Fault-Tolerant Quantum Computing in the Pauli or Clifford Frame with Slow Error Diagnostics

Christopher Chamberland,[1,*] Pavithran Iyer,[2,†] and David Poulin[2,‡]

[1] *Institute for Quantum Computing and Department of Physics and Astronomy, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada*

[2] *Département de Physique and Institut Quantique, Université de Sherbrooke, Sherbrooke, Québec, J1K 2R1 Canada*

We consider the problem of fault-tolerant quantum computation in the presence of slow error diagnostics, either caused by slow measurement readouts or slow decoding algorithms. Our scheme offers a few improvements over previously existing solutions, for instance it does not require active error correction and results in a reduced error-correction overhead when error diagnostics is much slower than the gate time. In addition, we adapt our protocol to cases where the underlying error correction strategy chooses the optimal correction amongst all Clifford gates instead of the usual Pauli gates. The resulting Clifford frame protocol is of independent interest as it can increase error thresholds and could find applications in other areas of quantum computation.

PACS numbers: 03.67.Pp

I. INTRODUCTION AND BACKGROUND

In fault-tolerant quantum computation, syndrome measurements are used to detect and diagnose errors. Once diagnosed, an error can be corrected before it propagates through the rest of the computation. In this article, we are concerned with the impact of slow error diagnostics on fault-tolerance schemes. There are two origins for this concern. First, in certain solid-state and ion-trap qubit architectures, measurement times can be between 10 to 1000 times slower than gates times [1–5]. Thus, on the natural operating time-scale of the quantum computer, there is a long delay between an error event and its detection. Second, processing the measurement data to diagnose an error—i.e., decoding—can be computationally demanding. Thus, there can be an additional delay between the data acquisition and the error identification. At first glance, these delays might cause the error probability to build up between logical gates, thus effectively decreasing the fault-tolerance threshold. But as we will see, this is not necessarily the case.

One of the key tricks to cope with slow error diagnostics is the use of error frames [6]. While the basic idea of error correction is to diagnose and correct errors, it can often be more efficient to keep track of the correction in classical software instead of performing active error correction. In particular, this saves us from performing additional gates on the system, thus removing potential sources of errors. In most quantum error-correction schemes, the correction consists of a Pauli operator, i.e., tensor products of the identity I and the three Pauli matrices X, Y, and Z. Thus, at any given time, the computation is operated in a random but known *Pauli frame*: its state at time t is $P|\psi(t)\rangle$ for some Pauli operator P and where $|\psi(t)\rangle$ denotes the ideal state at time t. When error-diagnostics is slow, the system will unavoidably evolve in an unknown error frame for some time.

The problem of slow measurements in solid-state systems was addressed by DiVincenzo and Aliferis [7] in the context of concatenated codes. In addition to error diagnostics, concatenated schemes require measurements to prepare certain ancilla states used to fault-tolerantly extract the error syndrome and to inject magic states to complete a universal gate set. DiVincenzo and Aliferis' scheme hinges on the fact that the logical gate rate decreases exponentially with the number of concatenation levels; thus, at a sufficiently high level, measurement and gate times become commensurate. To concretely realize this simple observation, they combine a number of known and new techniques including ancillla correction, high-level state injection, and Pauli frames.

One limitation of this solution is that it only applies to concatenated codes realizing a universal gate set through magic state injection. This leaves out for instance surface codes [8] or concatenated codes with alternate universal gate constructions [9–14]. In particular, they inject noisy magic-states directly at high concatenation levels, thus loosing the benefit of low-level correction. In addition, when decoding times are very slow, this solution could wastefully use additional layers of concatenation with the sole purpose of slowing down the logical gates. Another drawback of their scheme is that it requires active error correction to ensure that high-level corrections are always trivial.

In this article, we introduce an alternative scheme to cope with slow error diagnostics which applies broadly to all codes and universal gate construction. Like the DiVincenzo and Aliferis scheme, the key idea will be to slow-down the logical gate rate to learn the error frame before it propagates to the rest of the computation. This is simply achieved by spacing out gates in the logical circuit and thus circumvents the unnecessary additional

* c6chambe@uwaterloo.ca
† pavithran.iyer.sridharan@usherbrooke.ca
‡ david.poulin@usherbrooke.ca

qubit overhead associated to extra concatenation layers. Our scheme does not require active error correction, it is entirely realized in an error frame. In addition, our scheme is compatible with a more general form of error correction which uses *Clifford frames*, where at any given time t, the state of the computer is $C|\psi(t)\rangle$ where C is a tensor-product of single-qubit Clifford group elements (defined below).

While the DiVincenzo and Aliferis scheme was motivated by slow measurements, most of it applies directly to the case of slow decoding. Our scheme too is oblivious to the origin of slow error diagnostics. We emphasize that slow decoding is a very serious concern. For instance, numerical simulations used to estimate the threshold or overhead of the surface code are computationally dominated by the decoding algorithm and require intense computational resources. Depending on the code distance, a single decoding cycle can take well above $1\mu s$ [15] on a standard processor, considerably slower than the natural GHz gate rate in the solid state.

Regarding slow measurements, it is important to distinguish two time scales. First, the time t_o it takes for the outcome of a measurement to become accessible to the *outside* world (e.g. to a field programmable gate array). For instance, t_o could be caused by various amplification stages of the measurement. Second, the measurement *repetition* time t_r, the minimum time between consecutive measurements of a given qubit. In principle, t_r can be made as small as the two-qubit gate time, provided that a large pool of fresh qubits are accessible. Indeed, it is possible to measure a qubit by performing a CNOT to an ancillary qubit initially prepared in the state $|0\rangle$ and then measuring this ancillary qubit. The ancillary qubit needs to wait a time t_o before it can be reset and used again, but other fresh qubits can be brought in to perform more measurements in the meantime. The scheme we present here and the one presented in [7] are designed to cope with large t_o, but both require small t_r. Indeed, the increase of the error threshold with t_r appears to be unavoidable.

