

Thresholds for the distributed surface code in the presence of memory decoherence

de Bone, Sébastien; Möller, Paul; Bradley, Conor E.; Taminiau, Tim H.; Elkouss, David

DOI

[10.1116/5.0200190](https://doi.org/10.1116/5.0200190)

Publication date

2024

Document Version

Final published version

Published in

AVS Quantum Science

Citation (APA)

de Bone, S., Möller, P., Bradley, C. E., Taminiau, T. H., & Elkouss, D. (2024). Thresholds for the distributed surface code in the presence of memory decoherence. *AVS Quantum Science*, 6(3), Article 033801. <https://doi.org/10.1116/5.0200190>

Important note

To cite this publication, please use the final published version (if applicable). Please check the document version above.

Copyright





Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy

Please contact us and provide details if you believe this document breaches copyrights. We will remove access to the work immediately and investigate your claim.

RESEARCH ARTICLE | JULY 01 2024

Thresholds for the distributed surface code in the presence of memory decoherence

Sébastien de Bone  ; Paul Möller  ; Conor E. Bradley  ; Tim H. Taminiau; David Elkouss 



AVS Quantum Sci. 6, 033801 (2024)

<https://doi.org/10.1116/5.0200190>



Thresholds for the distributed surface code in the presence of memory decoherence

Cite as: AVS Quantum Sci. 6, 033801 (2024); doi: 10.1116/5.0200190

Submitted: 25 January 2024 · Accepted: 17 May 2024 ·

Published Online: 1 July 2024



View Online



Export Citation



CrossMark

Sébastien de Bone,^{1,2}  Paul Möller,¹  Conor E. Bradley,^{1,3}  Tim H. Taminiau,¹ and David Elkouss^{1,4,a)} 

AFFILIATIONS

¹QuTech, Delft University of Technology, Lorentzweg 1, 2628 CJ Delft, The Netherlands

²QuSoft, CWI, Science Park 123, 1098 XG Amsterdam, The Netherlands

³Pritzker School of Molecular Engineering, University of Chicago, Chicago, Illinois 60637, USA

⁴Networked Quantum Devices Unit, Okinawa Institute of Science and Technology Graduate University, Okinawa, Japan

^{a)}Electronic mail: david.elkouss@oist.jp

ABSTRACT

In the search for scalable, fault-tolerant quantum computing, distributed quantum computers are promising candidates. These systems can be realized in large-scale quantum networks or condensed onto a single chip with closely situated nodes. We present a framework for numerical simulations of a memory channel using the distributed toric surface code, where each data qubit of the code is part of a separate node, and the error-detection performance depends on the quality of four-qubit Greenberger–Horne–Zeilinger (GHZ) states generated between the nodes. We quantitatively investigate the effect of memory decoherence and evaluate the advantage of GHZ creation protocols tailored to the level of decoherence. We do this by applying our framework for the particular case of color centers in diamond, employing models developed from experimental characterization of nitrogen-vacancy centers. For diamond color centers, coherence times during entanglement generation are orders of magnitude lower than coherence times of idling qubits. These coherence times represent a limiting factor for applications, but previous surface code simulations did not treat them as such. Introducing limiting coherence times as a prominent noise factor makes it imperative to integrate realistic operation times into simulations and incorporate strategies for operation scheduling. Our model predicts error probability thresholds for gate and measurement reduced by at least a factor of three compared to prior work with more idealized noise models. We also find a threshold of 4×10^2 in the ratio between the entanglement generation and the decoherence rates, setting a benchmark for experimental progress.

© 2024 Author(s). All article content, except where otherwise noted, is licensed under a Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>). <https://doi.org/10.1116/5.0200190>

I. INTRODUCTION

A distributed quantum computer^{1,2} realizes a large-scale processing system by using entanglement to link smaller quantum processing units. For example, the sub-units may be elements of a photonic chip or form the nodes of a quantum network on a larger scale.³ Fault tolerance is naturally achieved by establishing the connectivity according to the architecture of a topological error-correction code. The distributed approach provides advantages in terms of scalability but is limited by the availability of high-quality entanglement. This is because entangled states are required for both inter-node operations and for detecting local errors with the error-correction code.^{4–9}

In this paper, we focus on systems that are capable of generating remote two-qubit entanglement between pairs of connected nodes. There exist several physical systems suitable for generating this type of entanglement with optical interfaces.¹⁰ Examples of this are ion traps,

neutral atoms, and color centers in the diamond lattice, such as nitrogen-vacancy (NV),^{11–22} silicon-vacancy (SiV),^{23–27} or tin-vacancy (SnV)^{28–30} centers. As a concrete example, we investigate the distributed surface code with hardware-specific noise based on color centers, also known as defect centers. We emphasize that the obtained insights are more general and that our simulation tools allow for implementing error models based on general hardware implementations.

Diamond color centers host long-lived electron spins that exhibit coherent optical transitions, enabling their use as a communication qubit. This qubit is used to create entanglement with other nodes and can address up to tens of proximal nuclear spins and or other electron spins occurring in the host material.^{15,19,27} These nuclear spins can be used as local processing qubits—i.e., as a local memory to store and manipulate quantum states.¹⁶ Hereafter, in the context of diamond color centers, the term “memory qubits” specifically refers to such spins.

Physical systems suitable for distributed quantum computing can be operated as *fault-tolerant* quantum computers by employing a subset of their memory qubits as data qubits of an *error-correction code*, such as the toric surface code.^{31,32} The principle behind error-correction codes is that many physical qubits hold a smaller number of *logical* states, and unwanted errors can be detected and corrected by measuring the *stabilizer operators* of the code. For such a system, fault-tolerance goes hand-in-hand with the existence of *thresholds* for local sources of error: if one manages to keep the error sources below their threshold values, one can make the logical error rate arbitrarily small by increasing the dimension of the error-correction code.

The toric code has a depolarizing *phenomenological* error probability threshold of approximately 10% to 11%.³³ This error model assumes that all qubits of the code are part of the same quantum system, stabilizer measurements can be carried out perfectly, and the qubits experience depolarizing noise in between the stabilizer measurement rounds. A more precise analysis with a *circuit-level* error model yields error probability thresholds between 0.9% and 0.95%.³⁴ In this model, the stabilizer measurement circuit is simulated with noisy gates and measurements, and it is implicitly assumed that the connectivity of the system allows direct two-qubit gates between adjacent qubits of the code topology. Therefore, this error model corresponds to a *monolithic* architecture.

If one wants to implement the toric code in a network setting, where every data qubit of the code is part of a separate network node, the stabilizer operators can be measured with the aid of four-qubit Greenberger–Horne–Zeilinger (GHZ) states. These GHZ states can be created by *fusing* three or more Bell pairs created between the involved nodes. Nickerson *et al.* analyzed the distributed toric code in this setting.^{34,35} They included protocols with a relatively large number of *entanglement distillation* steps that create high-quality GHZ states from imperfect Bell pairs. They found³⁴ thresholds for the local qubit operations between 0.6% and 0.82%—i.e., slightly below the monolithic thresholds. In their threshold calculations, Nickerson *et al.* do not explicitly consider circuit operation times and do not include qubit memory *decoherence* during entanglement creation—i.e., the notion that the quality of the code’s data qubits decays over time. However, in current physical systems of interest, decoherence during entanglement creation typically constitutes a large source of error. For state-of-the-art NV centers, coherence times during optical Bell pair generation are one to two orders of magnitude lower than estimated by Nickerson *et al.*^{36–38} The influence of this decoherence is further increased by the reality that success probabilities per optical Bell pair generation attempt currently fall significantly short of unity.^{15,39}

Therefore, next to the errors in operations and in entangled states considered in Refs. 34 and 35, decoherence of quantum states over time emerges as the third primary source of noise for accurate assessment of distributed quantum computing systems. The influence of memory qubit decoherence during entanglement creation can be captured with the *link efficiency* η_{link}^* .²¹ This parameter quantifies the average number of entangled pairs that can be generated within the coherence times.

To investigate the influence of the coherence times, we develop a time-tracking simulator and implement realistic operation durations. Additionally, considering the pivotal role of the operation order in this new scenario, we formulate a strategy for scheduling operations. We

find that, with realistic operation and coherence times, the thresholds with the GHZ generation protocols of Refs. 34 and 35 disappear. We investigate the quantitative impact of memory decoherence and optimize over GHZ generation protocols with less distillation that can overcome this. For a range of different coherence times during entanglement generation, we find two-qubit gate error and measurement error probability thresholds for diamond color centers up to 0.24%. We find that fault-tolerance is reachable with $\eta_{\text{link}}^* \approx 4 \times 10^2$. This improves on the prior results of $\eta_{\text{link}}^* = 2 \times 10^5$ for the idealized time scale estimates of Nickerson *et al.*³⁴ However, this link efficiency is still above the state-of-the-art hardware²¹ reaching up to $\eta_{\text{link}}^* \approx 10$.

In the remainder of this paper, Sec. II describes GHZ creation and distillation protocols necessary for the distributed surface code. Consequently, in Sec. III, we present the full cycle of stabilizer measurements of the surface code. In Sec. IV, we describe error models that allow us to investigate a specific hardware implementation in the distributed surface code setting: diamond color centers. Finally, in Sec. V, we investigate the parameter regimes necessary for fault tolerance with these error models.

II. GHZ GENERATION PROTOCOLS

As mentioned in Sec. I, the stabilizer operators of a distributed quantum error-correcting code can be measured by consuming GHZ states. In the following, we discuss protocols that create GHZ states by combining Bell pairs. For each GHZ protocol, we identify two parameters. The first one is the minimum number of Bell pairs k required to create the GHZ state. This number indicates the amount of distillation taking place in the protocol. The second one is the maximum number of qubits per node q necessary to generate the GHZ state. We summarize prior work in Sec. II A. In Sec. II B, we discuss our method for generating GHZ protocols.

A. Prior GHZ protocols

There is a plethora of prior work considering the generation and purification of GHZ states.^{40–53} Here, we focus on protocols that combine Bell pairs into a four-qubit GHZ state and discuss seven of them.

First, we consider two protocols that we used in an earlier study:²¹ the Plain ($k=3, q=2$) and Modicum ($k=4, q=2$) protocols. These protocols were designed to create a GHZ state with no distillation or only a single distillation step. The Plain protocol is the simplest protocol for creating a GHZ state from Bell pairs; it fuses three Bell pairs into a four-qubit GHZ state without any distillation. The Modicum protocol uses a fourth Bell pair to perform one round of distillation on the GHZ state.

On top of that, we consider five GHZ protocols found by Nickerson *et al.* in the context of distributed implementations of the toric code: Expedient ($k=22, q=3$) and Stringent ($k=42, q=3$) from Ref. 34, and Basic ($k=8, q=3$), Medium ($k=16, q=4$), and Refined ($k=40, q=5$) from Ref. 35.

B. Dynamic program to generate GHZ protocols

In this section, we present a method for optimizing GHZ creation with realistic noise models. We focus on creating GHZ states of the form $|\text{GHZ}_n\rangle = (|0\rangle^{\otimes n} + |1\rangle^{\otimes n})/\sqrt{2}$, where $n \in \{2, 3, 4\}$ represents the number of parties. We call n the weight of the GHZ state. For convenience, we use the notation $|\text{GHZ}_2\rangle$ to describe a Bell state.

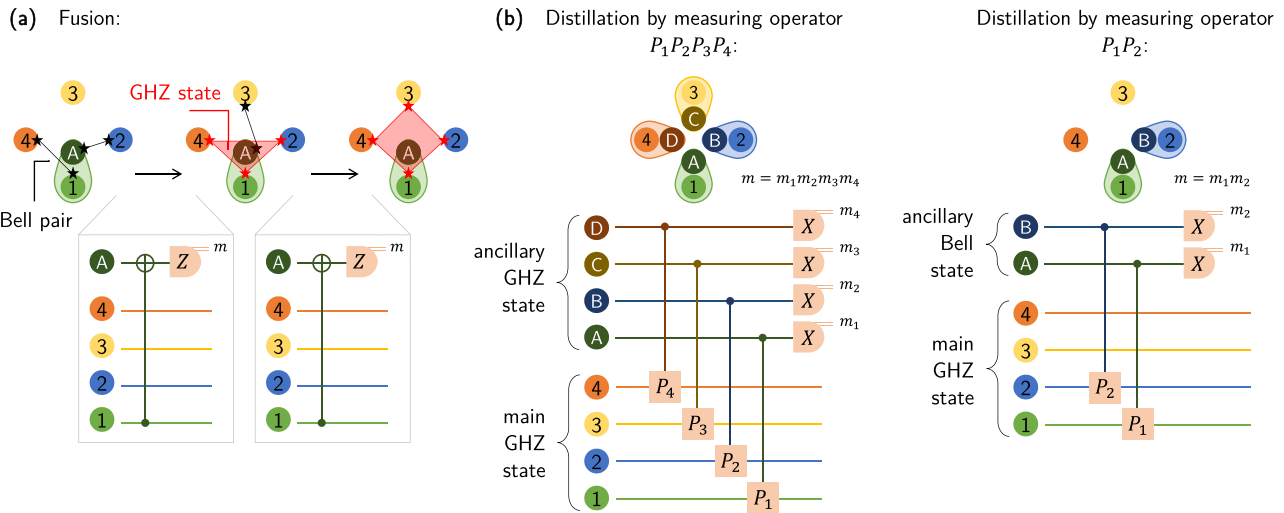


FIG. 1. (a) Example of the *fusion* operation applied in the dynamic program of Algorithm 1. Fusion can be applied on any two states $|\text{GHZ}_{n_1}\rangle$ and $|\text{GHZ}_{n_2}\rangle$ that overlap in one or more network nodes. (b) Examples of the *distillation* operation of Algorithm 1. Distillation consists on using one state of the form $|\text{GHZ}_{n_1}\rangle$ to measure a stabilizer of $|\text{GHZ}_{n_2}\rangle$ —this could, e.g., be the operator $X_1X_2\dots X_{n_2}$, or Z_1Z_2 .

We use a *dynamic program* to optimize over the space of GHZ protocols. This program generates GHZ protocols by using two main operations: *fusing* Bell pairs to create GHZ states, and *distilling* or *purifying* Bell or GHZ states by consuming other ancillary Bell pairs or GHZ states. Figure 1 depicts the two building blocks.

Distillation involves the use of an ancillary state to non-locally measure a stabilizer of the main Bell or GHZ state.⁵⁴ In this process, local control-Pauli gates between ancillary and main state qubits are followed by individual measurements of the ancillary state qubits in the Pauli-X basis. Obtaining an even number of -1 measurement outcomes marks a successful distillation attempt. If distillation fails, the post-measurement state is discarded and (part of) the protocol has to be carried out again.

Fusion is executed to create GHZ states out of Bell pairs and to create a larger GHZ state. A state of the form $|\text{GHZ}_{m_1}\rangle$ can be fused with a state $|\text{GHZ}_{m_2}\rangle$ by applying a CNOT gate between one qubit of both states and measuring out the target qubit in the Pauli-Z basis. Obtaining a $+1$ measurement outcome results in the state $|\text{GHZ}_{m_1+m_2-1}\rangle$. A -1 measurement outcome leads to the same state after local Pauli-X corrections.

In Algorithm 1, we present a schematic, pseudo-code version of the dynamic program we used to generate and evaluate GHZ protocols. This algorithm is an expanded version of the dynamic program in Ref. 55. In this algorithm, each protocol is created with either a fusion or a distillation operation that combines two smaller GHZ protocols encountered earlier in the search. The protocols created in this fashion can be depicted with a directed *binary tree* graph. An example graph is given on the left side of Fig. 2. For the distillation steps in the binary tree diagrams, we consume the state on the right to distill the state on the left.

Algorithm 1. Base dynamic program for GHZ protocols search.

```

Require:  $n_m$ : number of qubits of final GHZ state
 $k_m$ : minimum number of Bell pairs used
 $V$ : set with model parameters used
 $N_b$ : protocols stored in buffer per step
 $N_{so}$ : Monte Carlo shots used per protocol
for  $\{(n, k) \mid 2 \leq n \leq n_m, n - n_m + k_m \geq k \geq n - 1\}$  do
  # Try all non-local measurement combinations.
  for  $g \in \{\text{stabilizers of } |\text{GHZ}_n\rangle\}$  do
     $n' \leftarrow$  weight of  $g$ 
    for  $k' \in [n' - 1, k - n + 1]$  do
      for  $(p_1, p_2) \in [1, N_b] \times [1, N_b]$  do
         $\mathcal{P}_1 \leftarrow$  protocol  $p_1$  in buffer at  $(n, k - k')$ 
         $\mathcal{P}_2 \leftarrow$  protocol  $p_2$  in buffer at  $(n', k')$ 
        Construct binary tree protocol  $\mathcal{P}_{\text{new}}$  that
        measures  $g$  on  $\mathcal{P}_1$  by consuming  $\mathcal{P}_2$ 
        Construct protocol recipe  $\mathcal{R}_{\text{new}}$  and evaluate
        quality over  $N_{so}$  iterations times using  $V$ 
        Store protocol if average performance
        is better than worst protocol in buffer
      # Try all fusion combinations.
    for  $n_2 \in [2, n - 1]$  do
       $n_1 \leftarrow n - n_2 + 1$ 
      for  $k_2 \in [n_2 - 1, k - n + 1]$  do
         $k_1 \leftarrow k - k_2$ 
  
```

26 July 2024 12:54:07

```

for  $(p_1, p_2) \in [1, N_b] \times [1, N_b]$  do
   $\mathcal{P}_1 \leftarrow$  protocol  $p_1$  in buffer at  $(n_1, k_1)$ 
   $\mathcal{P}_2 \leftarrow$  protocol  $p_2$  in buffer at  $(n_2, k_2)$ 
  for  $(i, j) \in [1, n_1] \times [1, n_2]$  do
    Construct binary tree protocol  $\mathcal{P}_{new}$  by
    fusing  $\mathcal{P}_1$  at qubit  $i$  and  $\mathcal{P}_2$  at qubit  $j$ 
    Construct protocol recipe  $\mathcal{R}_{new}$  and evaluate
    quality over  $N_{so}$  iterations using  $V$ 
    Store protocol if average performance
    is better than worst protocol in buffer
    
```

Each binary tree corresponds to multiple inequivalent protocols depending on the time ordering of the steps. We define a *protocol recipe* as a set of instructions for implementing the protocol. The recipe includes the ordering of operations and state generation. An example of a protocol recipe can be seen on the right side of Fig. 2. This step was not required in previous research on distributed surface codes, as the noise models used in previous research did not include memory decoherence. Without a notion of time, the execution order of the tree's branches is irrelevant.

