IEEE Transactions on Quantum Engineering

Received 9 November 2022; revised 6 February 2023; accepted 22 February 2023; date of publication 2 March 2023; date of current version 23 March 2023.

Digital Object Identifier 10.1109/TQE.2023.3251358

Hardness of Braided Quantum Circuit Optimization in the Surface Code

KUNIHIRO WASA[1], SHIN NISHIO[2,3,4], KOKI SUETSUGU[2], MICHAEL HANKS[5], ASHLEY STEPHENS[2], YU YOKOI[2], AND KAE NEMOTO[2,3,4]

[1]Faculty of Science and Engineering, Hosei University, Tokyo 184-8584, Japan
[2]National Institute of Informatics, Tokyo 101-8430, Japan
[3]Department of Informatics, School of Multidisciplinary Sciences, The Graduate University for Advanced Studies (SOKENDAI), Tokyo 101-8430, Japan
[4]Quantum Information Science and Technology Unit, Okinawa Institute of Science and Technology Graduate University, Okinawa 904-0495, Japan
[5]QOLS, Blackett Laboratory, Imperial College London, London SW7 2AZ, U.K.

Corresponding author: Kunihiro Wasa (e-mail: wasa@hosei.ac.jp).

This work was supported in part by the Japan Society for the Promotion of Science KAKENHI under Grant JP21H04880, Grant JP19K20350, Grant JP20H05793, Grant JP22H03549, and Grant JP22J20882, in part by the MEXT Quantum Leap Flagship Program (MEXT Q-LEAP) under Grant JPMXS0118069605, and in part by the JST Moonshot R&D program under Grant JPMJMS2061.

ABSTRACT Large-scale quantum information processing requires the use of quantum error-correcting codes to mitigate the effects of noise in quantum devices. Topological error-correcting codes, such as surface codes, are promising candidates, as they can be implemented using only local interactions in a 2-D array of physical qubits. Procedures, such as defect braiding and lattice surgery, can then be used to realize a fault-tolerant universal set of gates on the logical space of such topological codes. However, error correction also introduces a significant overhead in computation time, the number of physical qubits, and the number of physical gates. While optimizing fault-tolerant circuits to minimize this overhead is critical, the computational complexity of such optimization problems remains unknown. This ambiguity leaves room for doubt surrounding the most effective methods for compiling fault-tolerant circuits for a large-scale quantum computer. In this article, we show that the optimization of a special subset of braided quantum circuits is NP-hard by a polynomial-time reduction of the optimization problem into a specific problem called PlanarRectilinear3SAT.

INDEX TERMS Braiding circuits, computational complexity, fault-tolerant quantum computation (FTQC), NP-complete, PlanarRectilinear3SAT, quantum circuit optimization, surface code.

I. INTRODUCTION

Quantum information processing has many applications, including quantum computation [1], [2], [3] and quantum communication [4], [5]. Implementations of noisy intermediate scale quantum systems already exist, but large-scale quantum information processing is required for the majority of proposed applications. Noise in these systems is an obstacle to this, and quantum error-correcting codes (QECCs) are the most promising methods to address this issue. QECCs provide a way of detecting and correcting errors in quantum systems by storing quantum information in a redundant system [6]. The ability to perform reliable computation in the presence of errors is called fault-tolerant quantum computation (FTQC) [7]. Topological stabilizer codes, such as surface codes [8], are good candidates for implementing FTQC due to their high thresholds and low-weight local stabilizers [9], [10].

However, many additional resources are required to implement fault-tolerant error correction, including in the case of surface codes due to their low encoding rate [11], [12], [13]. Furthermore, fault-tolerant quantum circuits are longer than unencoded ones since it is only possible to implement a finite set of gates fault-tolerantly within a given code and circuit depth increases significantly when continuous quantum gates are decomposed into gates from such a set [14]. For universal quantum computation, it is necessary to implement a set of quantum gates called a universal gate set [15], [16], [17], [18], [19], [20], [21], which must contain at least one quantum gate for multiple qubits. Transversal implementations of these gates in 2-D topological codes require nonlocal

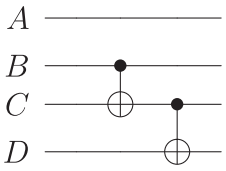

FIGURE 1. Simple example of a quantum circuit with CX gates.