Finally, to our knowledge, a theory of Clifford frames for fault-tolerant quantum computation has not been worked out before. By enabling them, our scheme offers a greater flexibility for error correction, thus potentially correcting previously non-correctable error models, or increasing the threshold of other noise models [16]. More generally, the possibility of fault-tolerantly computing in a Clifford frame opens up the door to new fault-tolerant protocols, e.g., using new codes or new hardware that have a different set of native fault-tolerant gates. For instance, Clifford frames were used as an accessory in measurement-based quantum computation with Majorana fermions [17]. Lastly, the possibility of computing in a Clifford frame could have applications to randomized compiling [18] which introduces random frames to de-correlate physical errors.

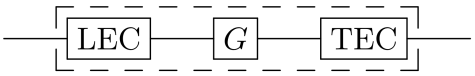

FIG. 1: Example of extended rectangle (exRec) for implementing the logical gate G. The leading and trailing EC circuits (LEC and TEC) perform fault-tolerant error correction for input errors and errors occurring during the implementation of G.

II. PAULI FRAME

Define $\mathcal{P}_n^{(1)}$ to be the n-qubit Pauli group (i.e. n-fold tensor products of the identity I and the three Pauli matrices X, Y, and Z). Then the *Clifford hierarchy* is defined by $\mathcal{P}_n^{(k)} = \{U : UPU^\dagger \in \mathcal{P}_n^{(k-1)} \quad \forall P \in \mathcal{P}_n^{(1)}\}$.

In physical implementations where measurement times are much longer than gate times or when error decoding is slow, if one were to perform active error correction, a large number of errors could potentially build up in memory during the readout times of the measurement. However, for circuits containing only Clifford gates, all Pauli recovery operators can be tracked in classical software without ever exiting the Pauli group. Indeed, suppose that at some given time during the computation, the state of the computer is in some Pauli frame defined by P—i.e., the state of the computer is $P|\psi\rangle$ where $|\psi\rangle$ is the ideal state. If we the apply a *Clifford gate* $U \in \mathcal{P}_n^{(2)}$, then the state will be $UP|\psi\rangle = P'U|\psi\rangle$ where P' is another Pauli operator. We thus see that the effect of U is to correctly transform the perfect state $|\psi\rangle$ and to change the Pauli frame P in some known way. Moreover, updating the Pauli frame $P' = UPU^\dagger$ can be done efficiently, with complexity $\mathcal{O}(n^2)$ [19]. This shows that as long as we only apply Clifford gates, there is no need to actively error-correct, we can instead efficiently keep track of the Pauli frame in classical software [6].

In concatenated codes for instance, error correction is performed in between the application of gates to ensure that errors don't accumulate in an uncontrolled fashion. A gate location in a quantum algorithm is thus replaced by an extended rectangle, where error correction is performed both before and after the application of the gate [20]. The leading error correction circuit in an extended rectangle is used to correct input errors that could have occurred prior to the application of the gate. The trailing error correction circuits correct errors that can occur during the application of the gate (see fig. 1). Each error correction circuit will multiply the current Pauli frame by a Pauli operator (the correction).

Since Clifford gates can be efficiently simulated on a classical computer, non-Clifford gates are needed for universal quantum computation. Universal quantum computation can be achieved for instance using gates from the Clifford group combined with the $T = \mathrm{diag}(1, e^{i\pi/4}) \in \mathcal{P}_1^3$ gate [21]. In general, Pauli operators will not remain in the Pauli-group when conjugated by

non-Clifford gates and so the Pauli frame cannot simply propagate through them.

Consider the application of a T gate in a quantum algorithm. Since measurement times are much longer than gate times, the Pauli frame right before the application of a T gate can only be known at a later time. Furthermore, by definition of the Clifford hierarchy, Pauli operators are mapped to Clifford operators under conjugation by $T \in \mathcal{P}_1^{(3)}$ gates. Therefore, the output correction after applying a logical T gate can be outside the Pauli frame.

In order to overcome these obstacles, we note that if error correction is performed immediately after the application of the T gate, the output correction can be written as a *logical* Clifford gate C times a Pauli matrix (a proof is presented in appendix A).

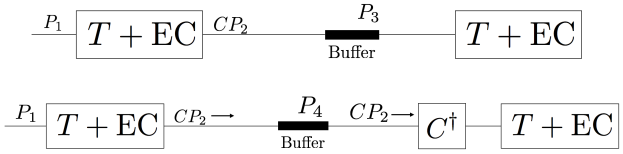

FIG. 2: Illustration of our scheme for propagating Pauli corrections through a T gate when error diagnostics are much longer than gate times. (TOP) When propagating the input Pauli P_1 through a T gate and performing error correction, the output can be written as CP_2 where C is a logical Clifford gate and P_2 is a Pauli matrix. A buffer is introduced to learn the Pauli frame immediately before applying the T gate which enables the logical Clifford correction C to be known. During the buffer, repeated rounds of error correction are performed to prevent the accumulation of errors for qubits waiting in memory. We denote the final Pauli correction arising from the EC rounds as P_3. (BOTTOM) We propagate the correction CP_2 through the buffer and apply a logical Clifford gate $C^\dagger$ in order to remove C thus restoring the Pauli frame. Although the propagation can map the buffer correction $P_3 \to P_4$, P_4 remains Pauli and can be known at a later time.