As can be seen in Fig. 2, the conversion to a protocol recipe contains SWAP gates. These gates are required to match the connectivity constraints of our example hardware model—see Sec. IV for more

details. The SWAP gates should, therefore, not be considered as fundamental elements of these protocols and can be circumvented or neutralized in hardware systems with more operational freedom. We implement SWAP gates as three CNOT gates.

Although we did not optimize over the conversion from binary tree to protocol recipe, we considered two heuristics to limit the influence of decoherence and of SWAP gates. To limit decoherence, we prioritize creating larger branches of the tree. Here, a branch is defined as an element of the binary tree (i.e., an operation in the GHZ protocol) including all elements that (in)directly point toward it. The size of the branch is the number of elements it contains. Because, generally speaking, creating small branches is faster than creating large branches, this heuristic aims to minimize waiting times for completed intermediate branches of the GHZ protocol.

The SWAP gate count can be limited by making sure a Bell pair that is consumed in a distillation step is the last state to be generated. This prevents the protocol from having to swap this state back and forth between the memory. For this reason, if two branches have equal size, we prioritize creating the left branch over the right one.

In constructing the protocol recipe, we first use these heuristics to determine the order in which the elementary Bell pairs are generated—i.e., the leaves of the binary tree. By following this order, we then check for each Bell pair if other Bell pairs in non-overlapping network nodes can be generated simultaneously. Here, we prioritize based on

Septimum protocol

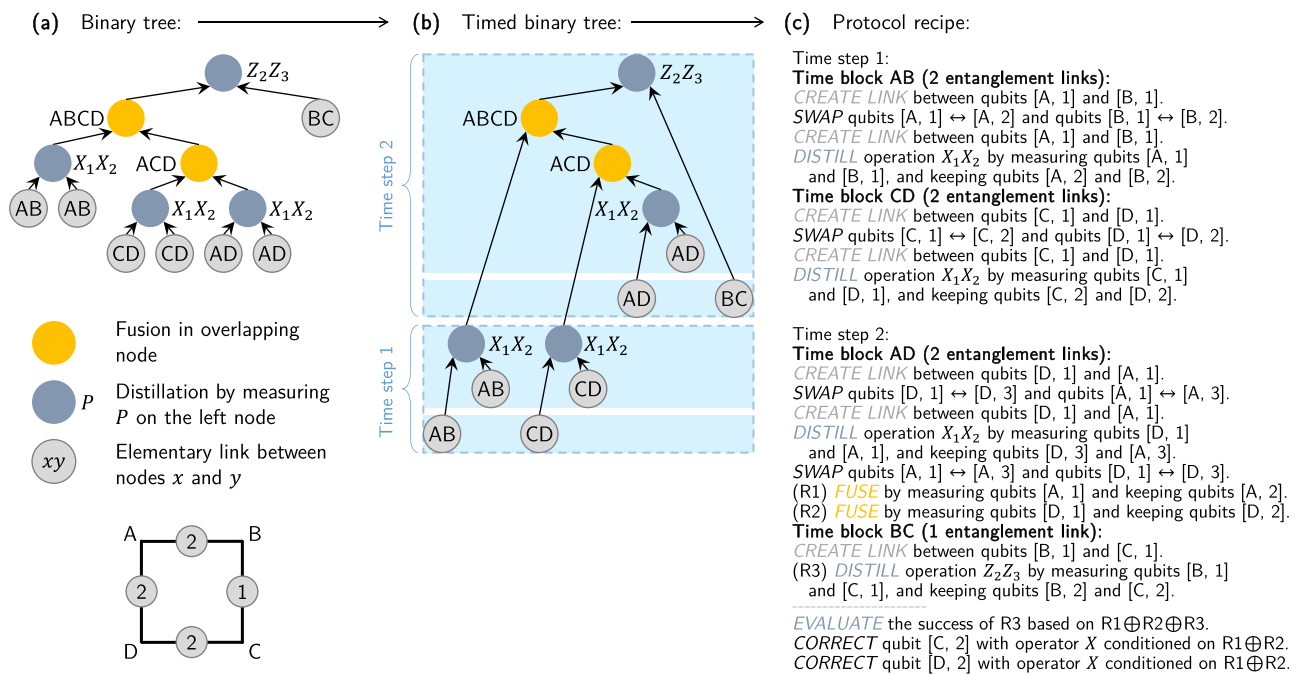


Fig. 2. (a) Binary tree with $k=7$ found with the dynamic program of Algorithm 1: the Septimum protocol. In this directed graph, the top vertex represents the final state. Each vertex describes how its corresponding state is created from a fusion or distillation operation involving its two children. At the origin of each branch, we find the elementary links: the Bell pairs. (b) We split the binary tree into multiple time steps that describe the order in which the protocol is carried out. The subtree involving the links between nodes C and D is identified as the part that we want to carry out first, since it is the left part of the largest branch of the binary tree. The subtree involving links between A and B is also added to time step 1, because it can be carried out in parallel (see Sec. II B for more information). (c) The timed binary tree is converted to an explicit set of operations: a protocol recipe. Here, we also add necessary SWAP gates, conditional corrections for fusion operations, and evaluations of distillation operations. For distillation operations, we also add instructions in case of failure (not printed here). During the execution of this protocol, the system waits until all branches of a time step are completed before continuing to the next time step. This protocol recipe uses a maximum of $q=3$ qubits per network node to generate the GHZ state.

26 July 2024 12:54:07

proximity in the binary tree. We include instructions for distillation, fusion, and SWAP operations at the earliest possible point of execution. This approach gives rise to a unique conversion from binary tree to protocol recipe. More detailed descriptions of the protocol recipe construction and execution procedure can be found in Ref. 56 and the supplementary documents of the repository of Ref. 57.

While our dynamic program explores a large class of protocols, not all of the seven protocols that we introduced in Sec. II A can be generated. This is because, to suppress calculation time, the program limits distillation steps to operations that use an ancillary entangled state to non-locally measure a stabilizer operator of the main state, in a sequential manner. The protocols Refined, Expedient, and Stringent, however, make use of the so-called *double selection* distillation block⁵⁸ that does not directly fit into this stabilizer distillation framework.^{59–61}

III. DISTRIBUTED TORIC CODE

In this section, we discuss the steps of a distributed toric code and our approach for its simulation.

A. The toric surface code

In the toric surface code,^{31,32} data qubits are placed on the edges of an $L \times L$ lattice with periodic boundary conditions. It encodes two logical qubit states. The stabilizers of the code come in two forms: the product of X operators on the four qubits surrounding every vertex of

the lattice, and the product of Z operators on the four qubits surrounding every face (or plaquette) of the code.

We consider a network topology with node connectivity matching the connectivity of the toric code lattice. We present a schematic impression in Fig. 3. Each data qubit of the toric code is placed in a separate network node—e.g., in a separate diamond color center. The nodes have access to local memory qubits to create and store entangled links between them. Entangled links can be used to create four-qubit GHZ states, as described in Sec. II, which are then consumed to measure the stabilizers of the code. Figure 4 shows a depiction of the procedure.

The outcomes of the stabilizer measurements, known as the *error syndrome*, are fed to a decoder to estimate the underlying error pattern. Here, we consider an implementation by Hu^{62,63} of the *Union-Find*⁶⁴ error decoder.

We point out that we simulate the toric surface code as a logical quantum *memory* and do not consider its resilience against, e.g., logical operations or operations required for initializing logical information. This means we restrict the study to the code’s ability to protect general logical states.

We opted for the toric surface code over the planar surface code (i.e., the surface code with boundaries) because, on top of the weight-4 stabilizer operators in the bulk, the planar code has lower-weight stabilizers at its boundaries. For distributed implementations, measuring these lower-weight stabilizers requires additional entanglement protocols. This makes simulating the planar code more complicated. Studies reveal that the introduction of boundaries typically has a limited effect

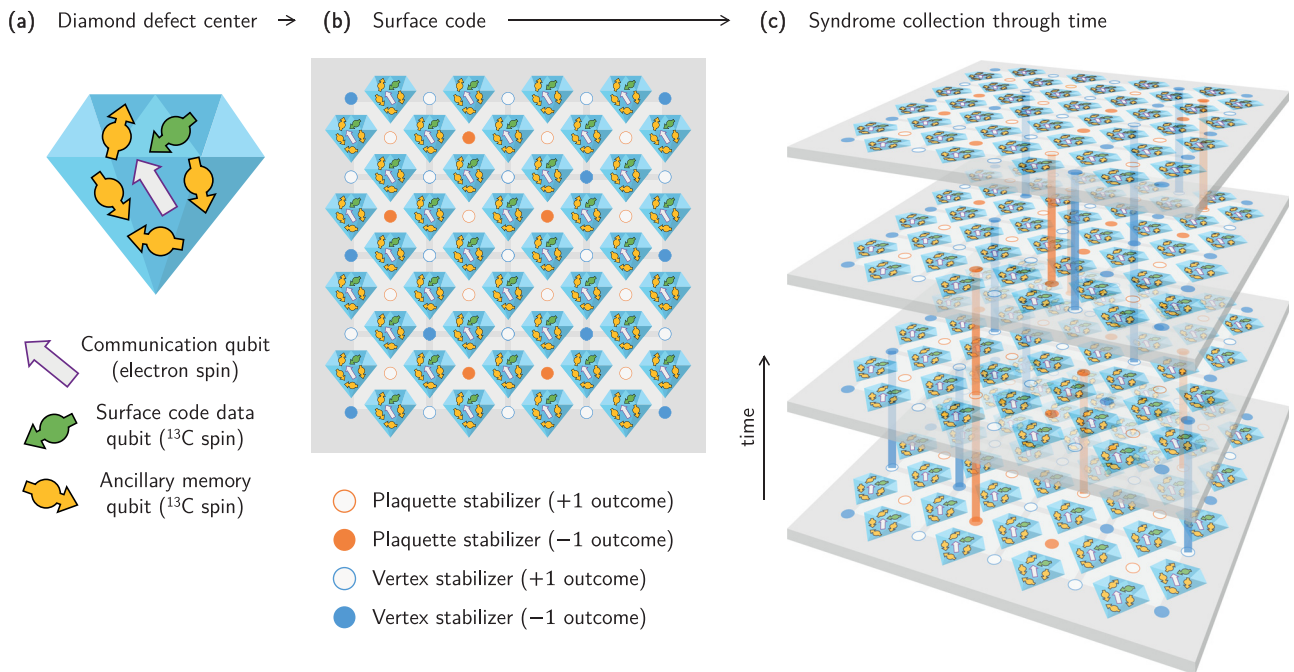


Fig. 3. (a) Schematic impression of a diamond defect center—also known as a diamond color center. The communication qubit is used to generate Bell pairs and to perform gates on the available qubits in the diamond color center. Out of all carbon spin memory qubits available, one is selected as the data qubit of the code. The other available memory qubits are used to store intermediate entangled states during the GHZ creation process. (b) Schematic impression of how a network of diamond color centers can be used to realize a surface code on a 4×4 square lattice. Each center holds one data qubit of the error-correction code on one of its memory qubits. GHZ states are generated to measure the stabilizer operators of the code, resulting in an *error syndrome* of +1 and -1 stabilizer measurement outcomes. (c) Stabilizer operators are measured consecutively in different time layers. A flip in stabilizer measurement outcome from one layer to the next is registered. The three-dimensional error syndrome that is created in this way is fed to an *error syndrome decoder* to locate errors.

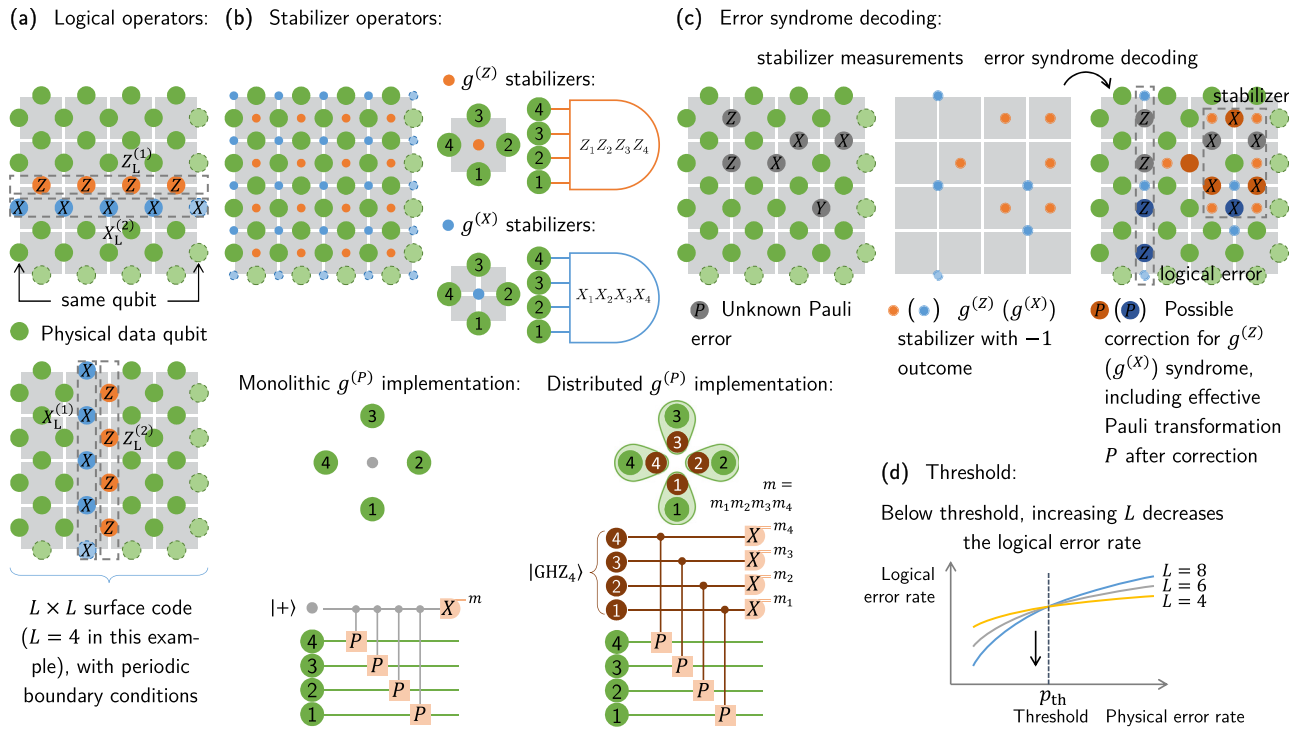


FIG. 4. General overview of the toric surface code. (a) The multi-qubit operators $Z_L^{(1)}$ and $X_L^{(2)}$ are the logical operators of the toric surface code. (b) The code's stabilizer generator operators are four-qubit operators surrounding every face $[g^{(Z)}]$ and every vertex $[g^{(X)}]$ of the lattice. In the distributed version, each stabilizer measurement requires the generation of an ancillary GHZ state between the nodes involved in the stabilizer measurement. (c) Unknown Pauli errors that appear on the physical data qubits can be tracked and corrected by measuring the stabilizer generators, and decoding the resulting *error syndrome*. The unknown errors and the correction can still lead to a logical error. (d) The logical error probability increases with the physical error probability. This relation depends on the lattice sizes L . This can give rise to a physical error rate *threshold*: below the threshold, the logical error rate decreases with the lattice size.

on the code's threshold values, yet it is likely to result in a slower suppression of the logical error rates below these thresholds.⁶⁵

B. Distributed toric code simulation

We split the simulation of the distributed toric code into two levels: simulation of the toric code's stabilizer measurements with the aid of GHZ states and simulation of L rounds of stabilizer measurements of the code itself.

The first level characterizes the stabilizer measurement. To this aim, we use Monte Carlo simulation to construct the (average) *Choi state* associated with using the protocol's GHZ state to measure the plaquette or star stabilizer operator on half of the maximally entangled state. Exploiting channel-state duality, the Choi state from each iteration is converted to a *superoperator* describing the stabilizer measurement channel. A formal introduction on channel characterization with a maximally entangled state can be found in Ref. 66. The superoperator construction is described in detail in Appendix C.

The second level is a toric code simulator that takes as noise model the average superoperator obtained in the first level. Following previous research,³⁴ we consider a stabilizer measurement cycle consisting of two rounds of plaquette stabilizer measurements and two rounds of star stabilizer measurements. This is because the constraint that each network node only has one single communication qubit in our example hardware model makes it impossible to simultaneously

generate entanglement for overlapping stabilizers—see Sec. IV for more details. By splitting the full cycle up into four rounds, each defect center becomes part of exactly one stabilizer measurement per round. This process is schematically depicted in Fig. 5. We note that a different hardware model could require, or benefit from, a different scheduling of the stabilizer measurements as the one used here.

Due to entanglement distillation steps in the GHZ creation protocol, GHZ generation is probabilistic. To fix the duration of the rounds we impose a “GHZ cycle time” t_{GHZ} : if GHZ generation is not successful within this time, it aborts. In that case, the corresponding stabilizer operator cannot be measured. This information could be given to the decoder in the form of an erasure symbol. However, to leverage existing decoders, we opt to duplicate the last measured value of the stabilizer. This choice is suboptimal and better thresholds could be expected for decoders that can handle erasures and noisy measurements. The GHZ cycle time is a free parameter that we explore in Sec. V. In Sec. 5 of Appendix B, we describe a heuristic-driven approach for selecting a suitable GHZ cycle time.

To model GHZ generation failures, at the first level, we construct two separate average superoperators per stabilizer type: a *successful* superoperator $\bar{S}_{success}$ for iterations where the GHZ state is created within t_{GHZ} , and an *unsuccessful* superoperator \bar{S}_{fail} for iterations where the GHZ could not be created. Both superoperators incorporate the influence of decoherence up to the cycle time on the code's data qubits.

