily on σ . Here, σ should be considered as an input to an environment circuit.

Let $V = \frac{WD}{\epsilon}$. Then, \overline{C} has width \overline{W} and depth \overline{D} satisfying the bounds

$$\overline{W} = O(W \log^{5.91}(V) \text{polyloglog}(V)) \quad (7.42)$$

$$\overline{D} = O(D \log^2(V) \text{polyloglog}(V)) . \quad (7.43)$$

On $x \in [0, \epsilon_*]$, the weight enumerator of \mathcal{F} satisfies the bound

$$\mathcal{W}(\mathcal{F}; x) \leq \epsilon . \quad (7.44)$$

Construction. We will defer the choice of the surface code size $d \in \mathbb{N}$ to the proof. Let $\epsilon_* = \min(\epsilon_{*, \text{PREPARE}}, \epsilon_{*, \text{TRANSVERSAL}})$.

C is Clifford+T, at the loss of a constant multiplicative factor in the depth, we can assume it contains gates and their classically controlled analogs (except for measurement) from the following set

$$\text{GATESET} = \{\text{T}, \text{I}, \text{X}, \text{YZ}, \text{CNOT}, \text{H}, \text{S}, \text{M}_X, \text{M}_Z, \text{INIT}_Z\} . \quad (7.45)$$

We begin by construction a set of gadgets that implement gates in **GATESET** and have identical depth. We construct a gadget G_T for **T** in the following way: First prepare a magic state with the gadget $\overline{\text{PREPARE}}_{|T\rangle}^{(d)}$ (Lemma 7.10), then consume it to apply a teleported **T** gate using the transversal gate gadgets (Lemma 7.4) implementing the **T** teleportation circuit (Fact 7.8). Let τ be the duration of this gadget.

For all other gates $g \in \text{GATESET} \setminus \{\text{T}\}$, the operation is in **TRANSVERSAL**. Therefore, we can construct the corresponding gadget G_g by padding gadgets from Lemma 7.4 to length τ with an additional error correction gadget.

Construct the family of circuits \overline{C}_d by replacing every gate g of C with the corresponding gadget G_g . ◀

Proof. First, since G_T is the composition of a constant number of gadgets with the state preparation gadget, $\tau = O(d^2)$ (i.e. the logical cycle time is not too long). Consider an operation $g \in \text{GATESET}$ and let \mathcal{F}_g be the weight enumerator for the gadget G_g . Applying the weight enumerator upper bounds (Lemma 7.4, Lemma 7.10), it follows that the previously constructed gadgets are (g, \mathcal{F}_g) -FT gadgets in the sense of Definition 4.8 where for $x \in [0, \epsilon_*]$, the weight enumerator of \mathcal{F}_g is bounded as

$$\mathcal{W}(\mathcal{F}_g; x) \leq \text{poly}(d) e^{-\beta d} . \quad (7.46)$$

Let \bar{V} be the set of gadgets of \overline{C}_d . For each gadget $g \in \bar{V}$ of \overline{C}_d , let \mathcal{F}_g be its bad fault paths. We define the bad fault paths of \overline{C}_d to

$$\mathcal{U}_d = \boxplus_{g \in \bar{V}} \mathcal{F}_g . \quad (7.47)$$

It follows from the construction and the application of Proposition 4.9 inductively on a topological sort of the bundled circuit corresponding to \overline{C}_d that \overline{C}_d is a (C, \mathcal{U}_d) -FT gadget. Using the weight

enumerator upper bounds and Proposition 2.6, it follows that for $x \in [0, \epsilon_*]$,

$$\mathcal{W}(\mathcal{U}_d; x) \leq \bar{V} \text{poly}(d) e^{-\beta d} . \quad (7.48)$$

Since $\bar{V} = O(WD)$, there exists $d = O(\log(V) \text{polyloglog}(V))$ such that $\mathcal{W}(\mathcal{U}_d; x) \leq \epsilon$. \bar{C} is \bar{C}_d with the minimum such choice of d .

By using the depth and width bounds of the corresponding gadgets, depth \bar{D} and width \bar{W} of \bar{C} satisfies

$$\begin{aligned} \bar{D} &= O(D\tau) = O(D \log(V)^2 \text{polyloglog}(V)) \\ \bar{W} &= O(Wd^{2+\log_2(15)}) = O(W \log(V)^{5.91} \text{polyloglog}(V)) . \end{aligned} \quad (7.49) \quad \square$$

Remark 7.12 (Spacetime overhead). It is possible to obtain a depth overhead of $\tilde{O}(\log(V))$ by growing the lattice all at once [Den+02; Li15], but this requires some more complicated combinatorics. It is possible to reduce the width overhead to $\tilde{O}(\log^{2+\gamma}(V))$ where γ is the exponent of the magic state distillation scheme distillation cost by using a different magic state distillation schemes (e.g. with post selection or different codes). For example, it is possible to achieve $\gamma \leq 2.47$ [BK05], $\gamma \leq 1.6$ [BH12], $\gamma \leq 0.68$ [HH18]. For states different from $|T\rangle$ but convertible to $|T\rangle$ with a catalyst state [Bev+20], it is possible to achieve $\gamma = o_{d \rightarrow \infty}(1)$ without substantially harming the time complexity [WHY24; GG25; Ngu25; NP24].

A direct corollary of the above theorem is that we can obtain a surface code fault tolerance scheme against local stochastic noise with a constant threshold. Using Proposition 5.3, we can also obtain a fault tolerance scheme against coherent noise, albeit the threshold is sub-constant.

Corollary 7.13 (Sub-constant threshold against coherent noise). There exists a polylogarithmically decaying function $\delta_* = O(1/\log^{6.91}(WD/\epsilon))$ such that, for any Clifford+T circuit C of width W and depth D and any $\epsilon \in (0, 1)$, there exists a circuit \bar{C} that is a fault-tolerant gadget for C , whose parameters are given in Theorem 7.11. When C is a circuit with classical inputs and classical outputs, \bar{C} has trivial input and output types. The outputs of C and \bar{C} are ϵ -close in trace distance when \bar{C} is subject to coherent noise \mathbf{f} of strength at most δ_* .

Proof. The construction is given in Theorem 7.11, and the proof is almost the same except we use Proposition 5.3 to turn the upper bound on the weight enumerator of the bad fault paths into a bound on the trace distance. Note that the condition in Proposition 5.3, which asks that the bad fault paths of the whole circuit is a sum of bad fault paths of individual gadgets, is satisfied. The number of gadgets m satisfies $m \leq WD$. We want

$$m(1 + \eta)^m \eta \leq \epsilon, \quad (7.50)$$

where $\eta = \max_{i \in m} \mathcal{W}(\mathcal{F}_i; 2(1+x)^{\Gamma-1}x)$, and Γ is defined as

$$\Gamma = \max_{i \in [m]} \max_{F \in \mathcal{F}_i} |V_i|/|F|, \quad (7.51)$$

for V_i being the set of locations in the i -th gadget.

To satisfy Eq. (7.50), it suffices for us to choose $\eta < \frac{\epsilon}{2m} = O(\epsilon/WD)$, as that would imply

$$m(1 + \eta)^m \eta \leq \epsilon e^{\eta m} / 2 \leq \epsilon e^{\epsilon/2} / 2 \leq \epsilon. \quad (7.52)$$

According to Lemma 7.4 and Lemma 7.10, the gadgets in the circuit has bad fault paths \mathcal{F}_i such that for $x \in [0, \epsilon_*]$, the weight enumerator is bounded as

$$\mathcal{W}(\mathcal{F}_i; x) \leq \text{poly}(d) e^{-\beta d} \quad (7.53)$$

for some absolute constant β .²⁶ Let us choose $d = O(WD/\epsilon)$ such that

$$\eta < \frac{\epsilon}{2m} \leq \text{poly}(d) e^{-\beta d}. \quad (7.54)$$

It suffices for us to choose δ_* such that for all $x \leq \delta_*$, we have

$$2(1 + x)^{\Gamma-1} x \leq \epsilon_*. \quad (7.55)$$

From Lemma 7.4 and Lemma 7.10, we know that the volume of each gadget is at most $O(d^{7.91})$, and the bad sets are all of weight at least $\Omega(d)$. This implies that $\Gamma = O(d^{6.91})$. Choose $\delta_* = \frac{\epsilon_* \log \Gamma}{2\Gamma}$, then for all $x \leq \delta_*$, it holds that

$$2(1 + x)^{\Gamma-1} x \leq \epsilon_* e^{\Gamma x} / \Gamma \leq \epsilon_* e^{\epsilon_* \log \Gamma / 2} / \Gamma \leq \epsilon_*. \quad (7.56)$$