If we were to keep track of the logical Clifford correction C in classical software, it could propagate through the next T gate, resulting in a correction involving even more T gates. It could also propagate through a logical two-qubit gate (such as a CNOT) resulting in a two-qubit correction (the exact transformation rules are derived in section III). The two-qubit corrections could then propagate through other gates in the quantum algorithm leading to a generic Clifford correction. To prevent these scenarios from occurring, a buffer can be inserted right before the next logical two-qubit gate or T gate part of the quantum algorithm. The role of the buffer is to learn what the Pauli frame was right before the application of the previous logical T gate. During the buffer, repeated rounds of error correction are performed to prevent the accumulation of a large number of errors. There could be leftover Pauli corrections arising from error correction rounds which would only be known at a later time. However, by propagating the logical Clifford correction through the buffer, the Pauli corrections would remain Pauli.

Once the logical Clifford correction is propagated through the buffer, we apply a logical $C^\dagger$ thus restoring the Pauli frame. Our protocol guarantees that only Pauli corrections would be propagated through the next logical T or two-qubit gates. Our scheme is outlined in fig. 2. As in [7], our approach also effectively slows down gate times making them comparable to measurement times[1]. However, buffers are only introduced when necessary and without having to increase the size of the code.

III. CLIFFORD FRAME

In this section, we extend the protocol used to cope with slow error-diagnostics to the case where error corrections uses Clifford gates. When performing error correction on a set of encoded qubits with a stabilizer code, if one measures a non-trivial syndrome value l, a recovery map R_l is applied to the data block. The recovery map can always be written in the form $R_l = \mathcal{L}(l)\mathcal{T}(l)\mathcal{G}(l)$ where $\mathcal{G}(l) \in \mathcal{P}_n^{(1)}$ is a product of stabilizer generators, $\mathcal{L}(l)$ is a product of logical operators and $\mathcal{T}(l) \in \mathcal{P}_n^{(1)}$ is a product of pure errors [22], see also appendix A. Pure errors form an abelian group of operators that commute with all of the code's logical operators and all but one of the codes stabilizer generators. The logical operators $\mathcal{L}(l)$ are chosen to maximize the probability of recovering the correct codeword. The operators in the set $\mathcal{L}(l)$ are not necessarily restricted to logical Pauli operators as they can be extended to include all logical operators that can be applied fault-tolerantly.

It is thus natural to restrict $\mathcal{L}(l)$ to gates that can be applied transversally on the code. In particular, we will consider logical corrections in $(\mathcal{P}_1^{(2)})^{\otimes n}$, the group generated by n-fold tensor products of single-qubit Clifford operators. This latter group is generated by

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad \text{and} \quad S = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \qquad (1)$$

i.e., $\mathcal{P}_1^{(2)} = \langle H, S \rangle$. The order of the group is $|\mathcal{P}_1^{(2)}| = 24$ (ignoring global phases). For the n-qubit case, $\mathcal{P}_n^{(2)} = \langle H_i, S_i, \mathrm{CNOT}_{ij} \rangle$ where the indices i, j indicate which qubits to apply the Clifford gates. We point out that 2-D color codes [13] admit transversal realizations of all gates in $\mathcal{P}_1^{(2)}$. Furthermore, the 5-qubit code [23, 24] admits transversal realizations of logical gates in the set generated by $\langle SH, X, Z \rangle$.

[1] Note that the buffer increases the overall time for implementing a non-Clifford gate. However, this impacts the overall running time of the quantum algorithm only by a constant factor.

For concatenated codes, we restrict the discussion to the case where logical Clifford corrections are performed at the last concatenation level only. If Clifford corrections were performed at every level, one would need to wait for the measurement outcomes of every level and actively perform Clifford corrections. This is because a level-k logical Clifford correction does not generally commute with the level-$(k+1)$ syndrome measurements. The goal of defining a Clifford frame is to avoid actively correcting since the corrections themselves can introduce more errors into the computation.

We now derive the transformation rules for Clifford operators propagating through CNOT gates. Two-qubit controlled unitary gates $C\text{-}U_{12}|a\rangle|b\rangle = |a\rangle U^a|b\rangle$, where the first qubit is the control and the second qubit is the target, can be written as

$$C\text{-}U_{12} = \frac{1}{2}(I+Z)\otimes I + \frac{1}{2}(I-Z)\otimes U. \tag{2}$$

Note that from this definition $\text{CNOT}_{12} = C\text{-}X_{12}$. Using eq. (2), it is straightforward to show the following relations

$$(S\otimes I)C\text{-}X_{12} = C\text{-}X_{12}(S\otimes I), \tag{3}$$
$$(I\otimes S)C\text{-}X_{12} = C\text{-}Y_{12}(I\otimes S), \tag{4}$$
$$(I\otimes H)C\text{-}X_{12} = C\text{-}Z_{12}(I\otimes H), \tag{5}$$
$$(H\otimes I)C\text{-}X_{12} = U_X(H\otimes I), \tag{6}$$

where we defined $U_X \equiv \frac{1}{2}(I+X)\otimes I + \frac{1}{2}(I-X)\otimes X$.

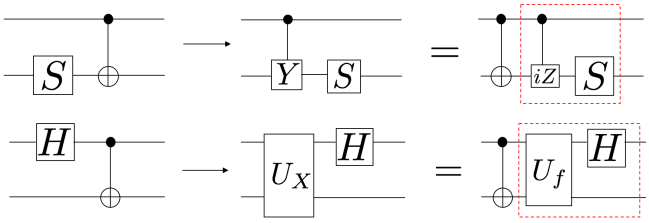

FIG. 3: (TOP) Propagation of $I\otimes S$ through a CNOT gate. (BOTTOM) Propagation of $H\otimes I$ through a CNOT gate where $U_f = \frac{1}{2}(I+iY)\otimes I + \frac{1}{2}(I-iY)\otimes X$. In both cases, if instead of correcting the Clifford errors prior to the CNOT gate we were to keep track of them using a Clifford frame, the corrections would involve two-qubit gates in addition to the original Clifford corrections.