26 July 2024 12:54:07

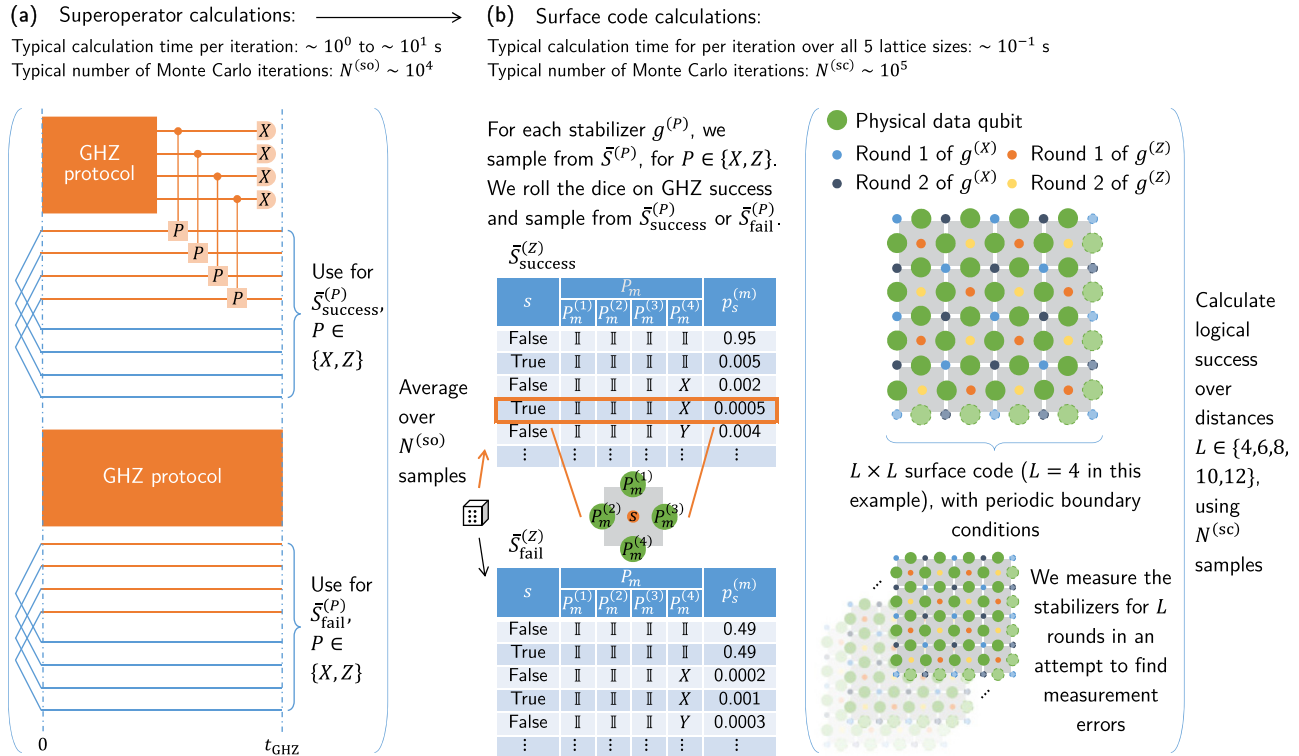


Fig. 5. Calculation process for error probability threshold simulations of the distributed surface code. This process is called for each specific protocol recipe and parameter set combination. The full calculation consists of two levels of Monte Carlo simulations: (a) the calculation of the superoperators $\bar{S}_{\text{success}}^{(P)}$ and $\bar{S}_{\text{fail}}^{(P)}$, for $P \in \{X, Z\}$, and (b) the surface code simulations using these superoperators.

IV. DIAMOND COLOR CENTER MODEL

In this section, we present an overview of the noise models and model parameters used in combination with the distributed surface code structure of Sec. III A. The noise models are based on the nitrogen-vacancy (NV) center.^{16,19,21} More information about the experimental characterization can be found in Appendix A. More details about the models can be found in Appendix B. The parameter values can be found in Table I.

In our model, qubits undergo both *generalized amplitude damping* and *phase damping* noise,⁶⁷ with separate T_1 and T_2 times. The generalized amplitude damping channel decoheres to the maximally mixed state $(|0\rangle\langle 0| + |1\rangle\langle 1|)/2$. Decoherence of the electron qubit is governed by $T_{1,\text{idle}}^e$ and $T_{2,\text{idle}}^e$ coherence times. For the memory qubits, we use different coherence times: $T_{1,\text{idle}}^n$ and $T_{2,\text{idle}}^n$ for when the node is idling vs $T_{1,\text{link}}^n$ and $T_{2,\text{link}}^n$ during optical entanglement creation. This is because the required operations on the electron qubit typically induce additional decoherence on the memory qubits.^{19,21} The Kraus operators of both channels can be found in Sec. 2 of Appendix B.

To mitigate decoherence from quasi-static noise processes in diamond color center experiments, *dynamical decoupling* is typically interleaved into gate sequences on both the electron and nuclear spins.¹⁶ Here, the coherence times that we consider are also those achieved for NV center spin registers with dynamical decoupling.¹⁶ Consequently, in our numerical models, gate operations must be performed only between two consecutive dynamical decoupling pulses—i.e., at the

refocusing point of the qubit spins involved in the operations. We define the center-to-center time of consecutive refocusing points as $t_{\text{DD}} = t_{\text{pulse}} + 2n_{\text{DD}}t_{\text{link}}$, where t_{pulse} is the time of a π -pulse, t_{link} is the duration of a single Bell pair generation attempt, and n_{DD} is a fixed number of Bell pair generation attempts. In Sec. 3 of Appendix B, we discuss how n_{DD} is optimized. In our model, we assume that all memory qubits of a node are decoupled synchronously.

We assume that each diamond color center only possesses a single communication qubit. Within each node, we further assume that measurements are only possible on its communication qubit, and local (i.e., intra-node) two-qubit gates always require the communication qubit to be the control qubit. These requirements mean we have to use SWAP gates to measure the state of a memory qubit or to use a memory qubit as the control of a two-qubit gate, as can be seen in Fig. 2. Finally, we assume that a Bell pair between two centers can only be generated between the communication qubits of the two centers. We note that, in general, one could design (combinations of) diamond color center nodes with multiple communication qubits. Although this limits the number of SWAP gates required for distillation, this gives rise to extra requirements on the memory robustness of the communication qubits.

The generation of Bell pairs between two color centers is modeled as a probabilistic process with a success probability p_{link} and time t_{link} per attempt. We constructed an analytic model to calculate p_{link} and the Bell pair density matrix after success, both for the *single-click* (i.e., the *single-photon*)⁶⁸ entanglement protocol and for the *double-click*

TABLE I. Simulation parameters used. The parameters are introduced in Sec. IV and Appendix B. More details on the values used for scaling parameters f_{dec} , f_{ϕ} , and f_{η} , as well as the relations used for $p_{\text{EE}}(f_{\phi})$, $F_{\text{link}}(f_{\phi})$, $\eta_{\text{ph}}(f_{\eta})$, $p_{\text{link}}(f_{\eta})$, and $\eta_{\text{link}}^*(f_{\eta})$, can be found in the captions of the respective figures.

	State-of-the-art (Refs. 16 and 21)	Figure 6	Figure 7	Figure 8
Bell pair model input				
Protocol	Single-click (Ref. 68)		Double-click (Ref. 69)	
F_{prep}	0.99 (Refs. 39 and 71)		0.999	
p_{EE}	0.04 (Ref. 15)	$p_{\text{EE}}(f_{\phi})$		0.01
μ	0.9 (Refs. 15 and 19)		0.95	
λ	0.984 (Refs. 15 and 70)		1	
η_{ph}	0.0046 (Ref. 72)	0.4472		$\eta_{\text{ph}}(f_{\eta})$
Bell pair model output				
p_{link}	0.0001		0.1	$p_{\text{link}}(f_{\eta})$
F_{link}	0.8966	$F_{\text{link}}(f_{\phi})$	0.9526	
Operation durations				
t_{link}		6×10^{-6} s		
t_{meas}		4×10^{-6} s		
$t_{X,Y}^e$		0.14×10^{-6} s		
$t_{X,Y}^n$		1.0×10^{-3} s		
$t_{Z,H}^e$		0.1×10^{-6} s		
$t_{Z,H}^n$		0.5×10^{-3} s		
$t_{\text{CZ,CX,CY}}$		0.5×10^{-3} s		
t_{SWAP}		1.5×10^{-3} s		
Decoherence				
$T1_{\text{idle}}^n$		300 s		
$T1_{\text{link}}^n$	0.03 s (Ref. 19)	0.3 s	$0.03f_{\text{dec}}$ s	0.3 s
$T1_{\text{idle}}^e$		300 s		
$T2_{\text{idle}}^n$		10 s		
$T2_{\text{link}}^n$	0.0075 s (Ref. 19)	0.075 s	$0.0075f_{\text{dec}}$ s	0.075 s
$T2_{\text{idle}}^e$		1.0 s		
t_{pulse}		1.0×10^{-3} s		
n_{DD}	500	18		Eq. (B7)
Link efficiency—see Eq. (1)				
η_{link}^*	2×10^{-1}	2×10^3	$200f_{\text{dec}}$	$\eta_{\text{link}}^*(f_{\eta})$
Operation noise				
p_{g}	0.01		Threshold	
p_{m}	0.01			

26 July 2024 12:54:07

(i.e., the *two-photon*)⁶⁹ entanglement protocol. The following five noise sources are included in this analytic model: the preparation error of the initial spin-photon state F_{prep} , the probability of an excitation error p_{EE} , a parameter λ based on the standard deviation of the phase uncertainty due to the path-length difference between the two arms (all modeled as dephasing channels on the involved qubits), the photon indistinguishability μ for each Bell state measurement (i.e., Hong–Ou–Mandel visibility, modeled with altered measurement operators⁷⁰), and the total photon detection probability η_{ph} in each arm (modeled with an amplitude damping channel).

For the double-click protocol, we assume the phase uncertainty to not be relevant and set $\lambda = 1$. The parameters F_{prep} , p_{EE} , and μ affect the fidelity F_{link} of the Bell pair state of the double-click protocol, whereas the parameter η_{ph} limits the success probability p_{link} of a single entanglement attempt. For the single-click protocol, the fidelity F_{link} is additionally influenced by η_{ph} and λ , and p_{link} depends on the indistinguishability μ . A full description of the density matrices and success probabilities of both entanglement protocols can be found in Sec. 1 of Appendix B.

The link efficiency η_{link}^* , introduced in Sec. I, is defined in terms of the parameters p_{link} , t_{link} , $T1_{\text{link}}^n$, and $T2_{\text{link}}^n$ as

$$\eta_{\text{link}}^* = \frac{2p_{\text{link}}}{t_{\text{link}}((T1_{\text{link}}^n)^{-1} + (T2_{\text{link}}^n)^{-1})}. \quad (1)$$

We assume that all operations take a finite amount of time. The time durations can be found in Table I. We neglect the influence of classical communication times, as we consider distances between network nodes to be relatively small, but include synchronization of network nodes when classical communication is required. Furthermore, we assume single-qubit gates to be noiseless, while noise in two-qubit gates is modeled with a depolarizing channel with probability p_g (see Sec. 4 of Appendix B). We model imperfect measurements by flipping the measurement outcome with probability p_m .

V. RESULTS

In this section, we investigate the sensitivity of the distributed toric surface code performance with respect to several physical parameters of the diamond color center hardware model. In particular, we investigate the influence of two-qubit gate and measurement noise, the entanglement success probability, the coherence times during entanglement generation, and the quality of the generated Bell pairs on the noise thresholds. The threshold values p_{th} are determined with fits of the logical success rates vs the lattice size (L) and local two-qubit gate and measurement error rate ($p_g = p_m$). The details of the fitting procedure can be found in Appendix D.

A. Current state-of-the-art parameter set

Let us first consider a parameter set inspired by state-of-the-art NV center hardware (see the first column of Table I). The operation times in this set are based on typical time scales in nitrogen-vacancy centers with a natural ¹³C concentration^{16,19,21}—see Appendix A for more details. The Bell pair parameter values are a collection of the best parameters in current NV center literature—see the first column of Table I for relevant citations. In the following, we explicitly refer to this parameter set as the “state-of-the-art” parameter set. As discussed in more detail in Appendix A, for this set, the single-click entanglement protocol outperforms the double-click protocol.

We did not find a noise threshold for the state-of-the-art parameter set—neither with existing GHZ protocols (see Sec. II A) nor when optimization over GHZ protocols (see Sec. II B). This finding is consistent with earlier investigations with a simplified NV center model.²¹ We identify two main limitations. The first one is the link efficiency: in this regime, the average entanglement generation times are longer than coherence times during entanglement generation—i.e., $\eta_{\text{link}}^* < 1$. On top of that, the Bell pair fidelity is relatively low. A low Bell pair fidelity requires complex distillation protocols to achieve high-quality GHZ states. This, in turn, magnifies the impact of decoherence.

B. Near-term parameter sets

As expected, further experimental progress and improved fidelities are required for fault-tolerant quantum computation. In the remainder of this section, we characterize two key parameters that drive the code performance in this regime. These findings can be used to guide future hardware development. Specifically, we investigate the effect of improving the Bell pair fidelity and the link efficiency.

1. Sensitivity to Bell pair fidelity

First, we investigate the influence of the Bell pair fidelity by using a near-future setting parameter set—see the second column in Table I. Compared to the state-of-the-art parameter set of Sec. V A, in this set coherence times during entanglement creation and the photon detection probability are one and two orders of magnitude higher, respectively. The double-click entanglement protocol now gives rise to the best combination of entanglement success probability and Bell pair fidelity, as explained in more detail in Appendix A. This means that these near-future parameters allow for an increase in the link efficiency by four orders of magnitude compared to the state-of-the-art parameter set of Sec. V A—see Eqs. (1), (B2), and (B4).

In our Bell pair model, several parameters contribute to the infidelity of the Bell pair states similarly—i.e., through the parameter ϕ of Eq. (B3) that captures all dephasing noise of the model. To investigate the sensitivity of the performance with respect to the Bell pair fidelity, we vary the influence of dephasing by scaling the probability of double excitation probability and off-resonant excitation errors. These are considered one of the leading error sources in present experiments.⁷³ We show the results in Fig. 6. In this figure, the bottom of the horizontal axis indicates the Bell pair fidelity; the top indicates the corresponding excitation error probability.

We find $p_g = p_m$ thresholds between $p_{\text{th}} = 0.0066(25)\%$ for $F_{\text{link}} \approx 0.78$ and $p_{\text{th}} = 0.193(4)\%$ for $F_{\text{link}} \approx 0.96$. Interestingly, the minimum fidelity for which we find a threshold, $F_{\text{link}} \approx 0.78$, is lower than the state-of-the-art Bell pair fidelity demonstrated with both the single-click and double-click protocols.^{39,73} This is possible because the link efficiency allows performing several distillation steps.

We find different optimal protocols as a function of the Bell pair fidelity. In particular, we find that the optimal protocols require more distillation steps as we reduce the Bell pair fidelity, ranging from $k = 12$ for $F_{\text{link}} \approx 0.78$ to $k = 7$ for $F_{\text{link}} \approx 0.96$. We find lower thresholds as we decrease the Bell pair fidelity since the more complex distillation protocols amplify the effect of decoherence and require more gates. Furthermore, since existing GHZ creation protocols either have a small number ($k \leq 8$) or many ($k \geq 16$) distillation steps, we can understand why the new protocols with $k \in \{10, 11, 12\}$ outperform them in this regime.

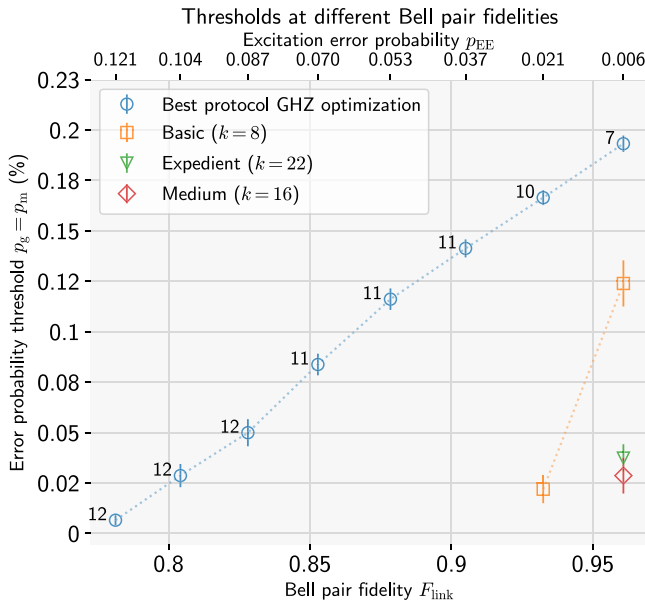


FIG. 6. Toric surface code error probability thresholds found for $p_g = p_m$, at various Bell pair fidelities F_{link} (see Table I for the parameter values). For all points on the horizontal axis of this plot, we have set $\mu = 0.95$ and $F_{\text{prep}} = 0.999$, and varied the excitation error probability p_{EE} . This leads to different values for the parameter ϕ that describes the fidelity of the Bell pairs—see Sec. 1 of Appendix B for more details. For $f_\phi \in \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$, we use $p_{\text{EE}}(f_\phi) = 1 - (\phi(f_\phi) / (\sqrt{\mu}(2F_{\text{prep}} - 1)^{1/2}))^{1/2}$, with $\phi(f_\phi) = 0.72 + 0.03f_\phi$. In case of similar performance for the best protocols found in the GHZ optimization, we show the protocol with the lowest k value. This value is printed above the blue markers.

2. Sensitivity to the link efficiency

Second, we investigate the influence of the link efficiency for near-future parameter values. In particular, we make use of a Bell pair fidelity $F_{\text{link}} \approx 0.95$ —close to the highest value in Sec. VB 1—and we investigate two options for varying the link efficiency.