The threshold delays polylogarithmically in the circuit volume, namely, $\delta_* = O(\log d / d^{6.91})$ where $d = O(WD/\epsilon)$. \square

7.2 Fault-tolerant quantum output

Lemma 7.14 (Surface code unencoding gadget). There exists a constant $\epsilon_{*, \text{UNENCODE}} \in (0, 1)$ such that we have a gadget $\text{UNENCODE}^{(d)}$ for identity with input code type $\text{SC}_{1/5}^{(d)}$ and trivial output code type and bad fault paths $\mathcal{F}_{\text{UNENCODE}}^d$ with weight enumerator bounded on $x \in [0, \epsilon_{*, \text{UNENCODE}}]$ as

$$\mathcal{W}(\mathcal{F}_{\text{UNENCODE}}^d; x) \leq c \cdot x \quad (7.57)$$

for some absolute constant $c > 0$. The depth is $O(d^2)$ and the width is $O(d^2)$.

Construction. Recall the growth unitary $U_{d' \leftarrow d}$ from Fact 7.5, let $\text{SHRINK}_{d' \rightarrow d}$ be the channel which applies the inverse unitary $U_{d' \leftarrow d}^\dagger$ and traces out the unencoded ancilla qubits. We define the unencoding gadget by alternatively apply SHRINK and EC .

$$\text{UNENCODE}^{(d)} = \text{SHRINK}_{d \rightarrow d-1} \circ \text{EC}^{\text{SC}(d-1)} \circ \text{UNENCODE}^{(d-1)} \quad (7.58)$$

$$= \text{SHRINK}_{d \rightarrow d-1} \circ \text{EC}^{\text{SC}(d-1)} \circ \dots \circ \text{SHRINK}_{3 \rightarrow 2} \circ \text{EC}^{\text{SC}(2)} \circ \text{SHRINK}_{2 \rightarrow 1} \quad (7.59)$$

◀

²⁶Technically, the constant β depends on the value of x , but we can always take the minimal value of β for this upper bound.

Proof. Note that $\text{UNENCODE}^{(d)}$ is defined analogously to the injection gadget $\text{INJECT}^{(d)}$, with the growth unitary inverted. The definition and analysis for its bad fault paths follow the same arguments as in Lemma 7.6. \square

Lemma 7.15 (Quantum output). There exists a constant $\epsilon_* \in (0, 1)$ such that, for any $\epsilon \in (0, 1)$ and any Clifford+T circuit C of width W and depth D outputting n qubits, there exists a circuit \bar{C} with trivial input and output types. For any family of bad error supports $\mathcal{U} \subseteq P([n])$, there is a collection of bad fault paths $\mathcal{F}_{\mathcal{U}}$ such that for any $\mathcal{F}_{\mathcal{U}}$ -avoiding Pauli fault \mathbf{f} , for any input state ρ , it holds that

$$\text{map}[\bar{C}[\mathbf{f}]](\rho) \lesssim_{\mathcal{U}} \text{map}[C](\rho). \quad (7.60)$$

Let $V = \frac{WD}{\epsilon}$. Then, \bar{C} has width \bar{W} and depth \bar{D} satisfying the bounds

$$\bar{W} = O(W \log^{5.91}(V) \text{polyloglog}(V)) \quad (7.61)$$

$$\bar{D} = O(D \log^2(V) \text{polyloglog}(V)) . \quad (7.62)$$

On $x \in [0, \epsilon_*]$, the weight enumerator of $\mathcal{F}_{\mathcal{U}}$ satisfies the bound

$$\mathcal{W}(\mathcal{F}_{\mathcal{U}}; x) \leq \epsilon + \mathcal{W}(\mathcal{U}; c \cdot x) \quad (7.63)$$

where $c > 0$ is an absolute constant.

Construction. Apply Theorem 7.11 with circuit C to get a fault-tolerant gadget C_{FT} for C with bad fault paths \mathcal{F} . Let d be the distance value chosen in Theorem 7.11. Apply the unencoding gadget $\text{UNENCODE}^{(d)}$ to all the output logical qubits of C_{FT} to obtain the circuit \bar{C} .

$$\bar{C} = \left(\text{UNENCODE}^{(d)} \right)^{\otimes n} \circ C_{\text{FT}} \quad (7.64)$$

◀

Proof. Recall that $\mathcal{F}_{\text{UNENCODE}}^d$ is the bad fault paths of the unencoding gadget $\text{UNENCODE}^{(d)}$. We define the bad fault paths of \bar{C} to be

$$\mathcal{F}_{\mathcal{U}} = \mathcal{F} \boxplus (\mathcal{U} \bullet \mathcal{F}_{\text{UNENCODE}}^d). \quad (7.65)$$

The upper bound on weight enumerator follows from Proposition 2.8.

$$\mathcal{W}(\mathcal{F}_{\mathcal{U}}; x) \leq \mathcal{W}(\mathcal{F}; x) + \mathcal{W}(\mathcal{U}; c \cdot x) \quad (7.66)$$

$$\leq \epsilon + \mathcal{W}(\mathcal{U}; c \cdot x) \quad (7.67)$$

For correctness, decompose \mathbf{f} into \mathbf{f}_1 , which is supported on the locations of C_{FT} and \mathbf{f}_2 , which is supported on the locations of the unencoding gadgets. From Theorem 7.11, we know that C_{FT} is a fault-tolerant gadget for C with output type $\text{SC}(d)_{1/5}$ on every output logical qubit. Let

σ be the output state of C given input ρ , then on every surface code block, the output state $\bar{\sigma} := \text{map}[C_{\text{FT}}[\mathbf{f}_1]](\rho)$ is $\text{SC}(d)_{1/5}$ -deviated from $\text{enc}_{\text{SC}(d)}^{\otimes n}(\sigma)$. Notationally, we write

$$\forall i \in [n], \bar{\sigma}_i = \mathcal{E}_i \circ \text{enc}_{\text{SC}(d)}(\sigma_i) \quad (7.68)$$

for some $\text{SC}(d)_{1/5}$ -avoiding superoperator error \mathcal{E}_i .

Now let $F \subseteq [n]$ denote the set of unencoding gadgets $\text{UNENCODE}^{(d)}$ on which the fault \mathbf{f}_2 is not $\mathcal{F}_{\text{UNENCODE}}^d$ -avoiding. From the definition of composition (the \bullet operation), we see that F is \mathcal{U} -avoiding. For all $i \notin F$, the i -th unencoding gadget correctly implements identity. For $i \in F$, let $\mathbf{f}_{2,i}$ denote the restriction of \mathbf{f}_2 to the locations of the i -th unencoding gadget, and define $\mathcal{N}_i := \text{map}[\text{UNENCODE}^{(d)}[\mathbf{f}_{2,i}]]$.

$$\text{map}[\bar{C}[\mathbf{f}]](\rho) = \text{map}\left[\left(\text{UNENCODE}^{(d)}\right)^{\otimes n}[\mathbf{f}_2] \circ \text{map}[C_{\text{FT}}[\mathbf{f}_1]](\rho)\right] \quad (7.69)$$

$$= \text{map}\left[\left(\text{UNENCODE}^{(d)}\right)^{\otimes n}[\mathbf{f}_2] \circ \left(\bigotimes_{i \in [n]} \mathcal{E}_i \circ \text{enc}_{\text{SC}(d)}\right)(\sigma)\right] \quad (7.70)$$

$$= \left(\bigotimes_{i \notin F} \text{dec}_{\text{SC}(d)} \circ \mathcal{E}_i \circ \text{enc}_{\text{SC}(d)}\right) \otimes \left(\bigotimes_{i \in F} \mathcal{N}_i \circ \mathcal{E}_i \circ \text{enc}_{\text{SC}(d)}\right)(\sigma) \quad (7.71)$$

$$= \left(\bigotimes_{i \in F} \mathcal{N}_i \circ \mathcal{E}_i \circ \text{enc}_{\text{SC}(d)}\right)(\sigma). \quad (7.72)$$

Since F is \mathcal{U} -avoiding, we see that $\text{map}[\bar{C}[\mathbf{f}]](\rho) \lesssim_U \sigma$. The bounds on the width and depth of \bar{C} follows a simple calculation from Theorem 7.11 and Lemma 7.14. \square

7.3 Constant space overhead fault-tolerance

Having constructed a fault-tolerance scheme with space and time overhead, we can use it to work our way up to a scheme with constant quantum space overhead [Got14; FGL20; TKY24; NP24]. Here, we follow roughly the construction of [TKY24] with surface codes replacing concatenated codes and non-single-shot error correction gadgets instead of single-shot ones. It should be noted that the theorem proved in [TKY24] is stronger in that the model of computation here does not restrict the size of the classical computation: Effectively, we have assumed the existence of a classical minimum-weight decoder which runs instantaneously. The exponent of the depth overhead in Theorem 7.21 can be reduced to $1 + o(1)$ in a model of computation with very limited or even constant depth classical computation [NP24] at the cost of a substantially more involved construction.