From fig. 3 and eqs. (3) to (6), it can be seen that propagating Clifford corrections through CNOT gates can result in both single and two-qubit Clifford corrections. By keeping track of logical Clifford corrections in classical software, these could grow due to other CNOT gates resulting in a generic Clifford correction. Furthermore, when propagating Clifford corrections through non-Clifford gates (such as the T gate), the output can potentially be outside of the Clifford hierarchy. These could then propagate through the remainder of the logical circuit resulting in a unitary gate correction expressed as a product of several T gates.

A. Clifford propagation through CNOT gates

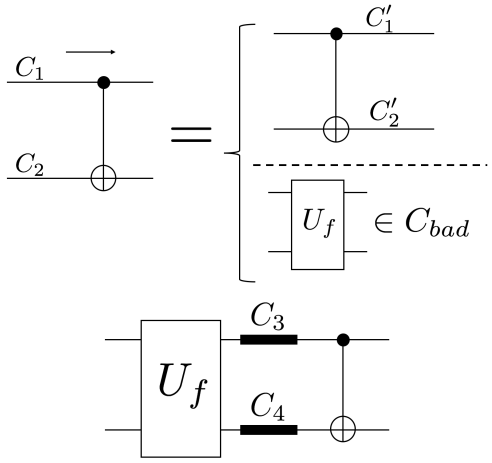

FIG. 4: (TOP) Propagating input Clifford gates $C_1\otimes C_2$ across a CNOT (part of a quantum algorithm) leads to two possible outcomes, one in C_{good} (defined in eq. (8)) and the other in C_{bad} (defined in eq. (9)). (BOTTOM) Buffers are introduced to learn if the outcome in (TOP) belongs to C_{good} or C_{bad}. Rounds of error correction in the buffers introduce the corrections $C_3\otimes C_4$. If the outcome in (TOP) belongs to C_{bad}, we apply a CNOT correction following the buffers. The protocol is repeated until the resulting gate belongs to C_{good}.

We first address the propagation of logical Clifford corrections (expressed in tensor product form) through CNOT gates. The goal is to prevent a two-qubit correction from spreading to multiple code blocks in order to avoid performing a generic Clifford correction. After the application of a logical CNOT gate (as part of a quantum algorithm), we can place a buffer before the next logical two-qubit gate (or non-Clifford gate) in order to determine what the Clifford frame was right before the application of the logical CNOT. Note that during the application of the buffer, repeated rounds of error correction are performed to prevent the build-up of a large number of errors, producing additional Clifford gates.

From eqs. (3) to (6), upon propagating the input Clifford gates $C_1\otimes C_2$ through a logical CNOT gate part of a quantum algorithm using a Clifford frame, two outcomes are possible. In the first case, we can have

$$C\text{-}X_{12}(C_1\otimes C_2) = (C'_1\otimes C'_2)C\text{-}X_{12}, \tag{7}$$

for some Clifford gates C'_1 and C'_2. In particular, we define

$$C_{good} = (C_1\otimes C_2)C\text{-}X_{12}. \tag{8}$$

If eq. (7) is not satisfied for $C_1\otimes C_2$, then the output will belong to the set C_{bad} defined to be

$$C_{bad} = \mathcal{P}_2^{(2)} \setminus C_{good}, \tag{9}$$

where $\mathcal{P}_2^{(2)}$ is the two-qubit Clifford group. Performing a computer search, we found that out of the $24^2 = 576$ possible input Clifford gates (expressed in tensor product form), 64 will satisfy eq. (7).

After the application of the CNOT gate part of the quantum algorithm, the buffers will introduce the Clifford corrections $C_3 \otimes C_4$ arising from the repeated rounds of error correction. Once the Clifford frame prior to applying the CNOT gate is known, we will be able to determine if the output from the propagation of the Clifford frame belongs to C_{good} or C_{bad}. If it belongs to C_{good}, then no further operations are required. In the case where it belongs to C_{bad}, we perform a logical CNOT correction after the buffer. A second set of buffers is introduced to determine if the resulting gate belongs to C_{good} or C_{bad}. We can repeat this process recursively until the resulting gate belongs to C_{good}.

Assuming the input Clifford gates and buffer Clifford corrections are chosen uniformly at random, we performed a simulation to estimate the transition probability from $C_{bad} \to C_{good}$. Performing 5×10^5 simulations, we found that the $C_{bad} \to C_{good}$ transition occurs with probability $\frac{1}{12}$. Thus, when computing in a random Clifford frame, each logical CNOT requires on average $12+1$ logical CNOTs.

Lastly, we point out that for noise models where the Clifford corrections $C_1 \otimes C_2$ arising from the buffer are biased towards the Pauli gates, it would be more advantageous to apply $C_1^\dagger \otimes C_2^\dagger$ before the following CNOT correction. More specifically, if the probability of acquiring a non-trivial Clifford correction in the buffer is ϵ, then the $C_{bad} \to C_{good}$ transition probability becomes $1 - \epsilon\frac{11}{12}$.

B. Clifford propagation through T gates

We now address the propagation of Clifford corrections through a logical T gate, which is a non-Clifford gate. When applying a logical T gate in a quantum algorithm, we could also place a buffer before the next logical gate part of the quantum algorithm to learn the Clifford frame immediately before applying the T gate. If the output correction is non-Clifford, we can perform appropriate corrections in order to restore the Clifford frame. If successful, this would guarantee that the input correction to the next gate would belong to the Clifford group.