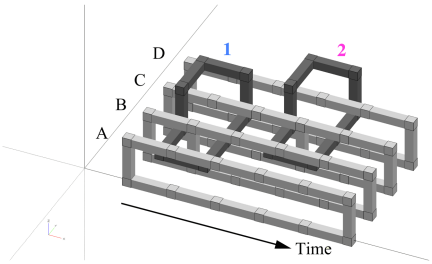

FIGURE 2. Defect braiding operation that implements the circuit of Fig. 1. Time flows from left to right, and the gray bars labeled with letters A to D are the defect pairs corresponding to the logical qubits. The black loops 1 and 2 correspond to the braiding operations (CX gates).

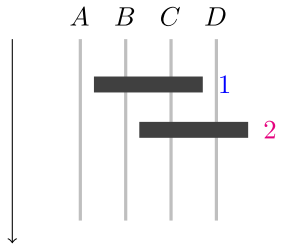

FIGURE 3. Simplified diagram (1-D braiding circuit representation) of Fig. 2. We ignore the direction of the CX gate for simplicity. Time flows from top to bottom. Vertical gray bars correspond to the logical qubits (pairs of punctures) and the black boxes correspond to the logical gate operations.

connectivity or higher dimensional architectures, and so alternative techniques, such as defect braiding [22] and lattice surgery [23], have been proposed. These operations can also take a long time to perform compared to performing logic on unencoded qubits, and therefore, optimization of these quantum circuits can significantly reduce the overhead for FTQC.

This project deals with the problem of optimizing defect braiding circuits to achieve FTQC using surface codes. Several practical optimizations for braiding circuits have been proposed [24], [25], [26], [27], but in order to achieve further optimizations, it is helpful to classify the optimization problem and prove the computational complexity. This allows us to estimate the cost of the optimization problem and define the requirements for the computer architecture. Our results show that the computational complexity for optimizing the depth of the defect braiding circuits by reordering is NP-hard, which matches previously known results for lattice surgery [28].

The surface code can be defined by placing qubits on the edges of a regular square lattice. Error detection is then achieved by the measurement of multiqubit Pauli operators localized at the nodes and across the faces of this lattice, which define the encoded subspace. Logical qubits are typically associated with lattice defects, such as boundaries or punctures, which correspond to regions of the lattice remaining unmeasured. In this work, we consider an encoding where a logical qubit is encoded with a pair of such punctures. Logical operations between pairs of logical qubits can then be performed via braiding of these pairs. Some restrictions are imposed on the braiding operations that can be performed, both by the topological nature of the surface code and the requirement to maintain a sufficient degree of error correction. For the purpose of our problem, the relevant restrictions are that defects must be separated by a minimum distance and that braiding operations do not commute. Note that in our analysis, we do not allow techniques, such as bridge compression [29], which may be used to obtain more compact braided quantum circuits in some circumstances.

A simple quantum circuit is presented in Fig. 1 as an example. Fig. 2 then shows a braided implementation as a 3-D space-time diagram, with a 1-D arrangement of logical qubits, while Fig. 3 shows a corresponding simplified diagram that we call the "1-D braiding circuit" representation. More generally, braiding circuits can be implemented in a high-dimensional representational space, and by imposing the constraints we obtain the 1-D braiding circuit. Although one might think the dimensional constraint in this representation is rather artificial, in some quantum computer architectures, the data logical qubits may be arranged in a quasi 1-D array. This may be due to the large distillation factory associated with this code implementation [30] or due to physical constraints in some architectures (for example, in quantum dot qubits in silicon and gallium arsenide, a quasi one-dimensional qubit arrangement may be required to accommodate the necessary control wiring).

In this article, we investigate the computational complexity of the problem of minimizing the depth of the 1-D braiding circuit. For this purpose, we formulate an optimization problem MinBraiding, which asks whether there exists an arrangement of qubits and gates that attains the required processing time.