We first need the following fact about the existence of an asymptotically good qLDPC code family shown by [PK22a], building upon a line of work [HHO21; PK22b; BE21]. The work of [PK22a] is later extended by [LZ22; Din+23].

Fact 7.16 (Good qLDPC codes). There exists an efficiently constructable qLDPC code family indexed by $i \in \mathbb{N}$ such that for some absolute constant $c > 1$, the i -th member of the family

qLDPC(i) has parameters $n_i = \Theta(c^i)$,

$$[[n_i, \Theta(n_i), \Theta(n_i)]] \quad (7.73)$$

Definition 7.17 (Good qLDPC code family). We use $\text{qLDPC}_c^{(i)}$ to refer to the code type constructed from the i -th member of the family of Fact 7.16 with the encoding map and bad sets constructed as in Definition 7.1.

Proposition 7.18 (Transversal Clifford gates for CSS qLDPC codes). There exists a constant $\epsilon_{*,\text{qLDPCTransversal}} \in (0, 1)$ and friendly gate gadgets for the set of operations

$$\text{CSSTransversal} = \{I, X, Z, \text{CNOT}, M_X, M_Z\} \quad (7.74)$$

(and their classically controlled analogs, if unitary) such that

- The gadgets for I, X, Z, CNOT and their classically controlled analogs has input and output code types $\text{qLDPC}_{1/5}^{(i)}$. The gadgets for M_X, M_Z has input code type $\text{qLDPC}_{1/5}^{(i)}$ and trivial output code type.
- Each gadget has width $O(k_i)$ and (identical) depth $O(d_i)$.
- For each operation, $g \in \text{CSSTransversal}$, the family of bad fault paths of the corresponding gadget \mathcal{F}_g has weight enumerator bounded on $x \in [0, \epsilon_{*,\text{qLDPCTransversal}}]$ as

$$\mathcal{W}(\mathcal{F}_g; x) \leq e^{-\beta d_i} O(\text{poly}(d_i)) \quad (7.75)$$

for some constant $\beta > 0$.

Proof. All CSS codes have transversal implementations of the operations in CSSTransversal . The construction and proof is identical to Lemma 7.4 with parameters modified in the obvious way. \square

Lemma 7.19 (qLDPC resource state gadget). Let k_i be the number of qubits encoded by $\text{qLDPC}(i)$ and d_i its distance. Consider $m = O(1)$ registers of size k_i and let ψ be an mk_i -qubit state prepared by a Clifford+T circuit C_ψ of width $O(mk_i)$ and depth $O(1)$.^a

There exists an absolute constant $\epsilon_{*,\text{qLDPC PREP}} \in (0, 1)$ and a circuit PREP_ψ^i such that PREP_ψ^i is a (C_ψ, \mathcal{F}) -FT gadget with output code types $\text{qLDPC}_{1/5}^{(i)}$ for each of the m registers. PREP_ψ^i has depth D and width W where

$$W = O(k_i d_i^{5.91} \text{polylog}(d_i)) \quad (7.76)$$

$$D = O(d_i^2 \text{polylog}(d_i)) . \quad (7.77)$$

On $x \in [0, \epsilon_{*,\text{qLDPC PREP}}]$,

$$\mathcal{W}(\mathcal{F}; x) \leq \text{poly}(d_i) e^{-d_i/10} . \quad (7.78)$$

^aDepth and m $O(1)$ here means an absolute constant such as 10.

Construction. Let $\text{enc}_{\text{qLDPC}(i)}^{\text{CLIFF}}$ be a Clifford encoder for $\text{qLDPC}(i)$ which can selected to be constant depth [Got14] and width $O(mk_i)$.²⁷ Let C' be a circuit that executes C_ψ and then encodes the output (non-fault tolerantly) into m $\text{qLDPC}(i)$ blocks using $\text{enc}_{\text{qLDPC}(i)}^{\text{CLIFF}}$.

$$C' = \left(\text{enc}_{\text{qLDPC}(i)}^{\text{CLIFF}} \right)^{\otimes m} \circ C_\psi \quad (7.79)$$

The gadget is the application of Lemma 7.15 to C' with $\epsilon \equiv \epsilon_{\text{prep}} \in (0, 1)$ to be selected later and $\mathcal{U} = \sum_{i=1}^m \text{qLDPC}_{1/5}^{(i)}$ ²⁸ to obtain a $(C', \mathcal{F}_{\mathcal{U}})$ -FT gadget PREP_{ψ}^i . \blacktriangleleft

Proof. Let $\epsilon_{*,\text{PREP}} \in (0, 1)$ be the threshold value from the statement of Lemma 7.15. From Lemma 7.15, for some absolute constant $c > 0$, the weight enumerator of $\mathcal{F} \equiv \mathcal{F}_{\mathcal{U}}$ can be bounded on $x \in [0, \epsilon_{*,\text{PREP}}]$ as

$$\mathcal{W}(\mathcal{F}; x) \leq \epsilon_{\text{prep}} + m \cdot \mathcal{W}\left(\text{qLDPC}_{1/5}^{(i)}; c \cdot x\right). \quad (7.80)$$

Let d_i be the distance of $\text{qLDPC}(i)$, we can apply Lemma 6.13 to bound the weight enumerator of the code type as

$$\mathcal{W}\left(\text{qLDPC}_{1/5}^{(i)}; c \cdot x\right) \leq \text{poly}(d_i) (O_{\Delta}(x))^{d_i/10}. \quad (7.81)$$

Let $\epsilon' \in (0, \epsilon_{*,\text{PREP}})$ be a small enough constant so that the term in the base of the exponent is at most $1/e$, so that on $x \in [0, \epsilon']$

$$\mathcal{W}\left(\text{qLDPC}_{1/5}^{(i)}; c \cdot x\right) \leq \text{poly}(d_i) (O_{\Delta}(x))^{d_i/10} \leq \text{poly}(d_i) e^{-d_i/10} \equiv f(d_i) \quad (7.82)$$

We now set $\epsilon_{\text{prep}} = f(d_i)$ and $\epsilon_{*,\text{qLDPC PREP}} = \min(\epsilon', \epsilon_{*,\text{PREP}})$, so that overall, the weight enumerator of the bad fault paths on $x \in [0, \epsilon_{*,\text{qLDPC PREP}}]$ is exponentially small in the code distance after using the restriction on m

$$\mathcal{W}(\mathcal{F}; x) \leq O(1) f(d_i). \quad (7.83)$$

Lemma 7.15 was invoked on a constant depth circuit of width $O(mk_i) = O(\text{poly}(d_i))$ with $\epsilon_{\text{prep}} = f(d_i)$. Let $V' = \frac{mk_i}{f(d_i)} = e^{d_i \text{polylog}(d_i)}$. It follows that PREP_{ψ}^i has depth D and width W where

$$\begin{aligned} W &= O(mk_i \log^{5.91}(V') \text{polyloglog}(V')) = O(k_i d_i^{5.91} \text{polylog}(d_i)) \\ D &= O(\log^2(V') \text{polyloglog}(V')) = O(d_i^2 \text{polylog}(d_i)). \end{aligned} \quad (7.84) \quad \square$$

²⁷The decoder must be consistent with the decoder unitary, see Remark 4.3. The precise choice of logical basis for $\text{qLDPC}(i)$ is arbitrary until this point.