First, we vary the link efficiency by varying the coherence times during entanglement generation. For this investigation, which we report in Fig. 7, we use the parameter set of the third column of Table I. In this set, we use a high photon detection probability $\eta_{\text{ph}} = 0.4472$, leading to an optical entanglement success probability of $p_{\text{link}} = 0.1$. The $p_g = p_m$ threshold values vary between $p_{\text{th}} = 0.020(8)\%$ with coherence times corresponding to $\eta_{\text{link}}^* = 4 \times 10^2$ and $p_{\text{th}} = 0.240(9)\%$ for coherence times corresponding to $\eta_{\text{link}}^* = 2 \times 10^5$. For coherence times corresponding to $\eta_{\text{link}}^* = 3 \times 10^2$ and lower, we did not find thresholds. At the other end of the spectrum, we evaluate the thresholds in an idealized modular scenario: in particular, in the absence of decoherence and with perfect SWAP gates (the last two points on the horizontal axis of Fig. 7). The last point corresponds to a scenario similar to the one analyzed in Ref. 34. We report a similar threshold value. For the Stringent protocol, the difference of $p_{\text{th}} = 0.775\%$ reported in Ref. 34 and $p_{\text{th}} = 0.601(29)\%$ found here can be attributed to the choice of the error-syndrome decoder and a reduced number of syndrome measurement cycles.

We now verify that, in this regime, similar thresholds can instead be found by varying the link efficiency via the entanglement generation

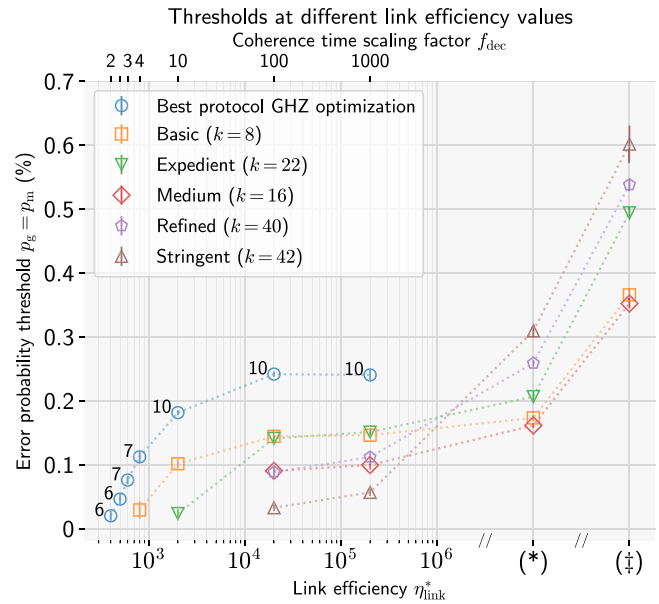


FIG. 7. Toric surface code error probability thresholds found for $p_g = p_m$, at various values of the coherence times during entanglement generation $T_{\text{link}}^n(f_{\text{dec}}) = 0.03f_{\text{dec}}$ s and $T_{\text{link}}^{2n}(f_{\text{dec}}) = 0.0075f_{\text{dec}}$ s. The f_{dec} considered are on the top horizontal axis of the plot. The other simulation parameters are in the third column of Table I. The corresponding link efficiency is $\eta_{\text{link}}^*(f_{\text{dec}}) = 2f_{\text{dec}} \times 10^2$. In case of similar performance for the best protocols found in the GHZ optimization, we show the protocol with the lowest k value. This value is printed above the blue markers. Point (*) shows calculations for a scenario without decoherence. Point (‡) shows calculations for a scenario without decoherence and with noiseless SWAP gates.

rate. Specifically, we vary the entanglement success probability by adjusting the total photon detection probability η_{ph} . For this investigation, which we report in Fig. 8, we use the parameter set in the fourth column of Table I. This set contains coherence times during entanglement generation that are ten times higher than the state-of-the-art coherence times of Sec. VA.19 We find $p_g = p_m$ thresholds between $p_{\text{th}} = 0.035(4)\%$ for a photon detection probability corresponding to $\eta_{\text{link}}^* \approx 4.7 \times 10^2$ and $p_{\text{th}} = 0.181(4)\%$ for a photon detection probability corresponding to $\eta_{\text{link}}^* = 2 \times 10^3$. At photon detection probability corresponding to $\eta_{\text{link}}^* \approx 3.6 \times 10^2$ and lower, we are not able to find threshold values.

The second investigation gives rise to a similar required link efficiency ($\eta_{\text{link}}^* \approx 4.7 \times 10^2$) as the first investigation ($\eta_{\text{link}}^* \approx 4 \times 10^2$). The small difference can be attributed to the slightly larger influence of the idling coherence time T_{idle}^n in a scenario with a smaller entanglement rate. This shows that the link efficiency captures the key trade-off between cycle duration and decoherence rate, even when experimental overhead such as dynamical decoupling is accounted for.

We find that the parameter set used determines which GHZ protocol works the best. However, for a large range of parameters close to the state-of-the-art set, one protocol with $k = 7$ performs the best. We call this protocol Septimum and detail it in Fig. 2. In particular, this protocol is (one of) the best-performing protocol(s) at $F_{\text{link}} \approx 0.96$ in Fig. 6, in the range $5 \times 10^2 \lesssim \eta_{\text{link}}^* \lesssim 2 \times 10^3$ in Fig. 7, and in the range $6.3 \times 10^2 \lesssim \eta_{\text{link}}^* \lesssim 1.1 \times 10^3$ in Fig. 8. We identify four additional well-performing protocols found with our dynamic program in Appendix E.

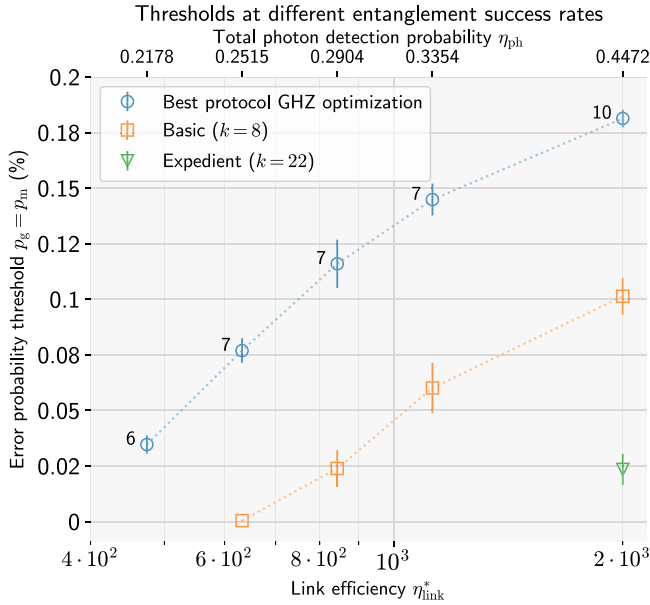


Fig. 8. Toric surface code error probability thresholds found for $p_g = p_m$, at various values of the total photon detection probability η_{ph} . This parameter takes on values $\eta_{ph}(f_\eta) = \sqrt{2} \times 10^{0.0625f_\eta - 0.8125}$, for $f_\eta \in \{0, 1, 2, 3, 5\}$. With the double-click protocol, this gives rise to $p_{link}(f_\eta) = 10^{0.125f_\eta - 1.625}$ and $\eta_{link}^*(f_\eta) = 0.002 \times 10^{0.125f_\eta + 5.375}$. The other simulation parameters are in the fourth column of Table I. In case of similar performance for the best protocols found in the GHZ optimization, we show the protocol with the lowest k value. This value is printed above the blue markers.

C. GHZ cycle time sensitivity

In the following, we investigate the sensitivity of threshold values to the GHZ cycle time and the associated GHZ completion probability for our diamond defect center hardware model. We present the results in Fig. 9. In this figure, we see a clear dependence of the optimal GHZ completion probability on protocol complexity. In particular, protocols that take longer to finish (i.e., protocols with more distillation steps) peak at lower GHZ completion probabilities than those that finish faster, due to their increased susceptibility to decoherence. We see that for a protocol with relatively small k , GHZ cycle times that correspond to GHZ completion probabilities between 99.2% and 99.8% give rise to the highest threshold values in the parameter regimes considered here, whereas protocols with a large k peak at GHZ completion probabilities between approximately 92.5% and 98.5%.

We notice that, for some GHZ protocols, noise thresholds are found at relatively low GHZ completion probabilities of 90% and lower. This behavior can be directly attributed to the decoder heralding failures in the GHZ generation: as mentioned in Sec. III B, we utilize the stabilizer outcome from the previous time layer if a GHZ protocol does not finish within the GHZ cycle time, as opposed to naively performing the stabilizer measurement with the state produced by an unfinished GHZ protocol.

The results in Fig. 9 show that thresholds can strongly vary on the GHZ cycle time. For computational reasons, except for the results in this subsection, we do not optimize over the GHZ cycle time. Instead, we use a heuristic method to select this time based on the k value of each protocol. We describe this method in Sec. 5 of Appendix B.

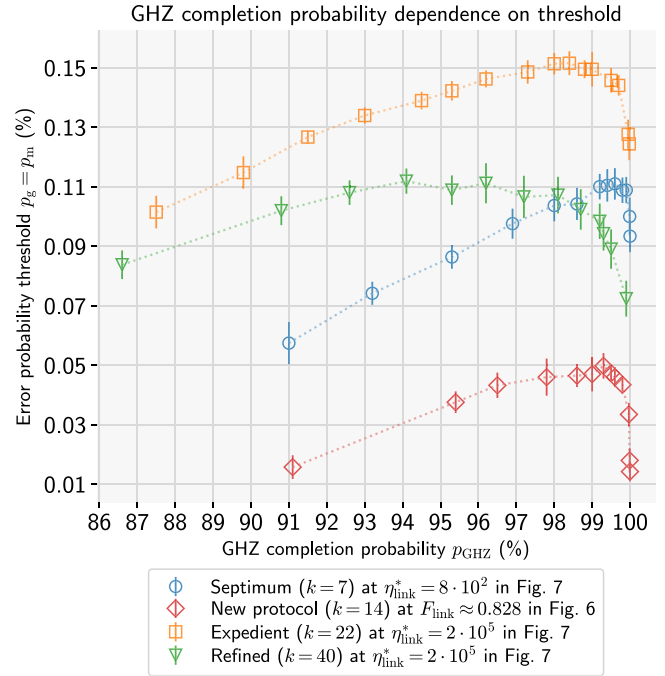


Fig. 9. Dependence of toric surface code error probability thresholds for $p_g = p_m$ on GHZ completion probability p_{GHZ} . The dependence is plotted for four protocols with a varying number of distillation steps k . Each data point is calculated with a different GHZ cycle time t_{GHZ} . The GHZ completion probability is the probability for a protocol to finish within t_{GHZ} . In Fig. 10, we plot the threshold values against the specific t_{GHZ} times used to achieve these results.

VI. DISCUSSION: FEASIBILITY OF PARAMETER SETS BELOW THRESHOLD

In Sec. V, we observed that the state-of-the-art parameter set is above the threshold. We identified two apparent drivers for this behavior: the Bell pair fidelity and the link efficiency. The sensitivity investigation shows that with a high link efficiency, the requirements on the Bell pair fidelity are modest, while even with a high Bell pair fidelity, a high link efficiency is still necessary.

Let us first discuss the experimental feasibility of the minimum link efficiency $\eta_{link}^* \geq 4 \times 10^2$ found in Fig. 8. First of all, the link efficiency can be increased by either increasing the coherence times of the data and memory qubits, or by increasing the entanglement success probability—or by a combination of both. In Sec. V, we found thresholds with a high success probability ($p_{link} = 0.1$) and a modest increase in the coherence times. However, we also found that with high coherence times during entanglement generation (ten times higher than the state-of-the-art¹⁹) and Bell pair fidelities of $F_{link} \approx 0.95$, the total photon detection probability needs to fulfill $\eta_{ph} \approx 0.19$ (Fig. 8). This is a factor 50 above the state-of-the-art parameter value.

The total photon detection probability is the product of multiple probabilities (see Sec. 1 of Appendix B). Present network experiments utilizing NV centers are particularly limited by two of these: the probability of emitting in the zero-phonon-line (ZPL, $\approx 3\%$)¹¹ and the total collection efficiency ($\approx 10\% - 13\%$).^{19,39} Both values are expected to increase by Purcell enhancement of the NV center emission, for example with the use of optical microcavities^{13,18} or nanophotonic

devices.^{74,75} However, even with such devices, the feasibility remains unclear. For microcavities, predicted ZPL emission and collection probabilities are of the order 10 to 46%^{13,18} and 49%,⁷¹ respectively. Moreover, the successful integration of Purcell-enhanced and optically coherent NV centers in nanophotonic devices remains an open research challenge due to the detrimental effects of surface charges.²²

This realization has led to an increased interest in other color centers in the diamond lattice, as, e.g., SiV and SnV defect centers.²⁷ These centers have higher intrinsic emission probabilities into the ZPL—for SnV centers this is reportedly in the area of 60%,⁷⁶ whereas SiV centers approximately emit 70 to 80% into the ZPL.⁷⁷ Additionally, the inversion symmetry of SnV and SiV centers makes them less susceptible to proximal charges, facilitating integration into nanophotonic devices. Nanophotonic structures offer advantages over microcavities, such as stronger cooperativities enabled by the small mode volumes,²³ and reduced sensitivity to vibrations of the cryostat hosting the emitter.¹⁸ A disadvantage of SnV and SiV centers over NV centers is the fact that they need to be operated at lower temperatures²⁴ or under high strain^{78,79} to achieve similar coherence times.

Additionally, these alternative trajectories provide opportunities for “direct” GHZ generation schemes, where a GHZ state is created without Bell pair fusion.⁸⁰ Contrary to the photon-emission-based Bell pair generation with NV centers, these direct GHZ state generation schemes could be based on the transmission or reflection of photons. Since, for nodes with a single communication qubit, SWAP gates are unavoidable when performing fusion, getting rid of SWAP gates during GHZ state generation could relax the requirements for, e.g., the link efficiency and the photon detection probability.

VII. CONCLUSION

In this paper, we investigated the influence of decoherence and other noise sources on a fault-tolerant distributed quantum memory channel with the toric code. For this, we developed an open-source package that optimizes GHZ distillation for distributed stabilizer measurements and quantifies the impact of realistic noise sources.⁵⁷ The GHZ protocols found with this package are compatible with a second open-source package that calculates logical error rates of the (distributed) surface code.⁶³

We focused our attention on a specific set of noise models inspired by diamond defect centers. We first observed that state-of-the-art nitrogen-vacancy center hardware does not yet satisfy the thresholds. A parameter-sensitivity analysis shows that the main driver of the performance is the link efficiency, giving a benchmark for future experimental efforts. The photon detection probability of state-of-the-art hardware appears to represent the main challenge for operating the surface code below threshold. Sufficient photon detection probabilities could be achieved with the help of Purcell enhancement of NV center emission, or using other color centers such as silicon-vacancy centers or tin-vacancy centers. The use of other color centers also presents opportunities for schemes that directly generate GHZ states between the communication qubits of more than two nodes—i.e., without fusing Bell pairs.⁸⁰

With our detailed noise models, we found threshold values up to 0.24%. This is three to four times lower than prior thresholds found with less-detailed models. Similarly, the optimal distillation protocols have a small number of distillation steps compared to prior work. For a large parameter regime of parameters, a protocol consuming a minimum of seven Bell pairs was optimal. Its experimental demonstration

would be an important step for showing the feasibility of this approach for scalable quantum computation.

We performed a thorough optimization of GHZ distillation protocols. However, further improvements in other elements of the distributed architecture could partially bridge the gap with the performance of monolithic architectures. For instance, the surface code decoder could model unfinished GHZ distribution rounds as erasure noise. The conversion of binary trees to protocol recipes can be optimized and Bell pair distribution could be scheduled dynamically. On top of that, since our software allows for the implementation of general hardware models, further research could focus on analyzing and understanding a broad range of physical systems in the distributed context. In addition to exploring alternative hardware systems, it would be intriguing to implement a more in-depth model of the system’s micro-architecture.

ACKNOWLEDGMENTS

The authors would like to thank Hans Beukers, Johannes Borregaard, Kenneth Goodenough, Fenglei Gu, Ronald Hanson, Sophie Hermans, Stacey Jeffery, Yves van Montfort, Jaco Morits, Siddhant Singh, and Erwin van Zwet for helpful discussions and feedback. The authors gratefully acknowledge support from the joint research program “Modular quantum computers” by Fujitsu Limited and Delft University of Technology, co-funded by the Netherlands Enterprise Agency under project number PPS2007. This work was supported by the Netherlands Organization for Scientific Research (NWO/OCW), as part of the Quantum Software Consortium Program under Project 024.003.037/3368. This work was supported by the Netherlands Organisation for Scientific Research (NWO/OCW) through a Vidi grant. This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 852410). D.E. was partially supported by the JST Moonshot R&D program under Grant JPMJMS226C. The authors thank SURF (www.surf.nl) for the support in using the National Supercomputer Snellius.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Sébastien de Bone, David Elkouss and Paul Möller designed and conceptualized the study. Conor E. Bradley and Tim H. Taminiau designed, built, and conducted experiments to characterize color centers. Conor E. Bradley and Tim H. Taminiau curated the hardware parameter sets. Sébastien de Bone built the software to generate and evaluate GHZ protocols. Sébastien de Bone and Paul Möller built the software to carry out distributed surface code simulations. Sébastien de Bone carried out and analyzed the numerical simulations. The manuscript is written by Sébastien de Bone with input and revision from all authors. David Elkouss supervised the project.

Sébastien de Bone: Conceptualization (equal); Data curation (lead); Formal analysis (lead); Methodology (equal); Software (equal); Visualization (lead); Writing – original draft (lead); Writing – review & editing (lead). **Paul Möller:** Conceptualization (equal); Methodology (equal); Software (equal). **Conor E. Bradley:** Resources

(lead); Writing – review & editing (supporting). **Tim H. Taminiau:** Resources (supporting); Supervision (supporting); Writing – review & editing (supporting). **David Elkouss:** Conceptualization (equal); Formal analysis (supporting); Supervision (lead); Writing – original draft (supporting); Writing – review & editing (supporting).

DATA AVAILABILITY

The data that support the findings of this study are openly available in 4TU.ResearchData, Ref. 57.