II. PROBLEM DEFINITION

Let $[n] = \{1, 2, \ldots, n\}$ denote the set of n logical qubits. We are given a family $\mathcal{R}$ of controlled-NOT gates, where each gate is represented by $X \in \mathcal{R}$, a subset of $[n]$ corresponding to the set of qubits to which the gate is applied. We are also given a partial order $\succeq$ on $\mathcal{R}$, which represents the order of operators on the same qubit. Sometimes gate operators on the same qubit are commutative, but for simplicity we ignore such commutativity in this article. Therefore, two gates $X, X' \in \mathcal{R}$ are comparable if and only if they share a qubit. That is, we have $X \succeq X'$ or $X' \succeq X$ if $X \cap X' \neq \emptyset$. We write $X \succ X'$ if $X \succeq X'$ and $X \neq X'$. For a permutation $\pi : [n] \to [n]$ of logical

IEEE Transactions on Quantum Engineering

qubits, we say that gates $X_1, \ldots, X_\ell \in \mathcal{R}$ can be *arranged* into one row with respect to π if, for any distinct $i, j \in \{1, 2, \ldots, \ell\}$, either $\max\{\pi(x) \mid x \in X_i\} + 1 < \min\{\pi(x') \mid x' \in X_j\}$ or $\max\{\pi(x') \mid x' \in X_j\} + 1 < \min\{\pi(x) \mid x \in X_i\}$ holds. This condition means that any pair of gates in the same row have no overlap, and furthermore there is a margin between them.

We say that the family $\mathcal{R}$ of gates can be *packed* with height $h \in \mathbf{Z}_+$ according to a partial order $\succeq$ if there exists a pair (π, μ) of a permutation $\pi : [n] \to [n]$ and a function $\mu : \mathcal{R} \to [h]$ satisfying the following two conditions:

i) for each $i \in [h]$, the gates in $\{X \mid \mu(X) = i\}$ can be arranged into one row with respect to a permutation $\pi : [n] \to [n]$;
ii) for any two distinct gates $X, X' \in \mathcal{R}$, if $X \cap X' \neq \emptyset$ and $X \succ X'$, then $\mu(X) > \mu(X')$.

For π and μ satisfying (i) and (ii), a pair (π, μ) is called a *packing* of $(\mathcal{R}, \succeq)$ with height h. We call condition (i) the *horizontal condition*, and condition (ii) the *vertical condition*. For a packing (π, μ), we call $\mu(X)$ the *level of* $X \in \mathcal{R}$.

Now, we define our problem, which we call MinBraiding.

Problem 1 (MinBraiding): Given a set $\mathcal{R}$ of gates, a partial order $\succeq$ on $\mathcal{R}$, and a positive integer h, output `Yes` if $\mathcal{R}$ can be packed according to $\succeq$ with height at most h.

In particular, if there is a packing (π, μ) of $(\mathcal{R}, \succeq)$ with height h but no packing with height $h' < h$, then we call (π, μ) a *minimum packing*.

III. HARDNESS RESULT

In this section, we show that MinBraiding is NP-complete. Clearly, MinBraiding is in NP. Therefore, we devote this section to showing the polynomial-time reduction from PlanarRectilinear3SAT, which is explained as follows.

A 3-CNF Boolean formula ϕ consists of n variables $X = \{x_1, \ldots, x_n\}$ and m clauses $\mathcal{C} = \{c_1, \ldots, c_m\}$ such that each clause c_i contains at most three literals $\ell_i^1, \ell_i^2, \ell_i^3$. Here, $\ell_i^j \in \{x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n\}$ for $1 \leq i \leq m$ and $1 \leq j \leq 3$. Given a 3-CNF Boolean formula ϕ, 3SAT asks to decide whether there exists a satisfying assignment for ϕ. In the following, we consider the drawing of the graph corresponding to an instance of 3SAT in the plane. The *associated graph* $G(\phi) = (V(\phi), E(\phi))$ of ϕ is a graph such that $V(\phi) = X \cup \mathcal{C}, E(\phi) = \{\{x, c\} \in X \times \mathcal{C} \mid x \in c \tilde{} \text{or} \tilde{} \neg x \in c\}$. PLANAR3SAT is a restriction of 3SAT such that the associated graph of a given formula has a planar embedding; that is, we can draw the graph in the plane in such a way that no edges cross each other. It is shown in [31] and [32] that any instance of PLANAR3SAT has a *rectilinear representation*, where a rectilinear representation of ϕ is a planar drawing of the graph $G(\phi)$ satisfying the following conditions:

1) vertices in $G(\phi)$ are drawn by rectangles whose sides are parallel to axis;
2) rectangles corresponding to variables are drawn on a horizontal line;
3) edges in $E(\phi)$ are drawn by vertical segments;
4) edges do not cross each other.