²⁸We attach the code type $\text{qLDPC}_{1/5}^{(i)}$ to bundles of qubits originating from each application of $\text{enc}_{\text{qLDPC}(i)}^{\text{CLIFF}}$ \mathcal{U} is the sum of $\text{qLDPC}_{1/5}^{(i)}$ imposed on each bundle of output qubits.

Corollary 7.20 (qLDPC gate gadget). There exists a constant $\epsilon_{*,\text{GATE}} \in (0, 1)$ such that for a one or two-qubit operation \mathbf{g} in the set

$$\text{GATESET} = \{\mathbf{T}, \mathbf{I}(A), \mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{CNOT}, \mathbf{H}, \mathbf{S}, \mathbf{M}_X(A), \mathbf{M}_Z(A), \mathbf{INIT}_Z(A)\} \quad (7.85)$$

with support on qubits from at most two registers of size k_i , there exists a circuit $\text{GATE}_{\mathbf{g}}$ that is a $(\mathbf{g}, \mathcal{F}_{\mathbf{g}})$ -FT gadget with input and output types $\text{qLDPC}_{1/5}^{(i)}$. On $x \in [0, \epsilon_{*,\text{GATE}}]$ the weight enumerator of $\mathcal{F}_{\mathbf{g}}$ is bounded as

$$\mathcal{W}(\mathcal{F}_{\mathbf{g}}; x) \leq \text{poly}(d_i) e^{-\beta d_i} . \quad (7.86)$$

$\text{GATE}_{\mathbf{g}}$ has width W and depth D where

$$W = O(k_i d_i^{5.91} \text{polylog}(d_i)) \quad (7.87)$$

$$D = O(d_i^2 \text{polylog}(d_i)) . \quad (7.88)$$

$\text{GATE}_{\mathbf{I}(A)}$ (identity) has width $W = O(k_i)$. The depth is independent of \mathbf{g} .

Construction. For a gate $\mathbf{g} = \text{GATESET} \setminus \{\mathbf{I}(A), \mathbf{INIT}_Z(A), \mathbf{M}_X(A), \mathbf{M}_Z(A)\}$ supported on m registers, we use Lemma 7.19 to prepare $|\mathbf{g}\rangle = (\mathbf{g} \otimes I) |\phi_+\rangle^{mk_i}$ where \mathbf{g} should be interpreted as acting on one side of the Bell pair. Let $t \leq 3$ be the level of the Clifford hierarchy that \mathbf{g} is in. $|\mathbf{g}\rangle$ can be consumed by a teleportation circuit using operations only in CSSTRANSVERSAL and one operation in level $t' = \max(t - 1, 1)$ of the Clifford hierarchy [GC99]. Let $C_{\mathbf{g}}$ be the circuit that inductively performs the previous procedure until $t' = 1$ and then directly applies an operation $\{\mathbf{I}, \mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$. The gadget is $C_{\mathbf{g}}$ with every gate replaced by a gadget from Proposition 7.18. For $\mathbf{g} \in \{\mathbf{I}(A), \mathbf{INIT}_Z(A), \mathbf{M}_X(A), \mathbf{M}_Z(A)\}$, $C_{\mathbf{g}}$ is the corresponding circuit from CSSTRANSVERSAL . The gadgets are padded with error correction gadgets $\text{EC}^{\text{qLDPC}(i)}$ (Theorem 6.23) such that they are of uniform size. ◀

Proof. Set $\epsilon_{*,\text{GATE}} = \min(\epsilon_{*,\text{qLDPCTransversal}}, \epsilon_{*,\text{qLDPC PREP}})$. The width and depth bounds follow from the respective gadgets. The weight enumerator bounds follow from having composed a constant number of gadgets, each with weight enumerator bounded as $\text{poly}(d_i)e^{-\beta d_i}$ on $[0, \epsilon_{*,\text{GATE}}]$, and from the depth bound (for padding). ◻

Theorem 7.21 (Constant space overhead threshold theorem). There exists a constant $\epsilon_* \in (0, 1)$ such that, for any Clifford+T circuit C with classical input and classical output of width W , depth D , and $\epsilon \in (0, 1)$ satisfying^a $W \geq \log^7\left(\frac{WD}{\epsilon}\right)$, there exists a circuit \overline{C} that is a fault-tolerant gadget for C with trivial input and output and bad fault paths \mathcal{F} . Let $V = \frac{WD}{\epsilon}$. Then, \overline{C} has width \overline{W} and depth \overline{D} satisfying the bounds

$$\overline{W} = O(W) \quad (7.89)$$

$$\overline{D} = O(D \log^{8.91}(V) \text{polyloglog}(V)) . \quad (7.90)$$

On $x \in [0, \epsilon_*]$, the weight enumerator of \mathcal{F} satisfies the bound

$$\mathcal{W}(\mathcal{F}; x) \leq \epsilon. \quad (7.91)$$

^aThis condition rules out exponentially deep circuits but is otherwise not very restrictive.

Construction. Let $i \in \mathbb{N}$ be a parameter to be determined later. Without loss of generality, we may assume C contains only gates from the following gate set

$$\text{GATESET} = \{\text{T}, \text{I}, \text{X}, \text{Z}, \text{CNOT}, \text{H}, \text{S}, \text{M}_X, \text{M}_Z, \text{INIT}_Z\}. \quad (7.92)$$

Otherwise, we may rewrite C in terms of the gates from GATESET at constant depth cost.

For each timestep $t \in [D]$ of C , let W_t be the number of qubits. We will have $m_t = \lceil W_t/k_i \rceil + 3 = \Theta(W_t/k_i)$ registers of k_i qubits. Arbitrarily²⁹ make an injective assignment of the quantum inputs (or output in the case of INIT_Z) of the gates of C at timestep t to the qubit coordinates $[k_i] \times [m_t]$ indexing the qubits of the m_t registers. We restrict the assignment such that when one of $\{\text{M}_X, \text{M}_Z, \text{INIT}_Z\}$ is assigned to a register, no other operations (including I) are assigned to that register. This assignment also induces an assignment of the outputs.

For any permutation of N qubits, there exists an efficiently computable depth-2 circuit of SWAP gates that implements it. We now construct a new circuit C' that implements the following steps for every timestep $t \in [D]$ of C :

1. Implement the permutation that routes the outputs of gates at timestep $t - 1$ of C to the appropriate location for timestep t . Note that there are “holes” in the register coordinates $[m_{t-1}]$ corresponding to blocks containing qubits that are acted on by (M_X/M_Z) and in the coordinates $[m_t]$ corresponding to blocks containing qubits that will be initialized by an INIT_Z .
2. Execute all gates of C at time t on the qubits with the targets given by the previous assignment to qubit coordinates. When a qubit of the block is acted on by one of $\{\text{M}_X, \text{M}_Z\}$ perform the operation on all qubits of the block (i.e. $\text{M}_X(A)$ or $\text{M}_Z(A)$). Likewise perform the $\text{INIT}_Z(A)$ to initialize a block when there is a qubit assigned to that block originating from INIT_Z . For any qubit that does not correspond to a qubit of C , execute I .

After replacing the $\{\text{INIT}_Z, \text{M}_X, \text{M}_Z\}$ operations supported on blocks with $\{\text{INIT}_Z(A), \text{M}_X(A), \text{M}_Z(A)\}$, every operation in C' is an operation for which there is a corresponding gadget from Corollary 7.20.

Let D' be the depth of C' and W' its width. Let $\ell \in [1, W'/k_i]$ be a parameter to be selected later that will control the maximum number of gates executed in a timestep. The circuit \bar{C} will be constructed from C' by executing at most ℓ gate gadgets in each timestep.

Let $t \in [D']$ be a timestep of C' . The operations will be executed over τ steps such that no step contains more than ℓ operations. This can be done by first constructing a maximum-degree k_i multigraph with a vertex for each register and an edge for each gate supported on qubits contained in both registers. An edge coloring c of the multigraph can be computed using at most $O(k_i)$ colors. We can compute a new edge coloring c' from c by subdividing each color into at most $O((W/k_i)/\ell)$ new colors such that the number of edges in each color is at most ℓ . c' has at most $O(W/\ell)$ colors (see [NP24, Serialization Lemma]). From c' , a partitioning of the gates³⁰ of C' in timestep t can

²⁹This can be computed greedily.