Suppose the input to the logical T gate is a Clifford correction C_1, so the resulting gate is TC_1. On the one hand, if $TC_1 = \tilde{C}_1 T$ for some Clifford $\tilde{C}_1$—or equivalently $TCT^\dagger \in \mathcal{C}_1^{(2)}$—then no further operations are necessary. Only gates in the set generated by $C_1 \in \langle S, X\rangle$ satisfy this property. In fact, if $C_1 \in \langle S, X\rangle$, then $TC_1T^\dagger \in \langle S, X\rangle$. Thus, we define

$$C_- = \langle S, X\rangle, \quad \text{and} \quad C_+ = \mathcal{C}_1^{(2)} \setminus C_-. \tag{10}$$

Note that $|C_-| = 8$ while $|C_+| = 24 - 8 = 16$. Once we learn that the input Clifford correction C_1 to the logical T gate belongs to C_-, then no further operations are necessary to restore the Clifford frame. If the buffer Clifford operations are uniformly distributed over the Clifford group, this occurs with probability $\frac{1}{3}$.

On the other hand, when we learn that the input Clifford correction C_1 to the logical T gate belongs to C_+, we apply another logical T in the hope to restore the Clifford frame. Since an additional Clifford gate C_2 was accumulated during the buffer, the resulting gate is TC_2TC_1. When $C_2 \in C_-$, then $TC_2TC_1 = \tilde{C}_2T^2C_1 \in \mathcal{P}_1^{(2)}$, since $T^2 = S$ is a Clifford transformation. At this stage, we have returned to a Clifford frame, but have removed the desired logical T gate: we are thus back at our starting point and can try anew.

Once again, whenever the Clifford corrections occurring during the buffer are biased to the Pauli group (e.g., when the probability of an error is low), before applying another T gate correction to restore the algorithm T gate, we should apply the Clifford transformation $(\tilde{C}_2SC_1)^\dagger$. In this way, we would increase the probability of being in a state of the form CT where $C \in C_-$ (thus restoring the Clifford frame). To take this possibility into account, we will henceforth assume that the Clifford corrections arising from the buffer belong to C_- with probability $1-p$ and to C_+ with probability p.

Define $\mathcal{T}^{(0)} = \mathcal{P}_1^{(2)}$ to be the set of single-qubit Clifford gates, and define

$$\mathcal{T}^{(k)} = \prod_{j=1}^{k} (TC_j) \quad \text{where} \quad C_j \in C_+. \tag{11}$$

Every time we learn that the previous buffer Clifford was in C_+, we apply a T gate and wait for another buffer. Since this buffer will belong to C_- with probability $1-p$ and to C_+ with probability p, each step of the above protocol can be seen as a step in a random walk over the sets $\mathcal{T}^{(k)}$ with transition $\mathcal{T}^{(k)} \to \mathcal{T}^{(k+1)}$ occurring with probability p and transition $\mathcal{T}^{(k)} \to \mathcal{T}^{(k-1)}$ occurring with probability $1-p$. Every time $\mathcal{T}^{(0)}$ is reached, there is a probability $1-p$ that a logical T gate is successfully realized at the next step.

To summarize, when attempting to implement a logical T gate starting in a Clifford frame $\mathcal{T}^{(0)}$, we either succeed with probability $1-p$ or end up applying a $\mathcal{T}^{(1)}$ gate with probability p. In the latter case, we enter a random walk over $k \in \mathbb{N}$, and our goal it to return to $k = 0$. This clearly requires an odd number of steps. The one-step process $1 \to 0$ occurs with probability $1-p$. The three step process $1 \to 2 \to 1 \to 0$ occurs with probability $p(1-p)^2$. Generalizing to $t = 2j+1$ time steps, where $j \in \mathbb{N}$, the number of paths that lead to a gate in $\mathcal{T}^{(0)}$ is given by the j'th Catalan number $C_j = \frac{1}{j+1}\binom{2j}{j}$ [25]. Therefore, the probability of returning to a gate in $\mathcal{T}^{(0)}$ after $t = 2j+1$ time steps is given by

$$P_{2j+1} = \frac{1}{j+1}\binom{2j}{j} p^j (1-p)^{j+1}. \tag{12}$$

Using the generating function for the Catalan number $c(x) = \sum_{j\geq 0} C_j x^j = (1 - \sqrt{1-4x})/(2x)$, the total

probability that the random walk terminates is given by

$$\sum_{k\geq 0} P_{2k+1} = \min\left\{\frac{1-p}{p}, 1\right\}. \tag{13}$$

If $p > 1/2$, then with finite probability the random walk will not terminate whereas if $p \leq 1/2$, the random walk is guaranteed terminate. This means that we must choose Clifford gates from C_- with probability greater than $1/2$. The latter condition can be satisfied in cases where Clifford corrections arising from the buffer are biased towards gates belonging to C_-, or in particular if they are biased towards Pauli gates or the identity. When the buffers are chosen uniformly over the Clifford group, then $p = 8/24 = 1/3 < 1/2$ so with some finite probability the procedure will not terminate.

We conclude this section by calculating the probability of obtaining a gate in $\mathcal{T}^{(0)}$ within n time steps, which we define as $F(p, n)$. This quantity gives the number of expected T gate corrections that need to be applied in order to restore the Clifford frame after propagation through a logical T gate. The probability $F(p, n)$ is obtained by summing eq. (12) with a cut-off of n time-steps. The result is given by

$$\begin{aligned} F(p,n) &= \sum_{k=0}^{n} P_{2k+1} \\ &= 1 - f(p,n), \end{aligned} \tag{14}$$

where

$$\begin{aligned} f(p,n) = \frac{1-p}{2+n}\binom{2(n+1)}{n+1}(p(1-p))^{n+1}\times \\ \times\ {}_2F_1(1, \frac{3}{2}+n; 3+n; 4p(1-p)), \end{aligned} \tag{15}$$

and ${}_2F_1(a, b; c; z)$ is the Hypergeometric function defined in [26]. Plots of eq. (14) are given in fig. 5.