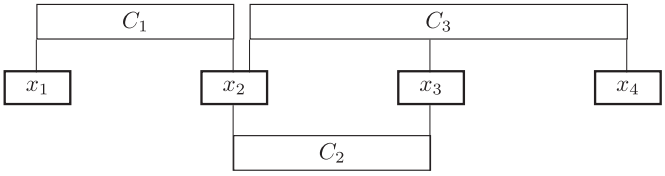

FIGURE 4. Example of the rectilinear representation of an instance $\phi_1 = C_1 \wedge C_2 \wedge C_3$ of PlanarRectilinear3SAT, where $C_1 = (\neg x_1 \vee x_2)$, $C_2 = (x_2 \vee x_3)$, $C_3 = (x_2 \vee x_3 \vee \neg x_4)$.

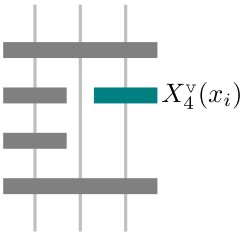

FIGURE 5. Variable gadgets. In this figure, the corresponding variable is assigned to true.

See Fig. 4 for an example of a rectilinear representation. The decision problem PlanarRectilinear3SAT is defined as follows: Given a 3-CNF formula ϕ and its rectilinear representation D, decide whether there exists a satisfying assignment for ϕ. PlanarRectilinear3SAT is known to be NP-complete [31].

To show the NP-completeness of MinBraiding, we construct the corresponding instance $(\mathcal{R}(\phi, D), \succeq (\phi, D), h(\phi, D))$ of MinBraiding for an instance (ϕ, D) of PlanarRectilinear3SAT so that $(\mathcal{R}(\phi, D), \succeq (\phi, D), h(\phi, D))$ is a YES-instance of MinBraiding if and only if (ϕ, D) is a YES-instance of PlanarRectilinear3SAT. For the construction, we introduce several gadgets.

Each gadget consists of two types of gates. A gate in a gadget is *fixed* if for any packing of the gadget with minimum height, the level of the gate is the same. Otherwise, a gate is said to be *movable*. Movable gates propagate the choice of the assignment of a variable. On the other hand, fixed gates restrict the placement of each movable gate. In each gadget, we call the set of fixed gates the *frame* of the gadget. In Figs. 5–10, the fixed gates are colored gray and the topmost gate is the largest gate. In addition, for figures of gadgets other than bending gadgets (see Fig. 9), we call green gates on the left side *inputs* and those of the right side *outputs*. Intuitively speaking, a gadget receives a variable assignment from the inputs and propagates the information to the outputs.

a) Variable gadgets: This gadget corresponds to a variable in ϕ. The variable gadget for a variable x_i consists of five gates $\mathcal{R}^{\mathrm{v}}(x_i) = \{X_1^{\mathrm{v}}(x_i), \ldots, X_5^{\mathrm{v}}(x_i)\}$ defined as follows: $X_1^{\mathrm{v}}(x_i) = X_5^{\mathrm{v}}(x_i) = \{1, 2, 3\}$, $X_2^{\mathrm{v}}(x_i) = X_3^{\mathrm{v}}(x_i) = \{1\}$, and $X_4^{\mathrm{v}}(x_i) = \{3\}$. Moreover, $X_1^{\mathrm{v}}(x_i) \prec X_b^{\mathrm{v}}(x_i)$ for $b > 1$,

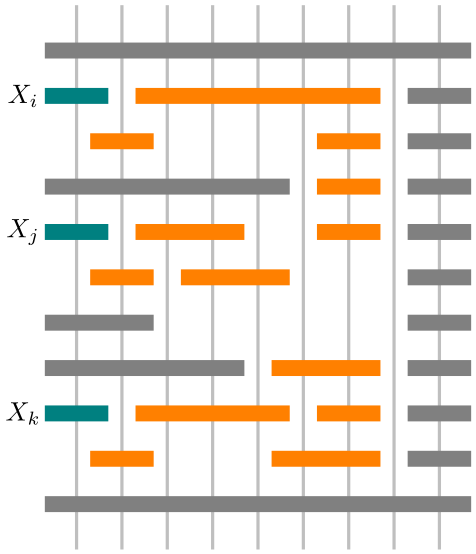

FIGURE 6. Example of a clause gadget. In this example, all variables appearing in the corresponding clause are assigned to true. If all variables are assigned to false, the height of this gadget is at least 12.