³⁰Single qubit gates are executed greedily subject to the constraints.

be computed such that:

1. There are at most ℓ non-trivial gates (not in \mathbf{l}) per partition.
2. In each partition, no non-trivial two gates are supported on the same register.
3. There are $O(W/\ell)$ partitions.

\overline{C} is constructed by the following procedure: For each timestep t of C' and for each gate g in the partitioning associated with timestep t , use the gadget GATE_g (from Corollary 7.20) to execute g and execute $\text{GATE}_{\mathbf{l}(A)}$ on all other blocks. \blacktriangleleft

Proof. Set $\epsilon_* = \epsilon_{*,\text{GATE}}$ from Corollary 7.20. Let $m = \max_{t \in [D]} m_t = O(W/k_i)$ be the maximum number of registers of C . \overline{C} is composed entirely of GATE gadgets from Corollary 7.20, so the width and depth bounds will follow. Let \overline{W} and \overline{D} be the width and depth of \overline{C} , respectively. In terms of a function $f(d_i) = \Theta(d_i^{5.91} \text{polylog}(d_i))$, (essentially corresponding to the width overhead of GATE) the width and depth satisfy the bounds

$$\overline{W} = O(\ell k_i f(d_i) + (m - \ell)k_i) \quad (7.93)$$

$$\overline{D} = O\left(D \frac{W}{\ell} d_i^2 \text{polylog}(d_i)\right). \quad (7.94)$$

Using Proposition 4.9, we can show that \overline{C} is a (C, \mathcal{F}) -FT gadget where \mathcal{F} is the sum of the bad fault paths of all GATE gadgets in the circuit. There are $O(m \cdot W/\ell \cdot D)$ of these. Thus, for $x \in [0, \epsilon_*]$, we have the bound

$$\mathcal{W}(\mathcal{F}; x) \leq O(m \cdot W/\ell \cdot D) \text{poly}(d_i) e^{-\beta d_i}. \quad (7.95)$$

We now set $\ell = \lceil m/f(d_i) \rceil$ and use the linear distance $k_i = \Theta(d_i)$, so that these bounds become

$$\mathcal{W}(\mathcal{F}; x) \leq O(WD f(d_i) \text{poly}(d_i) e^{-\beta d_i}) \quad (7.96)$$

$$\leq O(WD \text{poly}(d_i) e^{-\beta d_i}). \quad (7.97)$$

It follows that $d_i = O(\log(V) \text{polyloglog}(V))$ is sufficient such that the inequality $\mathcal{W}(\mathcal{F}; x) \leq \epsilon$ holds. In other words, we pick $i = \Theta(\log\log(V))$ (see Fact 7.16), utilizing the exponential spacing of the code family. The bounds on the width and depth follow from the parameter choices and the assumption that the circuit width satisfies $W \geq \log^7(V)$.

$$\overline{W} = O(W + k_i f(d_i)) \quad (7.98)$$

$$= O(W + \log^{6.91}(V) \text{polyloglog}(V)) \quad (7.99)$$

$$= O(W) \quad (7.100)$$

$$\overline{D} = O(Dk_i f(d_i) d_i^2 \text{polylog}(d_i)) \quad (7.101)$$

$$= O(Dd_i^{8.91} \text{polylog}(d_i)) \quad (7.102)$$

$$= O(D \log^{8.91}(V) \text{polyloglog}(V)) \quad (7.103)$$

This completes our threshold proof for a qLDPC code- and teleportation-based constant overhead fault-tolerant scheme. \square

Nomenclature

Weight Enumerators

$\mathcal{W}(\mathcal{F}; x)$	Weight enumerator function of a collection of sets \mathcal{F} , see Definition 2.3.
$\mathcal{F}, \boxplus, \otimes$	Ring of bad sets, see Definition 2.5. \mathcal{F} is a collection of bad sets, and \boxplus, \otimes denote addition and multiplication of such collections.
$\mathcal{F} \bullet \{\mathcal{S}_i\}_i$	Composition of families of sets, see Definition 2.7.

Types and Operators

$G = (V, E)$	A directed, acyclic multigraph that represents a network, and later a circuit. See Definition 3.1.
$\text{src}(e), \text{dest}(e)$	Source and destination vertices of a directed edge e .
$\text{in}(v), \text{out}(v)$	Edges that have v as destination and edges that have v as source.
conn	Labeling of edges in the network that specifies the sources and destinations of edges.
\top, \perp	Special input and output vertices of a network.
$\mathsf{T}: V \rightarrow [D]$	Time labeling of vertices in a network. D is the depth of the network. For every $t \in [D]$, the set of vertices at time t is called a timestep.
$\mathbf{t}, \mathbf{t}_\bullet$	Type of a connection, and a sequence of types, see Definition 3.3.
$\mathcal{H}_{\mathbf{t}_\bullet}, \mathsf{D}(\mathcal{H}_{\mathbf{t}_\bullet})$	Pure states and mixed states of type \mathbf{t}_\bullet . We abbreviate $\mathsf{D}(\mathcal{H}_{\mathbf{t}_\bullet})$ as $\mathsf{D}(\mathbf{t}_\bullet)$.
$\mathsf{L}(\mathbf{t}_\bullet^{\text{out}}, \mathbf{t}_\bullet^{\text{in}})$	Set of operators of input type $\mathbf{t}_\bullet^{\text{in}}$ and output type $\mathbf{t}_\bullet^{\text{out}}$, see Definition 3.6. We use the shorthand $\mathsf{L}(\mathcal{H}), \mathsf{L}(\mathbf{t}_\bullet)$ when the input and output spaces are the same.
$\mathcal{L}(\mathbf{t}_\bullet^{\text{out}}, \mathbf{t}_\bullet^{\text{in}})$	Set of superoperators of input type $\mathbf{t}_\bullet^{\text{in}}$ and output type $\mathbf{t}_\bullet^{\text{out}}$, see Definition 3.7.
\mathcal{G}	A gate, which is an element in $\mathcal{L}(\mathbf{t}_\bullet^{\text{out}}, \mathbf{t}_\bullet^{\text{in}})$.

Circuits and Faults

type	A labeling that assigns a type to every connection in a network.
gate	A labeling that assigns a gate to every vertex in a network, which has the correct types according to type .
C	A classical-quantum circuit is fully specified by $C = (V, E, \text{conn}, \text{type}, \text{gate})$, see Definition 3.9. We often refer to the vertices and edges of a circuit as locations and wires.
W, W_C	The quantum width and classical width of a circuit.

$\text{map}[C]$	The map implemented by circuit C , see Definition 3.11.
κ, C_κ	contraction map and contracted circuit, see Definition 3.12.
$\mathbf{f} = (\kappa, \widetilde{\text{gate}})$	A fault in a circuit is defined by a depth-preserving contraction map κ and a set of faulty gates $\widetilde{\text{gate}}$ which replaced the ideal gates, see Definition 3.13.
$\text{supp}(\mathbf{f})$	The fault path of a fault \mathbf{f} , i.e., the locations of the faulty gates.
C_{env}, C_Ω	Circuit C with environment refers to an (arbitrary) extension $C_{\text{env}} = (V \sqcup V_\Omega, E \sqcup E_\Omega, \text{conn}', \text{type}', \text{gate}')$ of the original circuit C . The sub-circuit C_Ω induced by the nodes V_Ω is called the environment circuit of C_{env} .