APPENDIX A: EXPERIMENTAL CHARACTERIZATION

The noise models and parameter values we use in this paper are based on experimental observations by Bradley *et al.*^{16,21} and Pompili *et al.*¹⁹ Experiments were performed on NV centers at temperatures of 3.7 and 4 K. Time scales for gates are based on microwave and radio frequency pulses for single-qubit gates on the electron and ¹³C qubits, respectively. Time scales for two-qubit gates are based on phase-controlled radio frequency driving of the carbon spins interleaved with dynamical decoupling sequences on the electron state, following the scheme described by Bradley *et al.*^{12,16}

Characterization of the decoherence model and the associated coherence times was achieved by monitoring the drop in probability of detecting Pauli matrix eigenstates at $t > 0$ after preparing the state at $t = 0$. For the T_1 relaxation time, the $|0\rangle$ and $|1\rangle$ states were used, and the expectation value $\langle Z \rangle$ of a measurement in the Pauli-Z basis was determined after several delay times—the coherence time then directly follows from the observed exponential drop in the expectation value. With the $|+\rangle$ and $|-\rangle$ states and the expectation $\langle X \rangle$ for measurement in the Pauli-X basis, and with $|+i\rangle$ and $| - i \rangle$ and $\langle Y \rangle$, the T_2 coherence time of our model could be determined. For these four states, the observed exponential decay T_{dec} corresponds to the T_2 time of our model via $1/T_2 = 1/T_1 - 2/T_{\text{dec}}$.

The full model is based on the so-called NV^- state, a spin-1 electron qubit with spin projection quantum number $m_s \in \{-1, 0, 1\}$. Stochastic ionization can convert the NV^- state to the NV^0 state with $m_s \in \{-1/2, 1/2\}$. Under the present understanding, this spin state is no longer usable as a qubit due to a fast orbital relaxation process.⁸¹ Since the electron-spin state is used to control the ¹³C spins, ionization accordingly dephases the ¹³C states. As such, the coherence times of the NV center memory are currently limited by these ionization effects. Reference 21 proposes a method to mitigate ionization-induced memory dephasing by actively recharging the NV^0 state. They show ionization and recharging can be performed with minimal loss in fidelity in the state of a ¹³C spin in an isotopically purified device. This marks an important avenue for future research. In our model, we do not specifically include ionization, but simply absorb its influence in the ¹³C coherence times.

Furthermore, Ref. 21 showed that isotopically engineered nitrogen-vacancy devices with a reduced ¹³C memory qubit concentration (0.01%) are able to store a quantum state over 10^5 entanglement attempt repetitions. Compared to samples with natural ¹³C abundance (1.1%), the memory qubits have a weaker coupling to their color centers. While this increases coherence times during entanglement generation by several orders of magnitude, it also leads to longer time scales for carrying out gates on the memory qubits and gates

between the communication and memory qubits. In natural abundance devices, a quantum state can typically be stored over 10^3 entanglement attempt repetitions,¹⁹ while demonstrated single qubit ¹³C gates are typically ≈ 13 times faster and two-qubit gates are typically ≈ 50 times faster than the isotopically purified samples.

Nonetheless, in this paper, we assume that the diamond color centers contain a natural abundance of carbon memory qubits (1.1%). Thus, we trade-off lower coherence times during entanglement generation for faster gates. This choice was made because it is believed that in future systems entanglement success rates are required to be several orders of magnitude higher than current state-of-the-art. In those regimes, fewer entanglement attempts are required and the influence of decoherence during entanglement generation becomes smaller. It is believed that one then benefits more from having the faster operations that samples with natural concentrations of ¹³C atoms offer.

On top of that, Ref. 21 found an idling carbon coherence time of $T_{\text{idling}}^{\text{p}} = 10$ s, which is too low for running the GHZ creation protocols described in this paper when considering the corresponding gate speeds. This time—comparable to those achieved in natural abundance devices¹⁶—was limited predominantly by other impurities in the diamond, but the expected linear scaling of coherence time with isotopic concentration remains to be demonstrated in future work. In regimes with high entanglement success rates, the time duration of the GHZ creation protocols is almost solely described by the duration of the two-qubit CZ, CNOT, C_Y, and SWAP gates, which are ≈ 50 slower for the isotopically purified samples. Thus, the operation time of any protocol also takes ≈ 50 time longer with isotopically purified samples. Equivalent performance for isotopically purified samples would require that $T_{\text{idling}}^{\text{p}}$ also increases by a factor of 50—i.e., from 10 to 500 s.

Further research into isotopically engineered diamond defect centers is ongoing. This will likely lead to a better understanding of the trade-off between the ¹³C concentration and the associated operation times and decoherence rates.

APPENDIX B: SIMULATION MODELS AND SETTINGS

1. Bell pair state

For generating entanglement, we assume that the diamond color centers make use of either the single-click⁶⁸ or double-click⁶⁹ Bell pair entanglement protocol. We denote the *bright state* population coefficient for the single-click protocol as α . Phase uncertainty originating from a path length difference between the two involved parties is modeled as a dephasing channel on one of the photonic states before the Bell state measurement, with fidelity^{70,71}

$$\lambda = \frac{1}{2} \left(1 + \frac{I_1(\sigma(\varphi)^{-2})}{I_0(\sigma(\varphi)^{-2})} \right). \quad (\text{B1})$$

Here, I_0 and I_1 are modified Bessel functions of the zeroth and first order, and $\sigma(\varphi)$ is the standard deviation of the phase instability. We only include phase uncertainty for the single-click protocol, for which a phase instability of 14.3° corresponds to $\lambda = 0.984$.¹⁵

The parameter η_{ph} describes the total photon detection probability per excitation. It can be considered as the product of the total collection efficiency (the transmissivity between defect center and

detector multiplied by the detector efficiency) with the probability that a photon is emitted in the detection (time) window and in the zero-phonon line. The photon detection probability only influences the success probability of the entanglement protocol.

We note that, in our model, we neglect photon detector *dark counts*. This is because we consider regimes in which dark counts are negligible.

The parameter p_{EE} describes the probability of an excitation error during a heralded entanglement generation event. This can occur because an extra photon was emitted during the excitation pulse (a phenomenon known as *double excitation*) or as a result of exciting the dark state. We assume that these excitation errors give rise to a dephasing channel on one of the qubits of the Bell pair.

For NV centers, the double excitation probability is estimated between 4% and 7%.^{15,19} The off-resonant excitations are typically assumed to be negligible. This is because the polarization of the light pulse only leads to a weak driving field on transitions close to the main (bright state) transition, and other transitions are sufficiently far off-resonant.

For other systems, the situation might be different. Here, one can design the excitation pulse to induce a π transition on the main transition and a full 2π rotation on the closest unwanted transition. Tiurev *et al.* created a model that, based on the energy difference between the main transition and the closest unwanted transition, allows one to estimate p_{EE} —i.e., both the double excitation probability and the probability of exciting this unwanted transition.^{82,83} Their model shows that, typically, the larger the energy difference is between the two transitions, the smaller p_{EE} becomes.

The single-click Bell pair state $\rho^{(sc)}$ is modeled in the following way:

$$\begin{aligned} \rho^{(sc)} &= F_+^{(sc)} |\Psi^+\rangle \langle \Psi^+| + F_-^{(sc)} |\Psi^-\rangle \langle \Psi^-| \\ &\quad + (1 - F_+^{(sc)} - F_-^{(sc)}) |00\rangle \langle 00|, \\ F_{\pm}^{(sc)} &= \frac{1}{p_{link}^{(sc)}} (1 \pm \phi) \eta_{ph} \alpha (1 - \alpha), \\ p_{link}^{(sc)} &= 2\eta_{ph} \alpha + \eta_{ph}^2 \alpha^2 \frac{\mu - 3}{2}. \end{aligned} \quad (B2)$$

Here, $|\Psi^{\pm}\rangle = (|01\rangle \pm |10\rangle) / \sqrt{2}$ are Bell states. $F_{link}^{(sc)} \equiv F_+^{(sc)}$ denotes the fidelity with respect to the target Bell state $|\Psi^+\rangle$. The parameter $p_{link}^{(sc)}$ denotes the success probability of a single attempt with this protocol. Furthermore, μ is the photon indistinguishability per Bell pair measurement.⁷⁰ The parameter ϕ contains all dephasing contributions of the model:

$$\phi = \sqrt{\mu(2F_{prep} - 1)^2(2\lambda - 1)(1 - p_{EE})^2}. \quad (B3)$$

In deriving these expressions, we have assumed the photon detectors to be non-photon-number resolving.

The dephasing parameter ϕ comes back in the expression for the double-click Bell pair state $\rho^{(dc)}$ with success probability $p_{link}^{(dc)}$:

$$\begin{aligned} \rho^{(dc)} &= F_{link}^{(dc)} |\Psi^+\rangle \langle \Psi^+| + (1 - F_{link}^{(dc)}) |\Psi^-\rangle \langle \Psi^-|, \\ F_{link}^{(dc)} &= \frac{1}{2} (1 + \phi^2), \\ p_{link}^{(dc)} &= \frac{\eta_{ph}^2}{2}. \end{aligned} \quad (B4)$$

Our single-click model combines different elements from NV center single-click models for large-scale networks.^{15,19,37,70–72} Our

model differs from these models in that we neglect dark counts in the photon detectors. More elaborate versions of the single-click and double-click models can be found in Refs. 84 and 85.

For the parameter value sets used in this paper, as presented in Table I of the main text, we use the single-click protocol to give an impression of the optimal Bell pair fidelity and success probability with state-of-the-art NV center parameter values. This is the parameter set identified as $F_{prep} = 0.99$, $p_{EE} = 0.04$, $\mu = 0.9$, $\lambda = 0.984$, and $\eta_{ph} = 0.0046$. In the simulations performed with near-future parameter values, i.e., the set based on $F_{prep} = 0.999$, $p_{EE} = 0.01$, $\mu = 0.95$, $\lambda = 1$, and $\eta_{ph} = 0.4472$, we use the double-click protocol to generate the Bell pair states.

This is because, for the state-of-the-art parameter set, the single-click protocol is the best option. As can be seen in Eqs. (B2) and (B4), for a specific set of parameter values, the double-click protocol has a fixed Bell pair fidelity and success probability. For the single-click protocol, on the other hand, the bright space population parameter α allows one to trade in a higher fidelity for a lower success probability, and vice versa. This gives us the option to select α such that the success probability is the same for both protocols. If, for that value of α , the state generated by the single-click protocol is better than the state generated with the double-click protocol, one could argue that the single-click protocol is the best choice.

This is the case for the state-of-the-art parameter set, as we get, with our double-click model, $p_{link}^{(dc)} \approx 1 \times 10^{-5}$ and $F_{link}^{(dc)} \approx 0.852$. With our single-click model, using $\alpha \approx 0.00115$, we get a similar success probability, but a state with higher fidelity: $F_{link}^{(sc)} \approx 0.905$. We note that setting α this low is typically not possible in practical situations. However, with the state-of-the-art parameter set, also for higher values of α the single-click model produces better success probabilities and fidelities than the double-click model. In Table I, we have used a higher value for α , leading to one order of magnitude higher $p_{link}^{(sc)}$, and slightly lower fidelity.

For the near-future parameters, the double-click model becomes favorable with $p_{link}^{(dc)} \approx 0.1$ and $F_{link}^{(dc)} \approx 0.953$. Setting α to get the same success probability with single-click now only leads to $F_{link}^{(sc)} \approx 0.873$. Technically, for this parameter set, it is possible to reach slightly higher fidelities with single-click than with double-click, but this gives rise to very low success probabilities—i.e., success probabilities unusable for the coherence times considered in this paper.

2. Decoherence

We describe decoherence noise channels $\mathcal{N}_{noise}(\rho) = \sum_{i=1}^{\kappa} K^{(i)} \rho (K^{(i)})^\dagger$ on a general state ρ in terms of their κ Kraus operators $\{K^{(i)}\}_{i=1}^{\kappa}$, with $\sum_{i=1}^{\kappa} K^{(i)} (K^{(i)})^\dagger = \mathbb{I}$.

The generalized amplitude damping channel and dephasing channel used to model NV center decoherence (see Sec. IV and Ref. 67) make use of the following Kraus operators [with $\gamma_1 = 1 - \exp(-t/T_1)$, where t is the time and T_1 is the coherence time]:

$$\begin{aligned} K_{GAD}^{(1)} &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma_1} \end{bmatrix}, & K_{GAD}^{(2)} &= \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & \sqrt{\gamma_1} \\ 0 & 0 \end{bmatrix}, \\ K_{GAD}^{(3)} &= \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{1-\gamma_1} & 0 \\ 0 & 1 \end{bmatrix}, & K_{GAD}^{(4)} &= \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 0 \\ \sqrt{\gamma_1} & 0 \end{bmatrix}. \end{aligned} \quad (B5)$$

For the phase damping channel, we have the following two Kraus operators [with $\gamma_2 = 1 - \exp(-t/T_2)$, where T_2 is the coherence time]:

$$K_{\text{PD}}^{(1)} = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma_2} \end{bmatrix}, \quad K_{\text{PD}}^{(2)} = \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{\gamma_2} \end{bmatrix}. \quad (\text{B6})$$

3. Dynamical decoupling sequence length

The coherence times used in our model are based on decoherence in NV center qubits that undergo dynamical decoupling (DD). In Sec. IV of the main text, we discuss how DD is implemented in our simulations. Each DD sequence has a length of $2n_{\text{DD}}t_{\text{link}} + t_{\text{pulse}}$. The values for n_{DD} used in the simulations were obtained by solving the following optimization problem:

$$n_{\text{DD}}(p_{\text{link}}) = \min_{n' \in \mathbb{Z}^+} \lim_{\mathcal{A} \rightarrow \infty} \sum_{i=1}^{\mathcal{A}} \sum_{j=1}^{\mathcal{A}} p_{\text{link}}^{(i)} p_{\text{link}}^{(j)} t_{n'}^{(i,j)}. \quad (\text{B7})$$

In Eq. (B7), we perform the minimization for n' as a member of the positive integers \mathbb{Z}^+ . The goal is to minimize n' over the average completion time of generating two Bell pairs in parallel, where we also wait for both nodes to finish their DD sequences. Here, $p_{\text{link}}^{(i)} = p_{\text{link}}(1 - p_{\text{link}})^{i-1}$ and $p_{\text{link}}^{(j)} = p_{\text{link}}(1 - p_{\text{link}})^{j-1}$ denote the probabilities of obtaining entanglement generation success at exactly the i th and j th attempt. Furthermore, $t_{n'}^{(i,j)} = \lceil \max(i,j)/(2n') \rceil \times (2n't_{\text{link}} + t_{\text{pulse}})$ is the effective time of performing the required entanglement attempts in this specific scenario, where $\lceil \max(i,j)/(2n') \rceil$ describes how many DD sequences are required for these attempts and $2n't_{\text{link}} + t_{\text{pulse}}$ describes the time of one DD sequence. To solve Eq. (B7) in a practical setting, it suffices to take a large number for \mathcal{A} instead of letting it go to infinity. Because finding n_{DD} in this way only minimizes the waiting and refocusing time during entanglement generation in two nodes, and not the waiting time during the other operations, this process does not lead to the optimal n_{DD} . We found that it does, however, give rise to values for n_{DD} that, typically, produce good results.

4. Gate and measurement noise

For gates, we consider a gate set consisting of the Pauli gates (X, Y, Z), the Hadamard gate, the CNOT (CX) gate, the CZ gate, and the CiY gate. These are not the native gates of the NV center, but their true gate set can be compiled into the gate set used here without additional costs in terms of two-qubit gates.²⁰ Noise on two-qubit gates are modeled with a depolarizing noise model:

$$\mathcal{N}_g(\rho) = (1 - p_g)\rho + \frac{p_g}{15} \sum_{(P_i, P_j)} (P_i \otimes P_j)\rho(P_i \otimes P_j)^\dagger, \quad (\text{B8})$$

where the sum is over $(P_i, P_j) \in \{\mathbb{I}, X, Y, Z\}^2 \setminus (\mathbb{I}, \mathbb{I})$.

Measurements are restricted to measuring (single-qubit) electron qubits in the Pauli-Z basis. Measuring in the Pauli-X basis is achieved with an additional Hadamard gate. Measurement errors are modeled by a probability p_m that the measurement projects onto the opposite eigenstate of the measured operator.

5. GHZ cycle time settings

In Sec. VC, we discuss the influence of the GHZ cycle time t_{GHZ} on the surface code threshold value. In this section, we discuss the heuristic method that we use for finding a suitable GHZ cycle time for a specific protocol at a specific set of error probabilities.

As discussed earlier, protocols with more distillation steps k take (on average) longer to finish. This means that, compared to a protocol with a smaller k , they require a longer t_{GHZ} to reach the same GHZ completion probability p_{GHZ} . However, because decoherence plays a larger role at a higher t_{GHZ} , we typically see that the threshold values obtained with protocols with higher k peak at a lower GHZ completion probability. This becomes clear in Figs. 9 and 10, where we plot how the threshold values change when using different GHZ cycle times for four protocols with varying k .

In an attempt to limit calculation times during our GHZ protocol search, we made use of a heuristic-driven method to estimate the optimal t_{GHZ} at a certain error probability configuration. Using the protocol-specific parameter k , we first identify an adequate GHZ completion probability as $p_{\text{GHZ}}^{(\text{aim})} = (100.2 - k/10)\%$. On top of that, we use prior knowledge for a rough estimate $p_{\text{th}}^{(\text{est})}$ of the value of the threshold. We then determine the distribution of the protocol's duration at the error probability $p_{\text{th}}^{(\text{est})}$, by running it without a GHZ cycle time. Finally, using this distribution, we determine t_{GHZ} by selecting a time at which at least a fraction $p_{\text{GHZ}}^{(\text{aim})}$ of the iterations will finish.

APPENDIX C: DETAILS REGARDING THE CALCULATION OF THE SUPEROPERATOR

1. Superoperator calculation

In this section, we describe how we calculate the superoperator that we use in the surface code simulations. Separately calculating a superoperator has the advantage that it breaks up the process of calculating GHZ state creation from the threshold simulations: this drastically decreases the complexity of the full calculation.