We now obtain an upper bound on the number of time steps n that are necessary to restore the Clifford frame with probability $q = 1 - \epsilon$. In other words, we would like to obtain an upper bound on n such that $F(p, n) > 1 - \epsilon$. We first obtain a lower bound for the function $f(p, n)$ by using the following inequalities

$$\binom{2(n+1)}{n+1} \geq \left(\frac{2(n+1)}{(n+1)}\right)^{n+1} = 2^{n+1}, \tag{16}$$

$${}_2F_1(1, \frac{3}{2}+n; 3+n; 4p(1-p)) \geq 2p+1. \tag{17}$$

Inserting eqs. (16) and (17) into eq. (14), we obtain

$$F(p,n) \leq 1 - \frac{(1-p)(2p+1)}{n+2}[2p(1-p)]^{n+1}. \tag{18}$$

Setting $F(p, n) > 1 - \epsilon$, we obtain

$$\frac{[2p(1-p)]^{n+1}}{n+2} \leq \frac{\epsilon}{(1-p)(2p+1)} \tag{19}$$

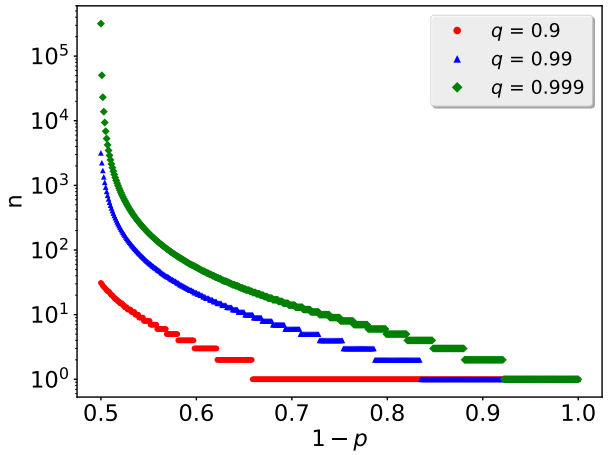

FIG. 5: For a fixed value of $1 - p$, we plot the value of n such that $F(p, n) > q$. We give plots for $q = 0.9$, $q = 0.99$ and $q = 0.999$. Hence, each curve corresponds to the expected number of T gate corrections that are required for obtaining a gate in $\mathcal{T}^{(0)}$ with probability greater than q.

For $n > 2$, we have $1/(n+2) > (\frac{1}{2})^{n+1}$, so eq. (20) becomes

$$[p(1-p)]^{n+1} \leq \frac{\epsilon}{(1-p)(2p+1)} \tag{20}$$

which shows that $n \sim |\log \epsilon|$—the T gate overhead for restoring the Clifford frame scales logarithmically in ϵ.

IV. CONCLUSION

In this paper, we considered performing fault-tolerant quantum computation when measurement times and/or decoding times are much slower than gate times.

This was realized by providing a new scheme for performing error correction using the Pauli frame by placing buffers after the application of a non-Clifford gate. We showed that the Pauli frame can always be restored by applying logical Clifford corrections prior to the application of the next two-qubit or non-Clifford gate part of a quantum algorithm.

Given that logical Clifford corrections can significantly increase a code's threshold value [16], in the remainder of the manuscript, we showed how to perform fault-tolerant error correction in the Clifford frame. We performed an in-depth study of the propagation of logical Clifford corrections through logical CNOT and T gates, and the same idea can be generalized to other universal gate sets.

For the propagation through CNOT gates, we placed buffers at strategic locations to ensure that the output Clifford corrections could be expressed in tensor product form. To achieve this, we found that on average 12 logical CNOT corrections would be required when Clifford

corrections arising from buffers are chosen uniformly at random. This ensures that two-qubit Clifford corrections would not propagate through the remainder of the circuit yielding a generic Clifford correction.

We also used buffers to keep track of the Clifford corrections propagating through T gates. We showed that in certain conditions, by applying enough T gate corrections, the Clifford frame could be restored with probability arbitrarily close to 1. Furthermore, to restore the Clifford frame with probability at least $1-\epsilon$, we showed that the number of T gate corrections scales as $|\log \epsilon|$.

V. ACKNOWLEDGEMENTS

We thank Tomas Jochym-O'Connor and Daniel Gottesman for useful discussions. C. C. would like to acknowledge the support of QEII-GSST and to thank the Institut Quantique at the University of Sherbrooke for their hospitality where most of this work was completed. This work was supported by the Army Research Office contract number W911NF-14-C-0048.

[1] J. R. Petta, A. C. Johnson, J. M. Taylor, E. A. Laird, A. Yacoby, M. D. Lukin, C. M. Marcus, M. P. Hanson, *et al.*, "Coherent manipulation of coupled electron spins in semiconductor quantum dots," *Science*, vol. 309, no. 5744, pp. 2180–2184, 2005.

[2] M. Veldhorst, J. C. C. Hwang, C. H. Yang, A. W. Leenstra, B. de Ronde, J. P. Dehollain, J. T. Muhonen, F. E. Hudson, K. M. Itoh, A. Morello, and A. S. Dzurak, "An addressable quantum dot qubit with fault-tolerant control-fidelity," *Nature nanotechnology*, vol. 9, no. 12, pp. 981–985, 2014.

[3] J. Stehlik, Y. Y. Liu, C. M. Quintana, C. Eichler, T. R. Hartke, and J. R. Petta, "Fast charge sensing of a cavity-coupled double quantum dot using a josephson parametric amplifier," *Physical Review Applied*, vol. 4, p. 014018, 2015.