Fault-Tolerant Gadgets

$\sigma \lesssim_{\mathcal{B}} \tilde{\sigma}$	Notation to represent that for two states σ and $\tilde{\sigma}$, $\tilde{\sigma}$ is \mathcal{B} -deviated from σ , see Definition 4.1.
$\mathcal{Q}, (U, \mathcal{B})$	A quantum code \mathcal{Q} with code type $\text{ctype}_{\mathcal{Q}} = (U, \mathcal{B})$, where U is the decoding unitary and \mathcal{B} is a collection of bad error supports, see Definition 4.2. We sometimes write $\text{ctype}_{\mathcal{Q}}$ -deviated and $\lesssim_{\text{ctype}_{\mathcal{Q}}}$ in place of \mathcal{B} -deviated and $\lesssim_{\mathcal{B}}$.
renc, rdec	Reversible encoders and decoders of a code.
enc, dec	Irreversible encoders and decoders of a code.
$\mathcal{E}, K_\mu, K'_\nu$	We use \mathcal{E} to denote an error superoperator acting on states. Such a superoperator can be written as a linear combination of Pauli superoperators, thereby admitting a decomposition $\mathcal{E}(\sigma) = \sum_{\mu, \nu} \alpha_{\mu, \nu} K_\mu \sigma K'_\nu$ for Pauli operators $\{K_\mu\}_\mu, \{K'_\nu\}_\nu$ and complex coefficients $\{\alpha_{\mu, \nu}\}_{\mu, \nu}$.
β, spec	β is a bundling of edges for a circuit C . spec is a code specification which assigns code types to bundled edges, see Definition 4.5, Definition 4.6.
$\mathcal{F}, \tilde{\mathcal{G}}_{\mathbf{f}}, \mathcal{R}_{\mathbf{f}}$	A circuit C is a fault-tolerant gadget that simulates \mathcal{G} with respect to the data $(\beta, \text{spec}, \mathcal{F})$ if for every \mathcal{F} -avoiding fault \mathbf{f} , there exists superoperators $\tilde{\mathcal{G}}_{\mathbf{f}}, \mathcal{R}_{\mathbf{f}}$ such that Eq. (4.15) and the properties defined in Definition 4.8 hold.
$\mathcal{N}_{\mathcal{G}, \mathbf{f}}, \mathcal{N}_{\mathcal{R}, \mathbf{f}}$	Noise superoperators acting on the syndrome subsystem of an unencoded state, which came from $\tilde{\mathcal{G}}_{\mathbf{f}}$ and $\mathcal{R}_{\mathbf{f}}$ of a gadget. See Lemma 4.10.

Error Correction for CSS Codes

H_X, H_Z, Δ	A CSS code has stabilizer check matrices H_X, H_Z . We say that the code is Δ -qLDPC if every row and column of H_X and H_Z has weight at most Δ .
$G_{\text{adj}}[H]$	The adjacency graph of a parity check matrix H , see Definition 6.2.
$\mathcal{CG}_{(d, t)}$	The (d, t) -clustering sets of a graph $G = (V, E)$ which are all size t subsets of vertices of d -vertex connected subgraphs of G . See Definition 6.4.

$\mathcal{CO}(m, S)$	For a set of vertices S of graph G , $\mathcal{CO}(m, S)$ is the collection of size m cluster covers of S , see Lemma 6.5.
$(U_{\mathcal{Q}}, \mathcal{B}_c)$	$\text{CompCode}_c = (U_{\mathcal{Q}}, \mathcal{B}_c)$ is the code type of a computational qLDPC code \mathcal{Q} , parametrized by a constant $c \in [0, 1]$. $U_{\mathcal{Q}}$ is the purification of a decoder. See Definition 6.7 and Definition 6.8.
H, T, I_t, J_t, ϕ	H is the spacetime check matrix (commonly referred to as the detector check matrix) of T rounds of measurements of H_Z . For $t \in [T]$, I_t denotes the detectors at time t , J_t^d, J_t^s denote the data bits and syndrome bits. ϕ is the flattening map. See Definition 6.14.
$H_{\text{bulk}}, \phi_{\text{bulk}}$	The bulk check matrix and flattening map, which is defined with J_1^d and I_1 removed.
H_{ext}	The extended spacetime check matrix with residue error J_{T+1}^d , see Definition 6.18.

Gadgets for qLDPC Codes

Gad_{EC}	The error correction gadget defined by performing d rounds of syndrome extraction, see Theorem 6.23.
$\text{INIT}_Z, \text{INIT}_X$	Gadgets which initialize a CSS code state in the all $ 0\rangle$ or $ +\rangle$ basis, see Lemma 6.30.
$\text{SC}(d), \text{SC}_c^{(d)}$	Surface code of distance d and its code type, parametrized by constant c .
$\text{INJECT}_{ \psi\rangle}^{(d)}$	Surface code state injection gadget, see Lemma 7.6.
$\text{PREPARE}_{ T\rangle}$	Logical level state distillation gadget, which prepares a $ T\rangle$ state by repeated rounds of 15-to-1 distillation.
$\text{UNENCODE}^{(d)}$	Surface code unencoding gadget, see Lemma 7.14.
$\text{qLDPC}(i)$	The i -th member of a family of asymptotically good codes. Its code type is denoted $\text{qLDPC}_c^{(i)}$.
$\text{PREP}_{ \psi\rangle}^i$	Resource state preparation gadget for qLDPC codes, see Lemma 7.19.
$\text{GATE}_{\mathbf{g}}$	qLDPC gate gadget for a universal set of gates, see Corollary 7.20.

References

- [ABO97] Dorit Aharonov and Michael Ben-Or. “Fault-tolerant quantum computation with constant error”. In: (1997), pp. 176–188 (cit. on pp. 1, 5, 18, 35).
- [AGP05] Panos Aliferis, Daniel Gottesman, and John Preskill. “Quantum accuracy threshold for concatenated distance-3 codes”. In: *arXiv preprint quant-ph/0504218* (2005) (cit. on pp. 3, 5, 22, 23, 26, 34, 35).
- [AGP07] Panos Aliferis, Daniel Gottesman, and John Preskill. “Accuracy threshold for post-selected quantum computation”. In: *arXiv preprint quant-ph/0703264* (2007) (cit. on pp. 39, 40).
- [Bev+20] Michael Beverland, Earl Campbell, Mark Howard, and Vadym Kliuchnikov. “Lower bounds on the non-Clifford resources for quantum computations”. In: *Quantum Science and Technology* 5.3 (2020), p. 035009 (cit. on p. 71).
- [BMD06] H. Bombin and M. A. Martin-Delgado. “Topological Quantum Distillation”. In: *Phys. Rev. Lett.* 97 (18 Oct. 2006), p. 180501 (cit. on p. 55).
- [Bom15] Héctor Bombín. “Gauge color codes: optimal transversal gates and gauge fixing in topological stabilizer codes”. In: *New Journal of Physics* 17.8 (2015), p. 083002 (cit. on pp. 54, 55).
- [Bom16] Héctor Bombín. “Dimensional jump in quantum error correction”. In: *New Journal of Physics* 18.4 (2016), p. 043038 (cit. on p. 55).
- [BMD09] Héctor Bombín and Miguel Angel Martin-Delgado. “Quantum measurements and gates by code deformation”. In: *Journal of Physics A: Mathematical and Theoretical* 42.9 (2009), p. 095302 (cit. on p. 55).
- [BSS16] S. Bravyi, G. Smith, and J. A. Smolin. “Trading Classical and Quantum Computational Resources”. In: *Physical Review X* 6.2 (June 2016). ISSN: 2160-3308 (cit. on p. 55).
- [Bra+18] Sergey Bravyi, Matthias Englbrecht, Robert König, and Nolan Peard. “Correcting coherent errors with surface codes”. In: *npj Quantum Information* 4.1 (2018), p. 55 (cit. on p. 5).
- [BH12] Sergey Bravyi and Jeongwan Haah. “Magic-state distillation with low overhead”. In: *Physical Review A—Atomic, Molecular, and Optical Physics* 86.5 (2012), p. 052329 (cit. on p. 71).
- [BK05] Sergey Bravyi and Alexei Kitaev. “Universal quantum computation with ideal Clifford gates and noisy ancillas”. In: *Physical Review A—Atomic, Molecular, and Optical Physics* 71.2 (2005), p. 022316 (cit. on pp. 3, 63, 66, 71).
- [BK98] Sergey B Bravyi and A Yu Kitaev. “Quantum codes on a lattice with boundary”. In: *arXiv preprint quant-ph/9811052* (1998) (cit. on pp. 60, 61).
- [BB24] Nikolas P. Breuckmann and Simon Burton. “Fold-Transversal Clifford Gates for Quantum Codes”. In: *Quantum* 8 (June 2024), p. 1372. ISSN: 2521-327X (cit. on p. 62).
- [BE21] Nikolas P. Breuckmann and Jens N. Eberhardt. “Balanced Product Quantum Codes”. In: *IEEE Transactions on Information Theory* 67.10 (2021), pp. 6653–6674 (cit. on p. 74).