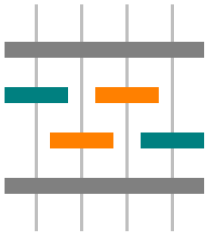

FIGURE 7. Not gadget. The level of the green gates is not the same.

$X_a^{\mathrm{v}}(x_i) \prec X_5^{\mathrm{v}}(x_i)$ for $a < 5$ and $X_2^{\mathrm{v}}(x_i) \prec X_3^{\mathrm{v}}(x_i)$. See Fig. 5 for the detail. Note that there is no input in variable gadgets. Clearly, the height of a minimum packing of $\mathcal{R}^{\mathrm{v}}(x_i)$ is four. In addition, we have only two minimum packings of $\mathcal{R}^{\mathrm{v}}(x_i)$; we can put $X_4^{\mathrm{v}}(x_i)$ at level 2 or 3. Note that the level is counted from bottom to top. Deciding the level of $X_4^{\mathrm{v}}(x_i)$ corresponds to an assignment of the variable x_i. If $X_4^{\mathrm{v}}(x_i)$ is on level 3, then we assign x_i to true. On the other hand, if $X_4^{\mathrm{v}}(x_i)$ is on level 2, then we assign x_i to false.

b) Clause gadgets: The clause gadget for a clause c_y consists of 29 gates. See Fig. 6 for the detail. The partial order on those gates are defined to be consistent with this figure. Note that there is no output in clause gadgets. In the gadget, three green gates X_i, X_j, and X_k correspond to variable assignments. For any minimum packing, there are no options for the level of each gray gate. On the other hand, we have two options for the level of X_i, X_j, and X_k; 9 and 10 for X_i, 6 and 7 for X_j, and 2 and 3 for X_k. If we, respectively, place X_i, X_j, and X_k on level 9, 6, and 2, then there is no packing of the gadget with height 11. Otherwise, there is a packing with height 11. This implies that at least one literal x_i, x_j, and x_k must be on the upper level. That is, to obtaining a minimum packing with height 11, c_y must be true.

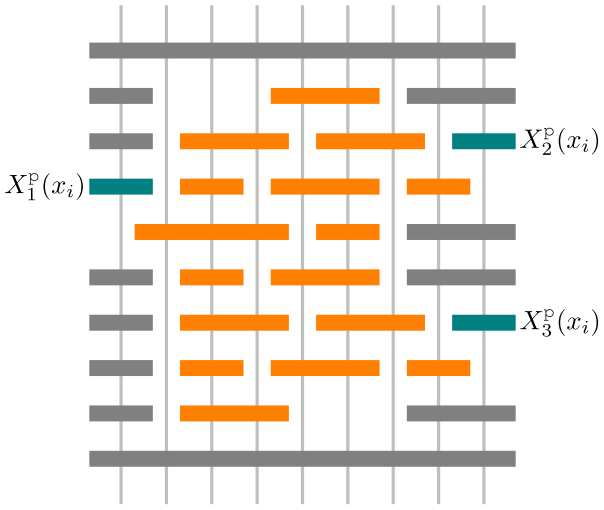

FIGURE 8. Copy gadget. If the left green gate is on the upper (resp. bottom) level, then the right green gates are on the upper (resp. bottom) level.

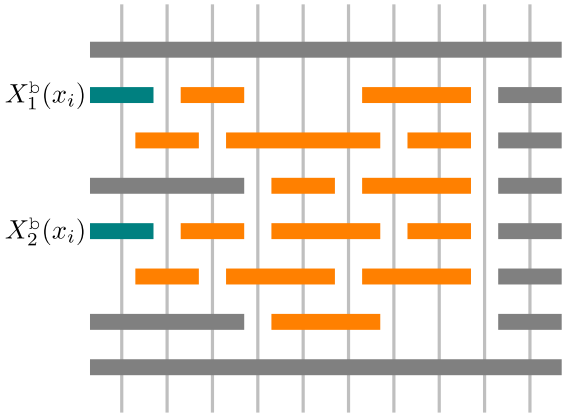

FIGURE 9. Bending gadget. If a green gate in the gadget is on the upper (resp. bottom) level, then the other green gate is also on the upper (resp. bottom) level. This gadget changes the direction of the propagation of the information for a variable assignment.