[4] S. Olmschenk, D. Hayes, D. N. Matsukevich, P. Maunz, D. L. Moehring, K. C. Younge, and C. Monroe, "Measurement of the lifetime of the $6p^2\mathrm{P}^{\mathrm{o}}_{1/2}$ level of YB^+," *Phys. Rev. A*, vol. 80, no. 2, p. 022502, 2009.

[5] H. Häffner, C. Roos, and R. Blatt, "Quantum computing with trapped ions," *Physics Reports*, vol. 469, no. 4, pp. 155 – 203, 2008.

[6] E. Knill, "Quantum computing with realistically noisy devices," *Nature*, vol. 434, no. 7029, pp. 39–44, 2005.

[7] D. P. DiVincenzo and P. Aliferis, "Effective fault-tolerant quantum computation with slow measurements," *Phys. Rev. Lett.*, vol. 98, p. 020501, 2007.

[8] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Phys. Rev. A*, vol. 86, p. 032324, 2012.

[9] E. Knill, R. Laflamme, and W. H. Zurek, "Threshold accuracy for quantum computation," *arXiv: quant-ph/9610011*, 1996.

[10] T. Jochym-O'Connor and R. Laflamme, "Using concatenated quantum codes for universal fault-tolerant quantum gates," *Phys. Rev. Lett.*, vol. 112, p. 010505, 2014.

[11] A. Paetznick and B. W. Reichardt, "Universal fault-tolerant quantum computation with only transversal gates and error correction," *Phys. Rev. Lett.*, vol. 111, p. 090505, 2013.

[12] J. T. Anderson, G. Duclos-Cianci, and D. Poulin, "Fault-tolerant conversion between the steane and reed-muller quantum codes," *Phys. Rev. Lett.*, vol. 113, p. 080501, 2014.

[13] H. Bombín, "Gauge color codes: optimal transversal gates and gauge fixing in topological stabilizer codes," *New J. Phys.*, vol. 18, p. 043038, 2016.

[14] T. J. Yoder, R. Takagi, and I. L. Chuang, "Universal fault-tolerant gates on concatenated stabilizer codes," *Phys. Rev. X.*, vol. 6, p. 031039, 2016.

[15] S. Devitt, A. Fowler, T. Tilma, W. Munro, and K. Nemoto, "Classical processing requirements for a topological quantum computing system," *Int. J. Quant. Inf.*, vol. 8, pp. 1–27, 2010.

[16] C. Chamberland, J. J. Wallman, S. Beale, and R. Laflamme, "Hard decoding algorithm for optimizing thresholds under general markovian noise," *Phys. Rev. A*, vol. 95, p. 042332, 2017.

[17] T. Karzig, C. Knapp, R. Lutchyn, P. Bonderson, M. Hastings, C. Nayak, J. Alicea, K. Flensberg, S. Plugge, Y. Oreg, C. Marcus, and M. H. Freedman, "Scalable designs for quasiparticle-poisoning-protected topological quantum computation with majorana zero modes," *arXiv:1610.05289*, 2016.

[18] J. J. Wallman and J. Emerson, "Noise tailoring for scalable quantum computation via randomized compiling," *Physical Review A*, vol. 94, no. 5, p. 052325, 2016.

[19] D. Gottesman, "The heisenberg representation of quantum computers, talk at," in *International Conference on Group Theoretic Methods in Physics*, Citeseer, 1998.

[20] P. Aliferis, D. Gottesman, and J. Preskill, "Quantum accuracy threshold for concatenated distance–3 codes," *Quant. Inf. Comput.*, vol. 6, pp. 97–165, 2006.

[21] P. O. Boykin, T. Mor, M. Pulver, V. Roychowdhury, and F. Vatan, "On universal and fault-tolerant quantum computing: A novel basis and a new constructive proof of universality for shor's basis," in *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pp. 486–494, IEEE, 1999.

[22] D. Poulin, "Optimal and efficient decoding of concatenated quantum block codes," *Phys. Rev. A.*, vol. 74, p. 052333, 2006.

[23] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, "Mixed state entanglement and quantum error correction," *Phys. Rev. A*, no. 54, p. 3824, 1996.

[24] R. Laflamme, C. Miquel, J. P. Paz, and W. H. Zurek, "Perfect quantum error correcting code," *Phys. Rev. Lett.*, vol. 77, no. 198, 1996.

[25] T. Koshy, *Catalan Numbers with Applications*. Oxford University Press, Oxford, England, 2008.

[26] G. E. Andrews, R. Askey, and R. Roy, "Special functions (encyclopedia of mathematics and its applications vol 71)," 1999.

[27] B. Rahn, A. C. Doherty, and H. Mabuchi, "Exact performance of concatenated quantum codes," *Phys. Rev. A.*, vol. 66, p. 032304, 2002.

[28] A. Jamiołkowski, "Linear transformations which preserve trace and positive semidefiniteness of operators," *Reports on Mathematical Physics*, vol. 3, no. 4, pp. 275 – 278, 1972.

[29] C. J. Wood, J. D. Biamonte, and D. G. Cory, "Tensor networks and graphical calculus for open quantum systems.," *Quantum Information & Computation*, vol. 15, no. 9&10, pp. 759–811, 2015.

Appendix A: Error correction for input Clifford errors

In this appendix we will explain why a generic $n-$qubit Clifford error on an encoded state transforms into a logical Clifford operation and a physical Pauli operation, upon doing stabilizer measurements.