- [BKV24] Jacob C. Bridgeman, Aleksander Kubica, and Michael Vasmer. “Lifting Topological Codes: Three-Dimensional Subsystem Codes from Two-Dimensional Anyon Models”. In: *PRX Quantum* 5.2 (Apr. 2024). ISSN: 2691-3399 (cit. on p. 54).
- [Cai+25] Madelyn Cain, Dolev Bluvstein, Chen Zhao, Shouzen Gu, Nishad Maskara, Marcin Kalinowski, Alexandra A Geim, Aleksander Kubica, Mikhail D Lukin, and Hengyun Zhou. “Fast correlated decoding of transversal logical algorithms”. In: *arXiv preprint arXiv:2505.13587* (2025) (cit. on p. 6).
- [CS96] A Robert Calderbank and Peter W Shor. “Good quantum error-correcting codes exist”. In: *Physical Review A* 54.2 (1996), p. 1098 (cit. on p. 55).
- [CFG24] Matthias Christandl, Omar Fawzi, and Ashutosh Goswami. “Fault-tolerant quantum input/output”. In: *arXiv preprint arXiv:2408.05260* (2024) (cit. on pp. 5, 22).
- [Coh+22] Lawrence Z. Cohen, Isaac H. Kim, Stephen D. Bartlett, and Benjamin J. Brown. “Low-overhead fault-tolerant quantum computing using long-range connectivity”. In: *Science Advances* 8.20 (2022), eabn1717. eprint: <https://www.science.org/doi/pdf/10.1126/sciadv.abn1717> (cit. on p. 55).
- [Cow24] A. Cowtan. “SSIP: automated surgery with quantum LDPC codes”. In: *arXiv preprint arXiv:2407.09423* (2024) (cit. on p. 55).
- [Cow+25] Alexander Cowtan, Zhiyang He, Dominic J. Williamson, and Theodore J. Yoder. “Parallel Logical Measurements via Quantum Code Surgery”. In: *arXiv preprint arXiv:2503.05003* (2025) (cit. on p. 55).
- [Cro+24] Andrew Cross, Zhiyang He, Patrick Rall, and Theodore Yoder. “Improved QLDPC Surgery: Logical Measurements and Bridging Codes”. In: *arXiv preprint arXiv:2407.18393* (2024) (cit. on p. 55).
- [Dav+24] Margarita Davydova, Nathanan Tantivasadakarn, Shankar Balasubramanian, and David Aasen. “Quantum computation from dynamic automorphism codes”. In: *Quantum* 8 (Aug. 2024), p. 1448. ISSN: 2521-327X (cit. on p. 55).
- [Den+02] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. “Topological quantum memory”. In: *Journal of Mathematical Physics* 43.9 (2002), pp. 4452–4505 (cit. on pp. 3, 38, 50, 60–63, 71).
- [Din+23] Irit Dinur, Min-Hsiu Hsieh, Ting-Chun Lin, and Thomas Vidick. “Good Quantum LDPC Codes with Linear Time Decoders”. In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. STOC 2023. Orlando, FL, USA: Association for Computing Machinery, 2023, 905–918. ISBN: 9781450399135 (cit. on p. 74).
- [FGL20] Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. “Constant overhead quantum fault tolerance with quantum expander codes”. In: *Communications of the ACM* 64.1 (2020), pp. 106–114 (cit. on pp. 54, 74).
- [FG18] Austin G. Fowler and Craig Gidney. *Low overhead quantum computation using lattice surgery*. 2018 (cit. on p. 55).
- [Fow+12] Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland. “Surface codes: Towards practical large-scale quantum computation”. In: *Phys. Rev. A* 86 (3 Sept. 2012), p. 032324 (cit. on p. 55).

- [FM01] Michael H Freedman and David A Meyer. “Projective plane and planar quantum codes”. In: *Foundations of Computational Mathematics* 1.3 (2001), pp. 325–332 (cit. on p. 60).
- [GB25] Craig Gidney and Thiago Bergamaschi. “A constant rate quantum computer on a line”. In: *arXiv preprint arXiv:2502.16132* (2025) (cit. on p. 5).
- [GG25] Louis Golowich and Venkatesan Guruswami. “Asymptotically good quantum codes with transversal non-clifford gates”. In: *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*. 2025, pp. 707–717 (cit. on p. 71).
- [Got14] Daniel Gottesman. “Fault-tolerant quantum computation with constant overhead”. In: *Quantum Information & Computation* 14.15-16 (2014), pp. 1338–1372 (cit. on pp. 1, 3, 4, 38, 39, 41, 42, 46, 50, 74, 76).
- [GC99] Daniel Gottesman and Isaac L Chuang. “Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations”. In: *Nature* 402.6760 (1999), pp. 390–393 (cit. on p. 77).
- [Gu+24] Shouzhen Gu, Eugene Tang, Libor Caha, Shin Ho Choe, Zhiyang He, and Aleksander Kubica. “Single-shot decoding of good quantum LDPC codes”. In: *Communications in Mathematical Physics* 405.3 (2024), p. 85 (cit. on p. 54).
- [Has17] Mathew B. Hastings. “Weight reduction for quantum codes”. In: *Quant. Inf. Comput.* 17.15-16 (2017), pp. 1307–1334 (cit. on p. 55).
- [Has21] Matthew B Hastings. “On quantum weight reduction”. In: *arXiv preprint arXiv:2102.1-0030* (2021) (cit. on p. 55).
- [HH18] Matthew B Hastings and Jeongwan Haah. “Distillation with sublogarithmic overhead”. In: *Physical review letters* 120.5 (2018), p. 050504 (cit. on p. 71).
- [HH21] Matthew B. Hastings and Jeongwan Haah. “Dynamically Generated Logical Qubits”. In: *Quantum* 5 (Oct. 2021), p. 564. ISSN: 2521-327X (cit. on p. 55).
- [HHO21] Matthew B. Hastings, Jeongwan Haah, and Ryan O’Donnell. “Fiber bundle codes: breaking the $n^{1/2}$ polylog(n) barrier for Quantum LDPC codes”. In: *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2021. Virtual, Italy: Association for Computing Machinery, 2021, 1276–1288. ISBN: 9781450380539 (cit. on p. 74).
- [He+25] Zhiyang He, Alexander Cowtan, Dominic J Williamson, and Theodore J Yoder. “Extractors: QLDPC Architectures for Efficient Pauli-Based Computation”. In: *arXiv preprint arXiv:2503.10390* (2025) (cit. on p. 55).
- [Hor+12] Dominic Horsman, Austin G Fowler, Simon Devitt, and Rodney Van Meter. “Surface code quantum computing by lattice surgery”. In: *New Journal of Physics* 14.12 (Dec. 2012), p. 123011. ISSN: 1367-2630 (cit. on pp. 5, 55).
- [Ide+24] Benjamin Ide, Manoj G Gowda, Priya J Nadkarni, and Guillaume Dauphinais. “Fault-tolerant logical measurements via homological measurement”. In: *arXiv preprint arXiv:2410.02753* (2024) (cit. on p. 55).
- [IP20] Joseph K Iverson and John Preskill. “Coherence in logical quantum channels”. In: *New Journal of Physics* 22.7 (2020), p. 073066 (cit. on p. 5).
- [Kit97] A Yu Kitaev. “Quantum computations: algorithms and error correction”. In: *Russian Mathematical Surveys* 52.6 (1997), p. 1191 (cit. on pp. 1, 5, 18, 22, 23, 61).