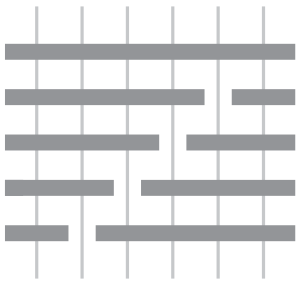

FIGURE 10. Gadget for fixing the ordering of six logical qubits.

c) NOT gadgets: To negate a variable, we use a not gadget. See Fig. 7, which represents gates constituting the gadget and a partial order defined on them. The height of the minimum packing of this gadget is clearly four. If the left green gate is on the upper level of possible choices, that is, assigning true

IEEE Transactions on Quantum Engineering

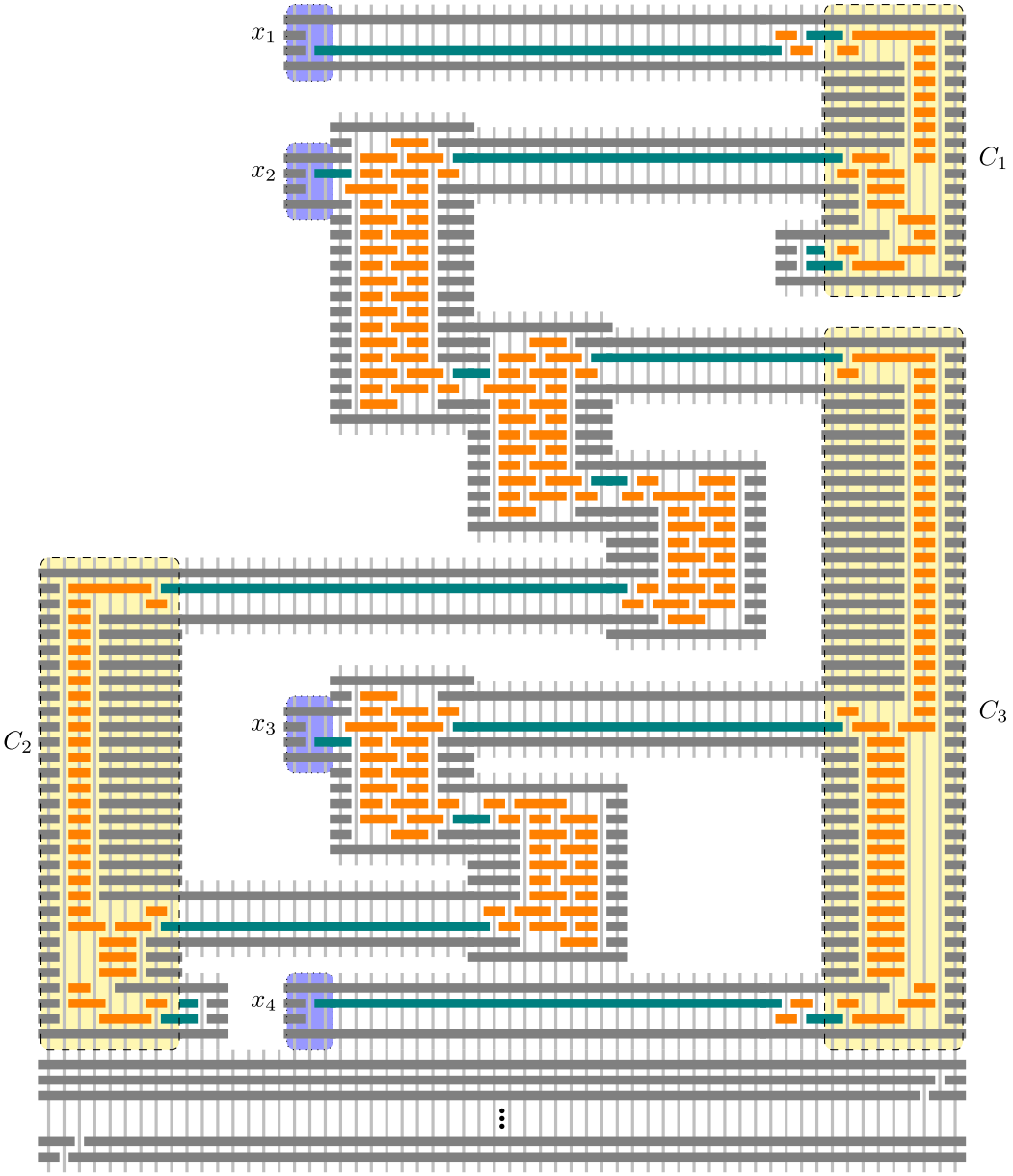

FIGURE 11. Corresponding instance of $\phi_1 = C_1 \wedge C_2 \wedge C_3$ in Fig. 4, where $C_1 = (\neg x_1 \vee x_2)$, $C_2 = (x_2 \vee x_3)$, $C_3 = (x_2 \vee x_3 \vee \neg x_4)$. The figure depicts the case when $(x_1, x_2, x_3, x_4) = (F, T, F, T)$. A blue area surrounded by a dotted line corresponds to a variable, and a yellow are surrounded by a dashed line corresponds to a clause. To propagate the information of the assignment of a variable gadget to a clause gadget, we use copying gadgets and bending gadgets, and extend these gadgets. Additionally, as C_1 and C_2 contain only two literals, we add dummy variable gadgets to force the third variable to be assigned false. C_1 and C_3 are adjacent to not gadgets since they contain negative literals.

to the corresponding variable, then the right green gate is on the lower level, that is, assigning false.

d) Copying gadgets: Copying gadgets are used to copy the assignment of a variable x_i. See Fig. 8, which represents gates constituting the gadget and a partial order defined on them. As before, we do not have options for the level of each gray gate. In addition, clearly, the height of a minimum packing of the gadget is 10. For any minimum packing of the gadget, if the green left gate $X_1^{\mathrm{p}}(x_i)$ is on the upper level of possible choices, then the other green gates $X_2^{\mathrm{p}}(x_i)$ and $X_3^{\mathrm{p}}(x_i)$ are also on the upper level. Similarly, if the left green gate is on the lower level of possible choices, then the other green gates are also on the lower level. These are the only two packings of the gadgets with height 10.