Following earlier work by Nickerson *et al.*, we assume that only Pauli errors occur on the data qubits during the toric code simulations.^{34,35} This simplifies the simulation, as every stabilizer measurement now deterministically measures either +1 or -1, and measurement results can be calculated by simply considering commutativity between Pauli errors and the stabilizer operators. In most situations, the stochastic Pauli error model can be considered a good approximation for coherent errors described by continuous rotations.⁸⁶ On top of that, since the nuclear spin qubits (i.e., the memory qubits) of NV centers have no states to leak to, it is believed that a depolarizing channel (i.e., Pauli noise) is a good approximation for noise on these qubits.

Our characterization of the toric code stabilizer measurements is carried out with density matrix calculations that do include more general errors. To align these calculations with the toric code calculations themselves, the stabilizer measurement channel is twirled over the Pauli group.⁸⁷⁻⁸⁹ This makes sure the superoperators describing the channel only contain Pauli errors. Each superoperator is constructed via the channel's Choi state—i.e., by using the GHZ state created by the concerning protocol to non-locally perform the stabilizer measurement on half of the maximally entangled state.

To explain this process in more detail, we consider the states $|\Psi_{\pm}\rangle$ that follow from projecting half of the maximally entangled state $|\Psi\rangle$ on the $+P^{\otimes 4}$ and $-P^{\otimes 4}$ subspaces with projectors $\Pi_{\pm} = (\mathbb{I}^{\otimes 8} \pm P^{\otimes 4} \otimes \mathbb{I}^{\otimes 4})/2$. Here, $P \in \{X, Z\}$ describes the two types of stabilizer measurements of the toric code, and $|\Psi\rangle$ is the eight-qubit maximally entangled state describing the four data qubits of the code. We also define states $|\Psi_{\pm}^{(m)}\rangle$ that describe Pauli

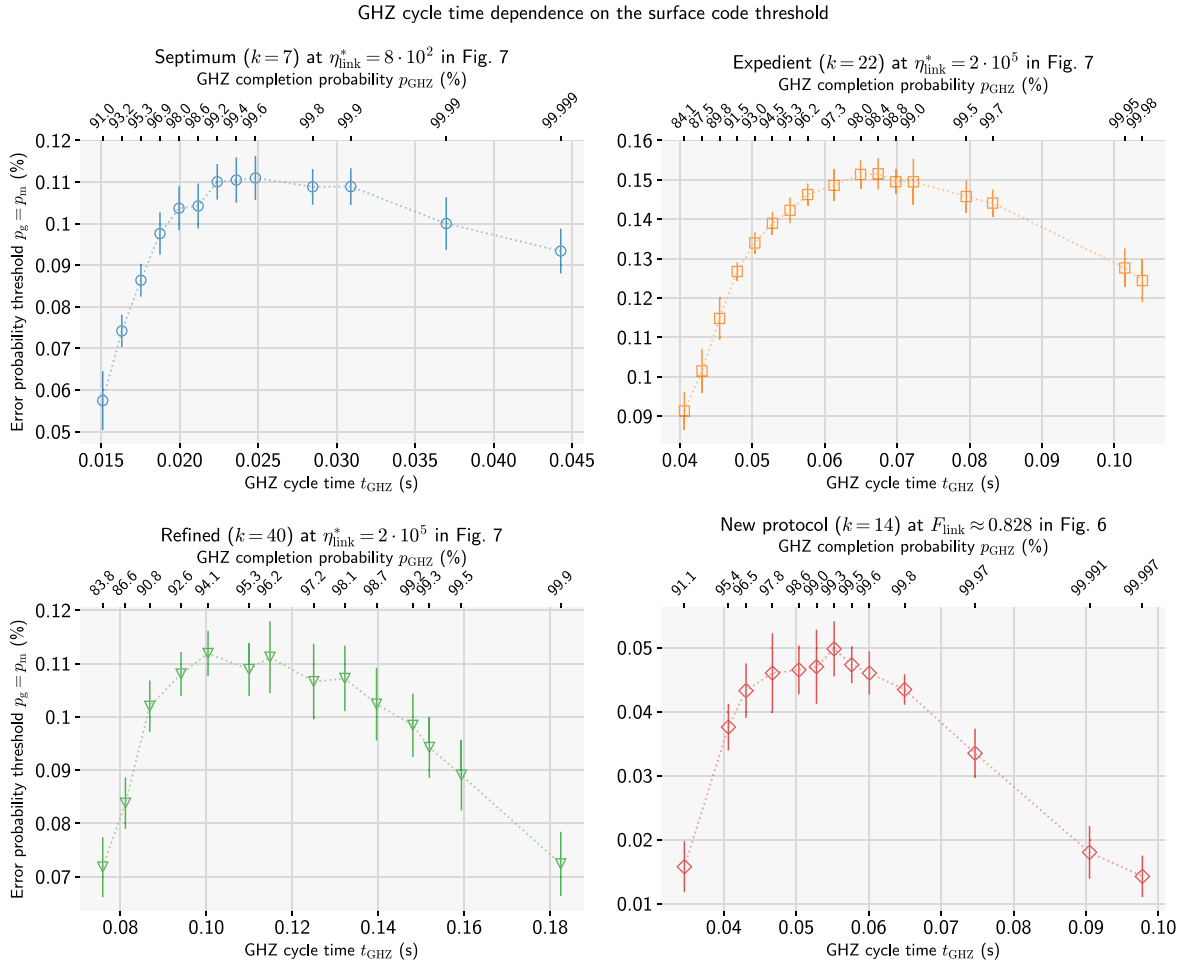


Fig. 10. Dependence of surface code error probability thresholds for $p_g = p_m$ on GHZ cycle time t_{GHZ} . Each t_{GHZ} gives rise to probability p_{GHZ} that a protocol has to finish within t_{GHZ} : for each t_{GHZ} , the associated probabilities are printed on the top x-axis of each plot. In Fig. 9, we directly plot the threshold values against p_{GHZ} for the same four protocols.

errors $P_m \in \{\mathbb{I}, X, Y, Z\}^{\otimes 4}$ occurring on the first half of $|\Psi_{\pm}\rangle$ after the projection with Π_{\pm} . We define

$$\begin{aligned}
 |\Psi\rangle &= \frac{1}{\sqrt{2^4}} \sum_{j=0}^{2^4-1} |j\rangle \otimes |j\rangle, \\
 |\Psi_{\pm}\rangle &= \frac{\mathbb{I}^{\otimes 8} \pm P^{\otimes 4} \otimes \mathbb{I}^{\otimes 4}}{\sqrt{2}} |\Psi\rangle, \\
 |\Psi_{\pm}^{(m)}\rangle &= (P_m \otimes \mathbb{I}^{\otimes 4}) |\Psi_{\pm}\rangle.
 \end{aligned}
 \tag{C1}$$

Later in the analysis, we only consider the subset of Pauli operators P_m that lead to orthogonal states $|\Psi_{\pm}^{(m)}\rangle$, i.e., we only use P_m that make sure we have

$$\langle \Psi_s^{(m)} | \Psi_{s'}^{(n)} \rangle = \delta_{mn} \delta_{ss'}
 \tag{C2}$$

with $(s, s') \in \{+, -\}^2$. We call this subset $\mathcal{E} \subseteq \{\mathbb{I}, X, Y, Z\}^{\otimes 4}$.

We define two versions of the full noisy stabilizer measurement channel: \mathcal{N}_+ , which projects with Π_+ , and \mathcal{N}_- , which projects with Π_- :

$$\mathcal{N}_s(\rho) = \sum_i K_s^{(i)} \Pi_s \rho \Pi_s (K_s^{(i)})^\dagger.
 \tag{C3}$$

Here, $s \in \{+, -\}$. The Kraus operators $K_s^{(i)}$ describe the noise on the data qubits. Each of them can be decomposed into Pauli matrices:

$$K_s^{(i)} = \sum_{P_q \in \{\mathbb{I}, X, Y, Z\}^{\otimes 4}} \zeta_{s,q}^{(i)} P_q.
 \tag{C4}$$

Using this decomposition, the channel's Choi state can be expressed in the following way:

$$\begin{aligned}
 \rho_{\text{Choi}} &= \sum_s (\mathcal{N}_s \otimes \mathbb{I}^{\otimes 4}) (|\Psi\rangle \langle \Psi|) \\
 &= \sum_s \sum_i \sum_{P_q, P_{q'}} \zeta_{s,q}^{(i)} (\zeta_{s,q'}^{(i)})^* |\Psi_s^{(q)}\rangle \langle \Psi_s^{(q')}|.
 \end{aligned}
 \tag{C5}$$

We now focus on ρ_{Choi} as the post-measurement state for stabilizer measurement outcome +1. The influence of noise can cause

26 July 2024 12:54:07

measurement errors, meaning ρ_{Choi} can contain terms projected with Π_- . One can extract coefficients $p_s^{(m)}$ from ρ_{Choi} by constructing the states $|\Psi_s^{(m)}\rangle$ from Pauli operators $P_m \in \mathcal{E}$ via

$$p_s^{(m)} = \langle \rho_{\text{Choi}} | \Psi_s^{(m)} \rangle = \sum_i |\zeta_{s,m}^{(i)}|^2. \quad (\text{C6})$$

These are the coefficients of the Pauli operators that act as the stabilizer measurement channel's Kraus operators after the channel is twirled over the Pauli group.

We see that this procedure gives us the probabilities required to construct the superoperator of the channel. If ρ_{Choi} is constructed by preparing the post-measurement state according to a +1 measurement outcome, the coefficients $p_+^{(m)}$ give rise to the Pauli errors $P_m \in \mathcal{E}$ without a measurement error on the stabilizer measurement, whereas the coefficients $p_-^{(m)}$ describe Pauli errors P_m accompanied with a measurement error. If ρ_{Choi} is constructed with a -1 measurement outcome, the role of $p_+^{(m)}$ and $p_-^{(m)}$ is inverted, but the parameter values themselves are the same.

The stabilizer fidelity is defined as the coefficient $p_s^{(m)}$ corresponding to $P_m = \mathbb{I}^{\otimes 4} \otimes \mathbb{I}^{\otimes 4}$ (i.e., no errors on the data qubits) and no stabilizer measurement error. In our search for well-performing GHZ creation protocols, a good reason for comparing two protocols by using the stabilizer fidelity over, e.g., the GHZ state fidelity is the fact that the surface code data qubits undergo more decoherence for protocols that take longer to finish. This aspect of the optimization problem is not taken into account if we just use the GHZ fidelity to compare the protocols.

2. Convergence of the average superoperator

The construction of a superoperator that we use in the surface code simulator requires averaging over a large number of Monte Carlo simulations. In this section, we investigate the convergence of the average superoperator over an increasing number of Monte Carlo samples. In Fig. 11, we calculate the average Choi state $\bar{\rho}_{\text{Choi}}$ over 3×10^5 Monte Carlo iterations and calculate the *trace distance* of this state with the average Choi state $\bar{\rho}_{\text{Choi}}^{(i)}$ after a smaller number of i iterations. As explained in more detail below, this figure suggests that, after 10^5 Monte Carlo samples, errors in the average superoperator elements are on the order of 10^{-4} .

For $s \in \{+, -\}$ and $P_m \in \mathcal{E}$, the superoperator used in threshold simulations is calculated as the average $\bar{S} = \{\bar{p}_s^{(m)}\}_{s,m}$ of the individual superoperators $S = \{p_s^{(m)}\}_{s,m}$ calculated with the method of Sec. 1 of Appendix C. Alternatively, this average superoperator can be constructed by calculating the average Choi state $\bar{\rho}_{\text{Choi}}$, as defined in Sec. 1 of Appendix C, and using Eq. (C6) to calculate $\{\bar{p}_s^{(m)}\}_{s,m}$ from this average Choi state.

The trace distance between two density matrices ρ and ρ' is defined as

$$T(\rho, \rho') = \frac{1}{2} \text{Tr} \sqrt{(\rho - \rho')^\dagger (\rho - \rho')}. \quad (\text{C7})$$

For an operator \mathcal{P} with eigenvalues $0 \leq \zeta_j \leq 1$, we can show that the following holds:⁶⁷

$$|\text{Tr}(\mathcal{P}(\rho - \rho'))| \leq T(\rho, \rho'). \quad (\text{C8})$$

This means that, for $\bar{\rho}_{\text{Choi}}^{(i)}$ after a certain iteration i , $T(\bar{\rho}_{\text{Choi}}, \bar{\rho}_{\text{Choi}}^{(i)})$ provides an upper bound on the difference between the average

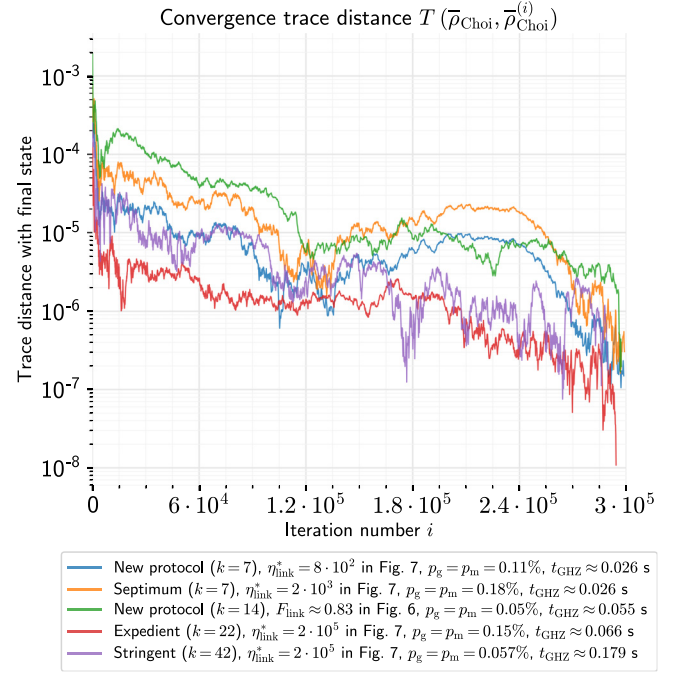


Fig. 11. Convergence of the trace distance between $\bar{\rho}_{\text{Choi}}$ (the average Choi state after $3 \cdot 10^5$ Monte Carlo iterations) and $\bar{\rho}_{\text{Choi}}^{(i)}$ (the average Choi state after i iterations). We track changes in the trace distances by varying i on the x-axis of the plot. For each data point, we add on the order of 100 new iterations to $\bar{\rho}_{\text{Choi}}^{(i)}$. In the plot, we include five GHZ protocols with a varying number of distillation steps k . We only include Choi states based on $X^{\otimes 4}$ stabilizer measurements and exclude iterations that did not finish within the GHZ cycle time t_{GHZ} .

superoperator $\bar{S}^{(i)}$ after iteration i and the average superoperator \bar{S} after the full number of iterations. This is because, for the difference in the elements of \bar{S} and $\bar{S}^{(i)}$, the following holds:

$$\begin{aligned} \Delta \bar{p}_s^{(m)} &= |\langle \Psi_s^{(m)} | \bar{\rho}_{\text{Choi}} | \Psi_s^{(m)} \rangle - \langle \Psi_s^{(m)} | \bar{\rho}_{\text{Choi}}^{(i)} | \Psi_s^{(m)} \rangle| \\ &= \left| \text{Tr} \left(|\Psi_s^{(m)}\rangle \langle \Psi_s^{(m)}| \left(\bar{\rho}_{\text{Choi}} - \bar{\rho}_{\text{Choi}}^{(i)} \right) \right) \right| \\ &\leq T \left(\bar{\rho}_{\text{Choi}}, \bar{\rho}_{\text{Choi}}^{(i)} \right). \end{aligned} \quad (\text{C9})$$

Calculating $T(\bar{\rho}_{\text{Choi}}, \bar{\rho}_{\text{Choi}}^{(i)})$, therefore, gives us information about the convergence of the average superoperator elements after i iterations.

In our simulations, as shown schematically in Fig. 5, the superoperator $\{\bar{p}_s^{(m)}\}_{s,m}$ is subsequently used in a second level of Monte Carlo simulations that emulates the operation of the surface code with this specific superoperator. In Appendix D, we discuss how, for a specific $\{\bar{p}_s^{(m)}\}_{s,m}$, the statistical uncertainty in the Monte Carlo simulations of the surface code leads to uncertainty in the calculated threshold value.

APPENDIX D: FITTING PROCEDURE FOR DETERMINING THRESHOLDS

In this appendix, we describe the fitting procedure we use to calculate toric surface code thresholds as well as the associated

uncertainties in the threshold values. To perform the fitting procedure described below, we make use of the `optimize.curve_fit` function of the SciPy package for Python.⁹⁰

1. Regression model

Calculating threshold values involves varying one or more of the error probabilities. For each combination of error probabilities p , we calculate an average superoperator using the methods described in Appendix C. At a specific p , we then use Monte Carlo simulations to calculate the logical success rate r of the toric surface code for multiple lattice sizes L .

We denote the observed logical success rate of a certain input combination (p_i, L_i) by r_i . We make use of n_C to describe the total number of input combinations: $\{(p_i, L_i)\}_{i=1}^{n_C}$. For a single (p_i, L_i) combination, the logical success rate is defined as the number of error-correction iterations M_i that do not induce a logical error divided by the full number of error-correction iterations N_i . In the context of this paper, N_i can be considered as the number of Monte Carlo iterations used for surface code calculations for a certain (p_i, L_i) and the exact hardware configuration used. We assume that the uncertainty in the observed logical success rates is described by the binomial distribution. This means that the standard deviation can be estimated via

$$\sigma_i = \sqrt{\frac{r_i(1-r_i)}{N_i}}, \quad \text{where } r_i = \frac{M_i}{N_i}. \quad (\text{D1})$$

Following Wang *et al.*, we fit the logical success rates with the following model:⁹¹

$$\hat{r} = \hat{a} + \hat{b}(p - \hat{p}_{\text{th}})L^{1/\hat{\kappa}} + \hat{c}(p - \hat{p}_{\text{th}})^2L^{2/\hat{\kappa}} + \hat{d}L^{-1/\hat{\zeta}}. \quad (\text{D2})$$

Using this model, we find estimates $\{\hat{r}_i\}_i$ of the logical success rates for all input combinations $\{(p_i, L_i)\}_i$. For a certain (p_i, L_i) , the residual $\hat{\epsilon}_i$ is defined as the difference between the observed logical success rate and the estimated value: $\hat{\epsilon}_i = r_i - \hat{r}_i$. Values for the seven fitting parameters \hat{a} , \hat{b} , \hat{c} , \hat{d} , \hat{p}_{th} , $\hat{\kappa}$, and $\hat{\zeta}$ are found by identifying their (local) minimum with respect to the sum Q of the “weighted” squared residuals. This sum is defined in the following way:

$$Q = \sum_{i=1}^{n_C} \left(\frac{\hat{\epsilon}_i}{\sigma_i}\right)^2 = \sum_{i=1}^{n_C} \left(\frac{r_i - \hat{r}_i}{\sigma_i}\right)^2. \quad (\text{D3})$$

We see that this approach makes sure that residuals of data points that are determined with high uncertainty (i.e., with a high standard deviation σ_i) are given less priority in the least-squares fit than data points with low uncertainty.