- [Kit03] A Yu Kitaev. “Fault-tolerant quantum computation by anyons”. In: *Annals of physics* 303.1 (2003), pp. 2–30 (cit. on p. 61).
- [KSV02] Alexei Yu Kitaev, Alexander Shen, and Mikhail N Vyalyi. *Classical and quantum computation*. 47. American Mathematical Soc., 2002 (cit. on p. 13).
- [KLZ96] Emanuel Knill, Raymond Laflamme, and Wojciech Zurek. “Threshold accuracy for quantum computation”. In: *arXiv preprint quant-ph/9610011* (1996) (cit. on p. 1).
- [KP13] Alexey A Kovalev and Leonid P Pryadko. “Fault tolerance of quantum low-density parity check codes with sublinear distance scaling”. In: *Physical Review A* 87.2 (2013), p. 020304 (cit. on pp. 38, 39, 41, 42).
- [KB15] Aleksander Kubica and Michael E Beverland. “Universal transversal gates with color codes: A simplified approach”. In: *Physical Review A* 91.3 (2015), p. 032330 (cit. on p. 55).
- [KV22] Aleksander Kubica and Michael Vasmer. “Single-shot quantum error correction with the three-dimensional subsystem toric code”. In: *Nature Communications* 13.1 (Oct. 2022). ISSN: 2041-1723 (cit. on p. 54).
- [KYP15] Aleksander Kubica, Beni Yoshida, and Fernando Pastawski. “Unfolding the color code”. In: *New Journal of Physics* 17.8 (Aug. 2015), p. 083026. ISSN: 1367-2630 (cit. on p. 55).
- [LZ22] A. Leverrier and G. Zemor. “Quantum Tanner codes”. In: *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*. Los Alamitos, CA, USA: IEEE Computer Society, Nov. 2022, pp. 872–883 (cit. on p. 74).
- [LZ23] Anthony Leverrier and Gilles Zémor. “Decoding quantum Tanner codes”. In: *IEEE Trans. Inf. Theory* (2023) (cit. on p. 54).
- [Li15] Ying Li. “A magic state’s fidelity can be superior to the operations that created it”. In: *New Journal of Physics* 17.2 (2015), p. 023037 (cit. on pp. 65, 71).
- [Lit19] D. Litinski. “A Game of Surface Codes: Large-Scale Quantum Computing with Lattice Surgery”. In: *Quantum* 3 (Mar. 2019), p. 128. ISSN: 2521-327X (cit. on p. 55).
- [Łod+15] Justyna Łodyga, Paweł Mazurek, Andrzej Grudka, and Michał Horodecki. “Simple scheme for encoding and decoding a qubit in unknown state for various topological codes”. In: *Scientific reports* 5.1 (2015), p. 8975 (cit. on p. 65).
- [Mou16] Jonathan E. Moussa. “Transversal Clifford gates on folded surface codes”. In: *Physical Review A* 94.4 (Oct. 2016). ISSN: 2469-9934 (cit. on p. 62).
- [Ngu25] Quynh T Nguyen. “Good binary quantum codes with transversal CCZ gate”. In: *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*. 2025, pp. 697–706 (cit. on p. 71).
- [NP24] Quynh T Nguyen and Christopher A Pattison. “Quantum fault tolerance with constant-space and logarithmic-time overheads”. In: *arXiv preprint arXiv:2411.03632* (2024) (cit. on pp. 2, 4, 5, 7, 32, 54, 71, 74, 78).
- [NC12] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, June 2012. ISBN: 9780511976667 (cit. on p. 65).

- [PR13] Adam Paetzniack and Ben W. Reichardt. “Universal Fault-Tolerant Quantum Computation with Only Transversal Gates and Error Correction”. In: *Physical Review Letters* 111.9 (Aug. 2013). ISSN: 1079-7114 (cit. on p. 55).
- [PK22a] Pavel Panteleev and Gleb Kalachev. “Asymptotically good quantum and locally testable classical LDPC codes”. In: *Proceedings of the 54th annual ACM SIGACT symposium on theory of computing*. 2022, pp. 375–388 (cit. on p. 74).
- [PK22b] Pavel Panteleev and Gleb Kalachev. “Quantum LDPC Codes With Almost Linear Minimum Distance”. In: *IEEE Transactions on Information Theory* 68.1 (2022), pp. 213–229 (cit. on p. 74).
- [PKP23] Christopher A Pattison, Anirudh Krishna, and John Preskill. “Hierarchical memories: Simulating quantum LDPC codes with local gates”. In: *arXiv preprint arXiv:2303.04798* (2023) (cit. on pp. 5, 60).
- [Sch+03] Alexander Schrijver et al. *Combinatorial optimization: polyhedra and efficiency*. Vol. 24. 2. Springer, 2003 (cit. on p. 51).
- [SPST25] Marc Serra-Peralta, Mackenzie H Shaw, and Barbara M Terhal. “Decoding across transversal Clifford gates in the surface code”. In: *arXiv preprint arXiv:2505.13599* (2025) (cit. on p. 6).
- [Sho96] P.W. Shor. “Fault-tolerant quantum computation”. In: *Proceedings of 37th Conference on Foundations of Computer Science*. 1996, pp. 56–65 (cit. on p. 1).
- [Sko+23] Luka Skoric, Dan E Browne, Kenton M Barnes, Neil I Gillespie, and Earl T Campbell. “Parallel window decoding enables scalable fault tolerant quantum computation”. In: *Nature Communications* 14.1 (2023), p. 7040 (cit. on p. 5).
- [Ste96] Andrew Steane. “Multiple-particle interference and quantum error correction”. In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 452.1954 (1996), pp. 2551–2577 (cit. on p. 55).
- [SBB23] Armands Strikis, Simon C Benjamin, and Benjamin J Brown. “Quantum computing is scalable on a planar array of qubits with fabrication defects”. In: *Physical Review Applied* 19.6 (2023), p. 064081 (cit. on p. 65).
- [SJOY24] Esha Swaroop, Tomas Jochym-O’Connor, and Theodore J Yoder. “Universal adapters between quantum LDPC codes”. In: *arXiv preprint arXiv:2410.03628* (2024) (cit. on p. 55).
- [TY25] Yugo Takada and Hayata Yamasaki. “Doubly-polylog-time-overhead fault-tolerant quantum computation by a polylog-time parallel minimum-weight perfect matching decoder”. In: *arXiv preprint arXiv:2503.13601* (2025) (cit. on p. 5).
- [TKY24] Shiro Tamiya, Masato Koashi, and Hayata Yamasaki. “Polylog-time-and constant-space-overhead fault-tolerant quantum computation with quantum low-density parity-check codes”. In: *arXiv preprint arXiv:2411.03683* (2024) (cit. on pp. 3, 4, 74).
- [Tan+23] Xinyu Tan, Fang Zhang, Rui Chao, Yaoyun Shi, and Jianxin Chen. “Scalable surface-code decoders with parallelization in time”. In: *PRX Quantum* 4.4 (2023), p. 040344 (cit. on p. 5).
- [Vui+19] Christophe Vuillot, Lingling Lao, Ben Criger, Carmen García Almudéver, Koen Bertels, and Barbara M Terhal. “Code deformation and lattice surgery are gauge fixing”. In: *New Journal of Physics* 21.3 (Mar. 2019), p. 033028. ISSN: 1367-2630 (cit. on p. 55).

- [WY24] Dominic J Williamson and Theodore J Yoder. “Low-overhead fault-tolerant quantum computation by gauging logical operators”. In: *arXiv preprint arXiv:2410.02213* (2024) (cit. on p. 55).
- [WHY24] Adam Wills, Min-Hsiu Hsieh, and Hayata Yamasaki. “Constant-overhead magic state distillation”. In: *arXiv preprint arXiv:2408.07764* (2024) (cit. on p. 71).
- [Yod+25] Theodore J Yoder, Eddie Schoute, Patrick Rall, Emily Pritchett, Jay M Gambetta, Andrew W Cross, Malcolm Carroll, and Michael E Beverland. “Tour de gross: A modular quantum computer based on bivariate bicycle codes”. In: *arXiv preprint arXiv:2506.03094* (2025) (cit. on p. 55).
- [ZL24] Guo Zhang and Ying Li. “Time-efficient logical operations on quantum LDPC codes”. In: *arXiv preprint arXiv:2408.01339* (2024) (cit. on p. 55).
- [Zho+24] Hengyun Zhou, Chen Zhao, Madelyn Cain, Dolev Bluvstein, Casey Duckering, Hong-Ye Hu, Sheng-Tao Wang, Aleksander Kubica, and Mikhail D Lukin. “Algorithmic fault tolerance for fast quantum computing”. In: *arXiv preprint arXiv:2406.17653* (2024) (cit. on pp. 6, 56).