Before we delve into the reasoning, some definitions first. Let $\bar{\rho}$ be an encoded state of a stabilizer code with stabilizer $\mathcal{S}$, that undergoes an error described by some CPTP map $\mathcal{E}$, i.e, $\bar{\rho} \mapsto \mathcal{E}(\bar{\rho})$. Let Π_0 denote the projector onto the code space and consequently Π_s denote the projector onto the syndrome space of s. Let T_s be a Pauli operation that takes a state from the syndrome s subspace to the code space, i.e, $\Pi_s = T_s \cdot \Pi_0 \cdot T_s$. Upon measuring the stabilizer generators to obtain a syndrome s and applying the corresponding T_s operation, the noisy state can be projected back to the code space,

$$\mathcal{E}(\bar{\rho}) \mapsto T_s \Pi_s \, \mathcal{E}(\bar{\rho}) \, \Pi_s T_s \, . \tag{A1}$$

Since the resulting state is in the codespace, it can be decoded back to a single qubit state ϕ, given by

$$\phi = \sum_a \mathrm{Tr}\left(T_s \Pi_s \, \mathcal{E}(\bar{\rho}) \, \Pi_s T_s \cdot \bar{P}_a\right) P_a \tag{A2}$$

Let us denote the combined effect of the encoder, noise model, the projection to code space and the decoder by a quantum channel called the *effective projection*. The effective projection can be seen as acting directly on ρ to result in ϕ [27].

However, in order to extract the (single qubit or logical) CPTP map defining the effective projection channel, we must make use of another tool, an isomorphism between channels and states, called the Choi-Jamiłowski isomorphism [28, 29]. Under this isomorphism, a single qubit CPTP channel $\mathcal{E}$ is mapped to a unique bipartite quantum state $\mathcal{J}(\mathcal{E})$ called the *Choi matrix* corresponding to $\mathcal{E}$, in the following manner.

$$\mathcal{J}(\mathcal{E}) = \frac{1}{4} \sum_{i,j} \mathcal{E}(P_i) \otimes P_j^T \tag{A3}$$

Furthermore, when the quantum channel is expressed as a process matrix Λ, where $\Lambda_{i,j} = \mathrm{Tr}\left(\mathcal{E}(P_i) \cdot P_j\right)$, where $P_i, P_j \in \{I, X, Y, Z\}$ are Pauli matrices, it follows that

$$\Lambda_{ij} = \mathrm{Tr}\left(\mathcal{J}(\mathcal{E}) \cdot (P_j \otimes P_i^T)\right) \, . \tag{A4}$$

Lastly, note that when $\mathcal{E}(\rho) = C \cdot \rho \cdot C^\dagger$ where C is a Clifford operation, the corresponding process matrix $\Lambda(\mathcal{E})$ is given by

$$\begin{aligned} \Lambda(\mathcal{E})_{ij} &= \mathrm{Tr}\left(C \cdot P_i \cdot C^\dagger \cdot P_j\right) \\ &= \mathrm{Tr}\left(P_{\sigma(i)} \cdot P_j\right) = \delta_{\sigma(i),j} \end{aligned} \tag{A5}$$

where σ is a permutation of $\{1, 2, 3, 4\}$ that depends on the Clifford operation. Hence the process matrix of a Clifford channel is a permutation matrix.

We now have all the necessary ingredients to prove our claim. Note that if ρ in Eq. A1 is the Bell state and $\mathcal{E}$ acts on one half of the encoded Bell state, then using Eq. A3, we know that ϕ in Eq. A2 is simply the Choi matrix corresponding to the effective projection channel. From Eq. A4, it follows that the process matrix for the effective projection channel, denoted by $\Lambda^{(1)}$, is given by

$$\begin{aligned} \Lambda^{(1)}_{ij} &= \mathrm{Tr}\left(T_s \Pi_s \, \mathcal{E}(\Pi_0 \cdot \bar{P}_i) \, \Pi_s T_s \Pi_0 \bar{P}_j\right) \\ &= \mathrm{Tr}\left(\mathcal{E}(\Pi_0 \cdot \bar{P}_i) \cdot \Pi_s \cdot \bar{P}_j\right) \\ &= \sum_{\substack{l \\ P_l \in \bar{P}_i \cdot \mathcal{S}}} \sum_{\substack{k \\ P_k \in \bar{P}_j \cdot \mathcal{S}}} c_k \Lambda_{lk} \\ &= \sum_{\substack{l \\ P_l \in \bar{P}_i \cdot \mathcal{S}}} \sum_{\substack{k \\ P_k \in \bar{P}_j \cdot \mathcal{S}}} c_k \delta_{l,\sigma(k)} \end{aligned} \tag{A6}$$

where $c_k \in \{+1, -1\}$ and in the last step, we have used the fact that Λ is the process matrix of a Clifford operation, Eq. A5. That $\Lambda^{(1)}$ is also a permutation matrix follows from two properties – (i) Every row of $\Lambda^{(1)}$ has a unique non-zero column and (ii) every column of $\Lambda^{(1)}$ has a unique non-zero row. We will show (i) explicitly in what follows, the proof for (ii) is almost identical. Suppose that there are two columns j' and j'' such that $\Lambda_{ij'} \neq 0$ and $\Lambda_{ij''} \neq 0$. Along with Eq. A6, it implies that there exists $P_{k'} \in \bar{P}_{j'} \cdot \mathcal{S}$ and $P_{k''} \in \bar{P}_{j''} \cdot \mathcal{S}$ such that $\delta_{l,\sigma(k')} = \delta_{l,\sigma(k'')}$. Hence, it must be that $\sigma(k') = \sigma(k'')$, in other words, $j' = j''$.

Hence the effective projection channel is indeed a Clifford operation, in other words, any $n-$qubit Clifford operation on the physical qubits can be promoted to a logical Clifford operation and a physical Pauli error (T_s in Eq. A1), by a syndrome measurement.