2. Weighted least-squares fitting procedure

To understand how the confidence intervals in the values of the fitting parameters are determined, we delve a bit deeper into how one could determine fitting parameters for a non-linear regression like Eq. (D2). In line with convention, we denote our input configuration as $X_i = (p_i, L_i)$, and we write \hat{r}_i as $\hat{r}_i = f(\hat{\beta}, X_i)$. Here, the function f is the function of Eq. (D2) and $\beta = (\hat{a}, \hat{b}, \hat{c}, \hat{d}, \hat{p}_{\text{th}}, \hat{\kappa}, \hat{\zeta})$ describes the converged values for the fitting parameters after the optimization. We define $n_p = 7$ as the number

of fitting parameters of the model. Furthermore, we use the notation $\beta^{(t)}$ to indicate the fitting parameter values at a certain step t during the optimization. We can now write $r_i = f(\beta^{(t)}, X_i) + \epsilon_i^{(t)}$ for each t . Here, the residuals also contain the superscript (t) to denote that they depend on the exact values of $\beta^{(t)}$.

Finding the least-squares fit is now achieved with the *Gauss–Newton algorithm*.^{92,93} This method is a variant of Newton’s method for finding the minimum of a non-linear function. We start with a guess $\beta^{(1)}$ for the fitting parameter values. These values are then iteratively updated by using the fact that we want to minimize the parameter Q of Eq. (D3) until they converge. To go from a certain $\beta^{(t)}$ to a new improved version $\beta^{(t+1)}$, one makes use of $r_i = f(\beta^{(t+1)}, X_i) + \epsilon_i^{(t+1)}$ to construct a new estimation in terms of the old fitting parameter values $\beta^{(t)}$. More explicitly, $f(\beta^{(t+1)}, X_i)$ is Taylor expanded around $\beta^{(t)}$, and the second and higher order terms are neglected. This allows one to write $r_i = f(\beta^{(t+1)}, X_i) + \epsilon_i^{(t+1)}$ as a matrix equation—i.e., with each of the n_C input and output combinations of X_i and r_i in a separate row of this equation. To this end, we put the fitting parameter values at step t of the optimization process into a $n_p \times 1$ column vector

$$\beta^{(t)} \equiv \begin{bmatrix} a^{(t)} & b^{(t)} & c^{(t)} & d^{(t)} & p_{\text{th}}^{(t)} & \kappa^{(t)} & \zeta^{(t)} \end{bmatrix}^T. \quad (\text{D4})$$

We do the same for the values of $\{X_i\}_i$ and $\{\epsilon_i^{(t)}\}_i$, and write them as $n_C \times 1$ column vectors \mathbf{X} and $\epsilon^{(t)}$, respectively. Finally, we define $\Delta\beta^{(t+1)}$ as $\Delta\beta^{(t+1)} \equiv \beta^{(t+1)} - \beta^{(t)}$. The full (Taylor expanded) version of $r_i = f(\beta^{(t+1)}, X_i) + \epsilon_i^{(t+1)}$ can now be rewritten as^{92,93}

$$\epsilon^{(t+1)} = \epsilon^{(t)} - \mathbf{J}^{(t)}\Delta\beta^{(t+1)}. \quad (\text{D5})$$

Here, $\mathbf{J}^{(t)}$ is an $n_C \times n_p$ matrix that contains the derivatives of f with respect to the fitting parameters, evaluated at $\beta^{(t)}$ and the inputs $\{X_i\}_i$. This matrix is also known as the *Jacobian matrix*.

The parameter Q from Eq. (D3) can also be rewritten as a matrix product as follows:

$$Q = (\epsilon^{(t+1)})^T \Sigma^{-1} \epsilon^{(t+1)} = (\epsilon^{(t)} - \mathbf{J}^{(t)}\Delta\beta^{(t+1)})^T \Sigma^{-1} (\epsilon^{(t)} - \mathbf{J}^{(t)}\Delta\beta^{(t+1)}). \quad (\text{D6})$$

Here, Σ is a diagonal matrix containing the variances of the observed values r_i as $\Sigma \equiv \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_{n_C}^2)$. More generally, one can use the full covariance matrix for Σ if this information is available.

To minimize Q with respect to $\Delta\beta^{(t+1)}$, one sets $\partial Q / \partial \Delta\beta^{(t+1)}$ to zero. This results in an expression that can be solved for $\Delta\beta^{(t+1)}$.⁹³

$$\Delta\beta^{(t+1)} = ((\mathbf{J}^{(t)})^T \Sigma^{-1} \mathbf{J}^{(t)})^{-1} (\mathbf{J}^{(t)})^T \Sigma^{-1} \epsilon^{(t)}. \quad (\text{D7})$$

In obtaining Eq. (D7), one has to assume that $\mathbf{J}^{(t)}$ does not depend on $\Delta\beta^{(t+1)}$ —it is exactly the incorrectness of this assumption for non-linear regression models that makes that the least-squares fitting parameters have to be found iteratively.

3. Uncertainty in fitting parameter values

The idea of the fitting procedure is that, after sufficient iterations $t \gg 1$, $\beta^{(t+1)}$ describes the final (converged) parameter values $\hat{\beta}$. Of course, in theory, in the limit $n_C \rightarrow \infty$, the fitting parameter

values $\hat{\beta}$ would converge to the *true* values, which we indicate with β . One can use another Taylor expansion to express $\hat{\beta}$ in terms of the true set of values β :

$$\hat{\beta} \approx \beta + (\hat{J}^T \Sigma^{-1} \hat{J})^{-1} \hat{J}^T \Sigma^{-1} \epsilon. \quad (\text{D8})$$

Here, ϵ describes the residuals $\epsilon_i = r_i - f(\beta, X_i)$ with respect to the true values of the fitting parameter. Furthermore, \hat{J} indicates the Jacobian evaluated with the calculated values $\hat{\beta}$. To get Eq. (D8), one has to assume that $\hat{\beta} \equiv \beta^{(t+1)} = \beta^{(t)}$ holds—i.e., the system of equations has fully converged.

Strictly speaking, in the limit $n_C \rightarrow \infty$, we have $\hat{\beta} = \beta$ and $\hat{\beta}$ does not have a distribution, since β contains constant values. For finite n_C , however, we can argue that $\hat{\beta}$ *does* have a distribution, and we use the last term of Eq. (D8) to estimate the uncertainty in the fitting parameters $\hat{\beta}$. This estimation again involves the assumption that \hat{J} is a constant matrix and does not depend on the fitting parameters.

Under this assumption for \hat{J} , we can make use of the fact that, for a general constant matrix A , the variance of AY is given by $\text{Var}(AY) = A \text{Var}(Y) A^T$. Together with the assumption that $\text{Var}(\epsilon)$ can be estimated by $\text{Var}(\epsilon) = \Sigma$, the covariance matrix of $\hat{\beta}$ can be expressed as^{92,93}

$$\text{Var}(\hat{\beta}) \approx (\hat{J}^T \Sigma^{-1} \hat{J})^{-1}. \quad (\text{D9})$$

The fact that we have a good idea of the uncertainties $\{\sigma_i\}_i$ of the observed $\{r_i\}_i$ means that we are able to estimate the quality of the obtained fit. The quality of the fit can be evaluated with the *reduced chi-squared* metric, which is defined as

$$\chi_\nu^2 = \frac{Q}{\nu} = \frac{1}{\nu} \sum_{i=1}^{n_C} \left(\frac{r_i - \hat{r}_i}{\sigma_i} \right)^2. \quad (\text{D10})$$

Here, $\nu = n_C - n_p$ describes the number of degrees of freedom of the fitting model. A χ_ν^2 of approximately one corresponds to the variance in the observations matching the variance of the residuals. On the other hand, $\chi_\nu^2 < 1$ indicates that the uncertainty of the model is too small to describe the data (indicating that the number of fitting parameters might be too large), whereas $\chi_\nu^2 > 1$ indicates that the model does not describe the data well enough.

For our fits, we were predominantly interested in the fitting parameter \hat{p}_{th} that indicates the threshold value of a certain configuration. We found that the regression model of Eq. (D2) worked relatively well in a close range around the true threshold value (see, e.g., Fig. 12). If using data over a larger range of p values, we would typically find fits with $\chi_\nu^2 > 1$. In those situations, we scaled up each σ_i with χ_ν , leading to

$$\text{Var}(\hat{\beta}) = \chi_\nu^2 (\hat{J}^T \Sigma^{-1} \hat{J})^{-1} \quad \text{if } \chi_\nu^2 > 1. \quad (\text{D11})$$

If one is in possession of $\text{Var}(\hat{\beta})$, the standard deviation of the least-squares fitting parameter value \hat{p}_{th} can be obtained from the square root of the corresponding diagonal element in $\text{Var}(\hat{\beta})$. One can then calculate confidence intervals for the fitting parameters by identifying with what factor the standard deviations should be multiplied to ensure the requested level of confidence. For this, we made use of the probability distribution f_{PD} of Student's t -distribution:

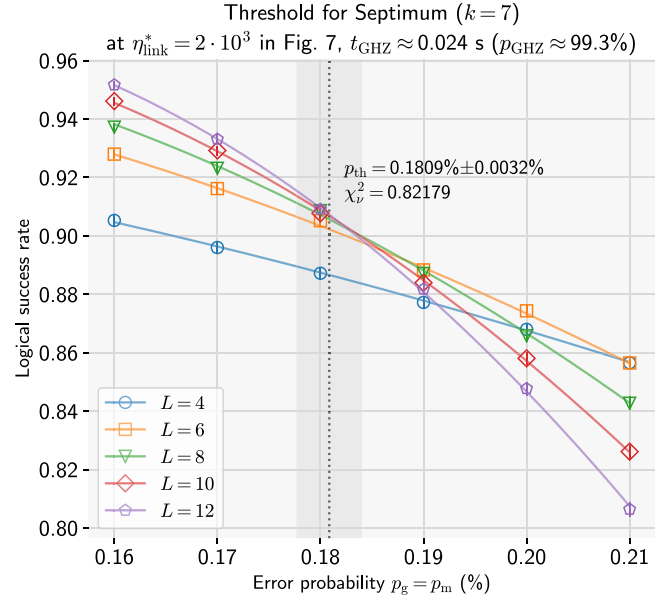


Fig. 12. Example of a threshold plot for the Septimum protocol, including a 95% confidence interval calculated with the method described in Sec. 3 of Appendix D.

$$f_{\text{PD}}(t_{\text{ci}}) = \Gamma'(\nu) \left(1 + \frac{t_{\text{ci}}^2}{\nu} \right)^{-(\nu+1)/2},$$

$$\Gamma'(\nu) = \begin{cases} \frac{(\nu-1)(\nu-3)\cdots 5 \cdot 3}{2\sqrt{\nu}(\nu-2)(\nu-4)\cdots 4 \cdot 2} & \text{if } \nu > 1 \text{ even,} \\ \frac{(\nu-1)(\nu-3)\cdots 4 \cdot 2}{\pi\sqrt{\nu}(\nu-2)(\nu-4)\cdots 5 \cdot 3} & \text{if } \nu > 1 \text{ odd.} \end{cases} \quad (\text{D12})$$

More specifically, confidence intervals can be calculated by finding the t_{ci} factor that corresponds to the confidence interval of choice for the distribution of Eq. (D12). In the plots in this paper, we show 95% confidence intervals. For large ν and a confidence interval of $I_{\text{ci}} = 95\%$, we have $t_{\text{ci}} \approx 1.96$. Smaller values of ν lead to t_{ci} values that are slightly bigger. In Fig. 12, we see an example of a threshold plot with a 95% confidence interval in the value found for the threshold fitting parameter.

APPENDIX E: SELECTION OF BEST-PERFORMING GHZ GENERATION PROTOCOLS

At the end of Sec. VB2, we discuss the Septimum protocol (depicted in Fig. 2): a GHZ generation protocol found with the dynamic program of Sec. IIB that gives rise to the highest thresholds for the simulation parameters used in several segments of Figs. 6–8. In this appendix, we identify four additional GHZ generation protocols that perform the best in multiple segments of the figures in Sec. VB: the protocols Sextimum, Decimum, Undecum, and Duodecum. We depict the timed binary trees of these four protocols in Fig. 13 and provide more information on their performance in Table II.

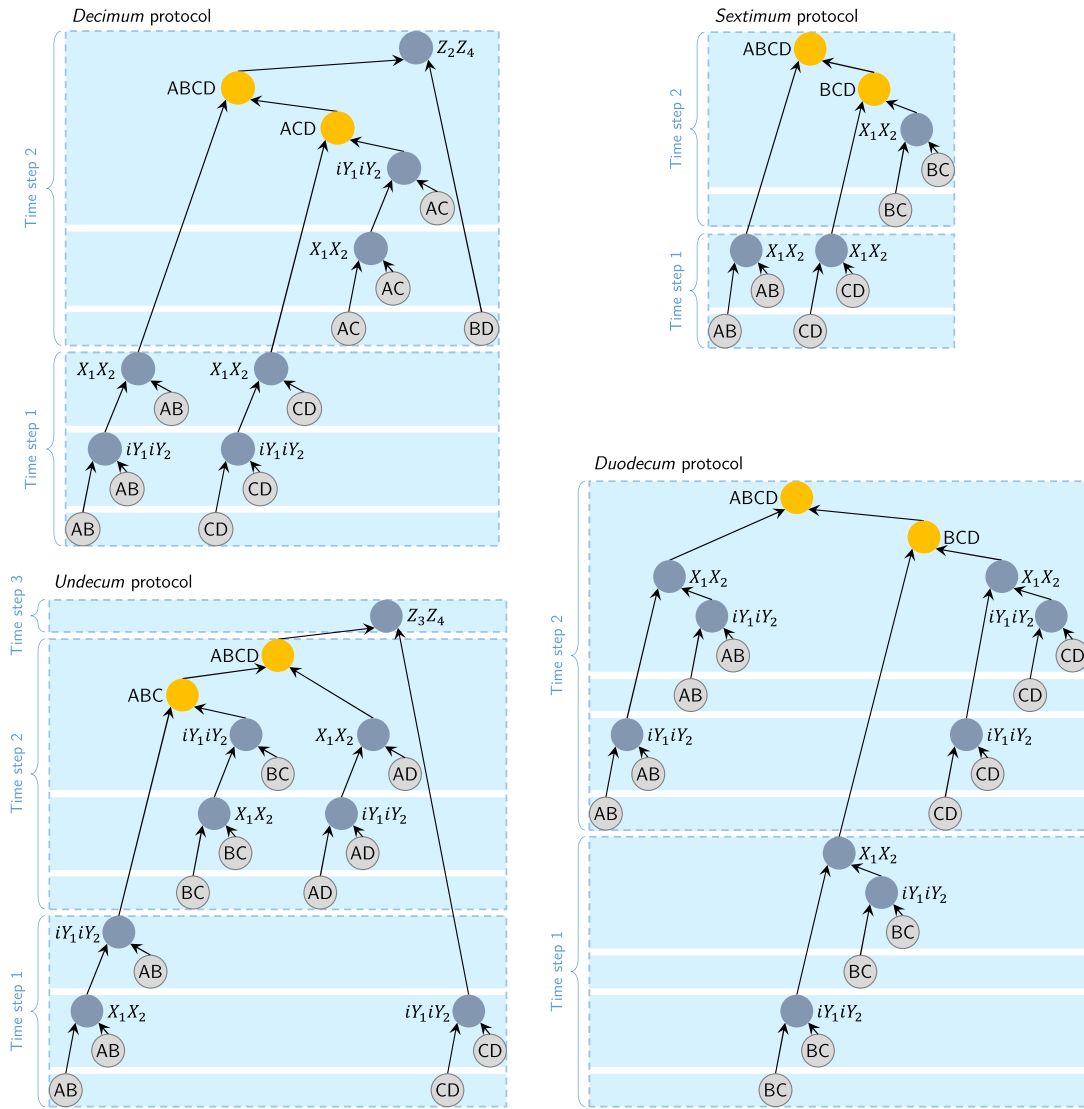


FIG. 13. Timed binary trees of a selection of best-performing protocols in the Bell pair fidelity and the link efficiency sensitivity studies of Sec. VB. All four protocols are found with the dynamic program of Sec. IIB for the simulation model and parameters used in Sec. VB. Clarification on the notation can be found in Fig. 2. More information on these protocols can be found in Table II. The associated protocol recipes used with these protocols can be found in the repository of Ref. 57.

TABLE II. Details about the GHZ generation protocols depicted in Figs. 2 and 13. These protocols are found with the dynamic program of Sec. IIB. The numbers k and q denote the minimum number of Bell pairs and the maximum number of qubits per node required to generate the GHZ state, respectively. The last three columns of the table denote locations or ranges on the x -axes of Figs. 6–8 in which these protocols are either the best-performing protocol or one of the best-performing protocols.

	k	q	Figure 6	Figure 7	Figure 8
Sextimum	6	3		$[4 \times 10^2, 5 \times 10^2]$	4.7×10^2
Septimum	7	3	0.96	$[5 \times 10^2, 2 \times 10^3]$	$[6.3 \times 10^2, 1.1 \times 10^3]$
Decimum	10	3	$[0.9, 0.93]$	$[2 \times 10^3, 2 \times 10^5]$	2×10^3
Undecum	11	3	$[0.85, 0.9]$		
Duodecum	12	4	$[0.78, 0.82]$		

26 July 2024 12:54:07

REFERENCES

- ¹L. K. Grover, “Quantum teleportation,” *arXiv:quant-ph/9704012* (1997).
- ²J. I. Cirac, A. K. Ekert, S. F. Huelga, and C. Macchiavello, *Phys. Rev. A* **59**, 4249 (1999).
- ³R. Van Meter and S. J. Devitt, *Computer* **49**, 31 (2016).
- ⁴L. Jiang, J. M. Taylor, A. S. Sørensen, and M. D. Lukin, *Phys. Rev. A* **76**, 062323 (2007).
- ⁵R. Van Meter, T. D. Ladd, A. G. Fowler, and Y. Yamamoto, *Int. J. Quantum Inform.* **8**, 295 (2010).
- ⁶K. Fujii, T. Yamamoto, M. Koashi, and N. Imoto, “A distributed architecture for scalable quantum computation with realistically noisy devices,” *arXiv:1202.6588* (2012).
- ⁷Y. Li and S. C. Benjamin, *New J. Phys.* **14**, 093008 (2012).
- ⁸C. Monroe, R. Raussendorf, A. Ruthven, K. R. Brown, P. Maunz, L.-M. Duan, and J. Kim, *Phys. Rev. A* **89**, 022317 (2014).
- ⁹J. Ramette, J. Sinclair, N. P. Breuckmann, and V. Vuletić, “Fault-tolerant connection of error-corrected qubits with noisy links,” *arXiv:2302.01296* (2023).
- ¹⁰D. D. Awschalom, R. Hanson, J. Wrachtrup, and B. B. Zhou, *Nat. Photonics* **12**, 516 (2018).
- ¹¹H. Bernien, B. Hensen, W. Pfaff, G. Koolstra, M. S. Blok, L. Robledo, T. H. Taminiau, M. Markham, D. J. Twitchen *et al.*, *Nature* **497**, 86 (2013).
- ¹²T. H. Taminiau, J. Cramer, T. Van Der Sar, V. V. Dobrovitski, and R. Hanson, *Nat. Nanotechnol.* **9**, 171 (2014).
- ¹³D. Riedel, I. Söllner, B. J. Shields, S. Starosielec, P. Appel, E. Neu, P. Maletinsky, and R. J. Warburton, *Phys. Rev. X* **7**, 031040 (2017).
- ¹⁴J. Cramer, N. Kalb, M. A. Rol, B. Hensen, M. S. Blok, M. Markham, D. J. Twitchen, R. Hanson, and T. H. Taminiau, *Nat. Commun.* **7**, 11526 (2016).
- ¹⁵P. C. Humphreys, N. Kalb, J. P. J. Morits, R. N. Schouten, R. F. L. Vermeulen, D. J. Twitchen, M. Markham, and R. Hanson, *Nature* **558**, 268 (2018).
- ¹⁶C. E. Bradley, J. Randall, M. H. Aboeih, R. C. Berrevoets, M. J. Degen, M. A. Bakker, M. Markham, D. J. Twitchen, and T. H. Taminiau, *Phys. Rev. X* **9**, 031045 (2019).
- ¹⁷M. H. Aboeih, J. Randall, C. E. Bradley, H. P. Bartling, M. A. Bakker, M. J. Degen, M. Markham, D. J. Twitchen, and T. H. Taminiau, *Nature* **576**, 411 (2019).
- ¹⁸M. Ruf, M. Weaver, S. van Dam, and R. Hanson, *Phys. Rev. Appl.* **15**, 024049 (2021).
- ¹⁹M. Pompili, S. L. N. Hermans, S. Baier, H. K. C. Beukers, P. C. Humphreys, R. N. Schouten, R. F. L. Vermeulen, M. J. Tiggeleman, L. dos Santos Martins *et al.*, *Science* **372**, 259 (2021).
- ²⁰M. H. Aboeih, Y. Wang, J. Randall, S. J. H. Loenen, C. E. Bradley, M. Markham, D. J. Twitchen, B. M. Terhal, and T. H. Taminiau, *Nature* **606**, 884 (2022).
- ²¹C. E. Bradley, S. W. de Bone, P. F. W. Möller, S. Baier, M. J. Degen, S. J. H. Loenen, H. P. Bartling, M. Markham, D. J. Twitchen *et al.*, *npj Quantum Inf.* **8**, 122 (2022).
- ²²L. Orphal-Kobin, K. Unterguggenberger, T. Pregolato, N. Kemf, M. Matalla, R.-S. Unger, I. Ostermay, G. Pieplow, and T. Schröder, *Phys. Rev. X* **13**, 011042 (2023).
- ²³A. Sipahigil, R. E. Evans, D. D. Sukachev, M. J. Burek, J. Borregaard, M. K. Bhaskar, C. T. Nguyen, J. L. Pacheco, H. A. Atikian *et al.*, *Science* **354**, 847 (2016).
- ²⁴D. D. Sukachev, A. Sipahigil, C. T. Nguyen, M. K. Bhaskar, R. E. Evans, F. Jelezko, and M. D. Lukin, *Phys. Rev. Lett.* **119**, 223602 (2017).
- ²⁵C. T. Nguyen, D. D. Sukachev, M. K. Bhaskar, B. MacHielse, D. S. Levonian, E. N. Knall, P. Stroganov, C. Chia, M. J. Burek *et al.*, *Phys. Rev. B* **100**, 165428 (2019).
- ²⁶C. T. Nguyen, D. D. Sukachev, M. K. Bhaskar, B. MacHielse, D. S. Levonian, E. N. Knall, P. Stroganov, R. Riedinger, H. Park *et al.*, *Phys. Rev. Lett.* **123**, 183602 (2019).
- ²⁷C. M. Knaut, A. Suleymanzade, Y.-C. Wei, D. R. Assumpcao, P.-J. Stas, Y. Q. Huan, B. Machielse, E. N. Knall, M. Sutula *et al.*, “Entanglement of nanophotonic quantum memory nodes in a telecommunication network,” *arXiv:2310.01316* (2023).
- ²⁸T. Iwasaki, Y. Miyamoto, T. Taniguchi, P. Siyushev, M. H. Metsch, F. Jelezko, and M. Hatano, *Phys. Rev. Lett.* **119**, 253601 (2017).
- ²⁹A. E. Rugar, S. Aghaeimeibodi, D. Riedel, C. Dory, H. Lu, P. J. McQuade, Z.-X. Shen, N. A. Melosh, and J. Vučković, *Phys. Rev. X* **11**, 031021 (2021).
- ³⁰R. Debroux, C. P. Michaels, C. M. Purser, N. Wan, M. E. Trusheim, J. Arjona Martínez, R. A. Parker, A. M. Stramma, K. C. Chen *et al.*, *Phys. Rev. X* **11**, 041041 (2021).
- ³¹S. B. Bravyi and A. Y. Kitaev, “Quantum codes on a lattice with boundary,” *arXiv:quant-ph/9811052* (1998).
- ³²E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, *J. Math. Phys.* **43**, 4452 (2002).
- ³³D. Browne, *Lectures on Topological Codes and Quantum Computation* (University of Innsbruck, Innsbruck, Germany, 2014).
- ³⁴N. H. Nickerson, Y. Li, and S. C. Benjamin, *Nat. Commun.* **4**, 1756 (2013).
- ³⁵N. H. Nickerson, J. F. Fitzsimons, and S. C. Benjamin, *Phys. Rev. X* **4**, 041041 (2014).
- ³⁶A. Reiserer, N. Kalb, M. S. Blok, K. J. M. van Bemmelen, T. H. Taminiau, R. Hanson, D. J. Twitchen, and M. Markham, *Phys. Rev. X* **6**, 021040 (2016).
- ³⁷N. Kalb, A. A. Reiserer, P. C. Humphreys, J. J. Bakermans, S. J. Kamerling, N. H. Nickerson, S. C. Benjamin, D. J. Twitchen, M. Markham *et al.*, *Science* **356**, 928 (2017).
- ³⁸N. Kalb, P. C. Humphreys, J. J. Slim, and R. Hanson, *Phys. Rev. A* **97**, 062330 (2018).
- ³⁹B. Hensen, H. Bernien, A. E. Dréau, A. Reiserer, N. Kalb, M. S. Blok, J. Ruitenbergh, R. F. L. Vermeulen, R. N. Schouten *et al.*, *Nature* **526**, 682 (2015).
- ⁴⁰M. Muroo, M. B. Plenio, S. Popescu, V. Vedral, and P. L. Knight, *Phys. Rev. A* **57**, R4075 (1998).
- ⁴¹E. N. Maneva and J. A. Smolin, *Contemp. Math.* **305**, 203 (2002).
- ⁴²W. Dür, H. Aschauer, and H. J. Briegel, *Phys. Rev. Lett.* **91**, 107903 (2003).
- ⁴³H. Aschauer, W. Dür, and H.-J. Briegel, *Phys. Rev. A* **71**, 012319 (2005).
- ⁴⁴S. Glancy, E. Knill, and H. M. Vasconcelos, *Phys. Rev. A* **74**, 032319 (2006).
- ⁴⁵C. Kruszynska, A. Miyake, H. J. Briegel, and W. Dür, *Phys. Rev. A* **74**, 052316 (2006).
- ⁴⁶E. Hostens, J. Dehaene, and B. De Moor, *Phys. Rev. A* **73**, 042316 (2006).
- ⁴⁷E. Hostens, J. Dehaene, and B. De Moor, *Phys. Rev. A* **74**, 062318 (2006).
- ⁴⁸K. H. Ho and H. F. Chau, *Phys. Rev. A* **78**, 042329 (2008).
- ⁴⁹M. Huber and M. Plesch, *Phys. Rev. A* **83**, 062321 (2011).
- ⁵⁰V. V. Kuzmin, D. V. Vasilyev, N. Sangouard, W. Dür, and C. A. Muschik, *npj Quantum Inf.* **5**, 115 (2019).
- ⁵¹F. Riera-Sábat, P. Sekatski, A. Pirker, and W. Dür, *Phys. Rev. A* **104**, 012419 (2021).
- ⁵²S. Krastanov, A. S. de la Cerda, and P. Narang, *Phys. Rev. Res.* **3**, 033164 (2021).
- ⁵³N. Rengaswamy, N. Raveendran, A. Raina, and B. Vasić, *Quantum* **8**, 1233 (2024).
- ⁵⁴K. Goyal, A. McCauley, and R. Raussendorf, *Phys. Rev. A* **74**, 032318 (2006).
- ⁵⁵S. de Bone, R. Ouyang, K. Goodenough, and D. Elkouss, *IEEE Trans. Quantum Eng.* **1**, 4102710 (2020).
- ⁵⁶S. de Bone and D. Elkouss, *ACM SIGMETRICS Perform. Eval. Rev.* **51**, 81 (2023).
- ⁵⁷S. de Bone, P. Möller, and D. Elkouss (2024). “Data/software underlying the publication: Thresholds for the distributed surface code in the presence of memory decoherence,” 4TU.ResearchData. <https://doi.org/10.4121/708d4311-49b1-4ec2-b3cb-292d267df6be>
- ⁵⁸K. Fujii and K. Yamamoto, *Phys. Rev. A* **80**, 042308 (2009).
- ⁵⁹S. Krastanov, V. V. Albert, and L. Jiang, *Quantum* **3**, 123 (2019).
- ⁶⁰S. Jansen, K. Goodenough, S. de Bone, D. Gijswijt, and D. Elkouss, *Quantum* **6**, 715 (2022).
- ⁶¹K. Goodenough, S. de Bone, V. Addala, S. Krastanov, S. Jansen, D. Gijswijt, and D. Elkouss, *IEEE J. Sel. Areas Commun.* **42**(7), 1830–1849 (2024).
- ⁶²S. Hu, “Quasilinear time decoding algorithm for topological codes with high error threshold,” Master’s thesis (Delft University of Technology, Delft, The Netherlands, 2020).
- ⁶³S. Hu, see <https://github.com/watermarkhu/qsurface/tree/4e31ae0> for “OOP surface code simulations.”
- ⁶⁴N. Delfosse and N. H. Nickerson, *Quantum* **5**, 595 (2021).
- ⁶⁵D. S. Wang, A. G. Fowler, A. M. Stephens, and L. C. L. Hollenberg, “Threshold error rates for the toric and surface codes,” *arXiv:0905.0531* (2009).
- ⁶⁶M. M. Wilde, *Quantum Information Theory* (Cambridge University Press, 2013).

- ⁶⁷M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, 2000).
- ⁶⁸C. Cabrillo, J. I. Cirac, P. García-Fernández, and P. Zoller, *Phys. Rev. A* **59**, 1025 (1999).
- ⁶⁹S. D. Barrett and P. Kok, *Phys. Rev. A* **71**, 060310 (2005).
- ⁷⁰A. Dahlberg, M. Skrzypczyk, T. Coopmans, L. Wubben, F. Rozpędek, M. Pompili, A. Stolk, P. Pawelczak, R. Knegjens *et al.*, “A link layer protocol for quantum networks,” in *Proceedings of the ACM Special Interest Group on Data Communication, SIGCOMM '19, New York, NY* (Association for Computing Machinery, 2019), pp. 159–173.
- ⁷¹F. Rozpędek, R. Yehia, K. Goodenough, M. Ruf, P. C. Humphreys, R. Hanson, S. Wehner, and D. Elkouss, *Phys. Rev. A* **99**, 052330 (2019).
- ⁷²T. Coopmans, R. Knegjens, A. Dahlberg, D. Maier, L. Nijsten, J. de Oliveira Filho, M. Papendrecht, J. Rabbie, F. Rozpędek *et al.*, *Commun. Phys.* **4**, 164 (2021).
- ⁷³S. L. N. Hermans, M. Pompili, H. K. C. Beukers, S. Baier, J. Borregaard, and R. Hanson, *Nature* **605**, 663 (2022).
- ⁷⁴P. E. Barclay, K.-M. C. Fu, C. Santori, A. Faraon, and R. G. Beausoleil, *Phys. Rev. X* **1**, 011007 (2011).
- ⁷⁵L. Li, T. Schröder, E. H. Chen, M. Walsh, I. Bayn, J. Goldstein, O. Gaathon, M. E. Trusheim, M. Lu *et al.*, *Nat. Commun.* **6**, 6173 (2015).
- ⁷⁶J. Görlitz, D. Herrmann, G. Thiering, P. Fuchs, M. Gandil, T. Iwasaki, T. Taniguchi, M. Kieschnick, J. Meijer *et al.*, *New J. Phys.* **22**, 013048 (2020).
- ⁷⁷G. Moody, V. J. Sorger, D. J. Blumenthal, P. W. Juodawlkis, W. Loh, C. Sorace-Agaskar, A. E. Jones, K. C. Balram, J. C. F. Matthews *et al.*, *J. Phys.: Photonics* **4**, 012501 (2022).
- ⁷⁸Y.-I. Sohn, S. Meesala, B. Pingault, H. A. Atikian, J. Holzgrafe, M. Gündoğan, C. Stavrakas, M. J. Stanley, A. Sipahigil *et al.*, *Nat. Commun.* **9**, 2012 (2018).
- ⁷⁹P.-J. Stas, Y. Q. Huan, B. Machielse, E. N. Knall, A. Suleymanzade, B. Pingault, M. Sutula, S. W. Ding, C. M. Knaut *et al.*, *Science* **378**, 557 (2022).
- ⁸⁰H. K. C. Beukers, M. Pasini, H. Choi, D. Englund, R. Hanson, and J. Borregaard, “Tutorial: Remote entanglement protocols for stationary qubits with photonic interfaces,” [arXiv:2310.19878](https://arxiv.org/abs/2310.19878) (2023).
- ⁸¹S. Baier, C. E. Bradley, T. Middelburg, V. V. Dobrovitski, T. H. Taminiau, and R. Hanson, *Phys. Rev. Lett.* **125**, 193601 (2020).
- ⁸²K. Tiurev, P. L. Mirambell, M. B. Lauritzen, M. H. Appel, A. Tiranov, P. Lodahl, and A. S. Sørensen, *Phys. Rev. A* **104**, 052604 (2021).
- ⁸³P. L. Mirambell, “Fidelity characterization of spin-photon entangled states,” Master’s thesis (University of Copenhagen, Copenhagen, Denmark, 2019).
- ⁸⁴S. L. N. Hermans, M. Pompili, L. D. Santos Martins, A. R.-P. Montblanch, H. K. C. Beukers, S. Baier, J. Borregaard, and R. Hanson, *New J. Phys.* **25**, 013011 (2023).
- ⁸⁵G. Avis, F. F. da Silva, T. Coopmans, A. Dahlberg, H. Jirovská, D. Maier, J. Rabbie, A. Torres-Knoop, and S. Wehner, *npj Quantum Inf.* **9**, 100 (2023).
- ⁸⁶D. Greenbaum and Z. Dutton, *Quantum Sci. Technol.* **3**, 015007 (2017).
- ⁸⁷W. Dür, M. Hein, J. I. Cirac, and H.-J. Briegel, *Phys. Rev. A* **72**, 052326 (2005).
- ⁸⁸M. R. Geller and Z. Zhou, *Phys. Rev. A* **88**, 012314 (2013).
- ⁸⁹Z. Cai and S. C. Benjamin, *Sci. Rep.* **9**, 11281 (2019).
- ⁹⁰P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, Í. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, see <https://scipy.org/> for “SciPy: Fundamental algorithms for scientific computing in python.”
- ⁹¹C. Wang, J. Harrington, and J. Preskill, *Ann. Phys.* **303**, 31 (2003).
- ⁹²A. Ruckstuhl, Introduction to Nonlinear Regression (IDP Institut für Datenanalyse und Prozessdesign, ZHAW Zürcher Hochschule für Angewandte Wissenschaften, 2010).
- ⁹³D. Constales, G. S. Yablonsky, D. R. D’hooge, J. W. Thybaut, and G. B. Marin, “Experimental data analysis: Data processing and regression,” in *Advanced Data Analysis & Modelling in Chemical Engineering*, edited by D. Constales, G. S. Yablonsky, D. R. D’hooge, J. W. Thybaut, and G. B. Marin (Elsevier, Amsterdam, 2017), Chap. 9, pp. 285–306.