e) Bending gadgets: Bending gadgets are used to rotate the direction of propagation of the assignment of a variable x_i. See Fig. 9, which represents gates constituting the gadget and a partial order defined on them. The minimum height is eight. As before, for any minimum packing, we have only two options for the level of $X_1^{\mathrm{b}}(x_i)$; 6 or 7. If the left green gate $X_1^{\mathrm{b}}(x_i)$ is at level 7, then the other green gate $X_2^{\mathrm{b}}(x_i)$ is also at the upper level. Similarly, if the left green gate $X_1^{\mathrm{b}}(x_i)$

is on level 6, then the other green gate $X_2^{\flat}(x_i)$ is also on the lower level. Note that this gadget is utilized to propagate the assignment of a variable to a different level by concatenating two bending gadgets so that the new gadget is "S"-shaped.

f) Fixing gadget: To fix the left-right direction of logical qubits, we add several gates on the bottom of the instance as follows. See Fig. 10 for the case that the number of logical qubits is six. Gates of this gadget are smaller than any gates in other gadgets. Thus, in the packing of the instance, this gadget appears in the bottom. For n logical qubits, the minimum height of this gadget is $n-1$. For $i=1,\ldots,n-2$, at level i, the gadget contains two gates $X_\ell^{\mathfrak{f}}(i)=\{1,\ldots,i\}, X_r^{\mathfrak{f}}(i)=\{i+1,\ldots,n\}$. Any other ordering of logical qubits cannot achieve this minimum height.

g) Construction: Now, we construct the corresponding instance $(\mathcal{R}(\phi),\succeq(\phi),h(\phi))$ for an instance ϕ of PlanarRectilinear3SAT. We first rotate the rectilinear representation of PlanarRectilinear3SAT by 90° in a clockwise manner. Next, we replace each vertex in the representation with the frame of the corresponding gadget. Then, we connect the gadgets by using not gadgets, copying gadgets, and bending gadgets and fill the empty place other than the inside of each frame by gray gates. Note that, to properly connect these gadgets, we may slightly modify each gadget by inserting gray gates and orange gates, and extending green gates. Additionally, if a clause contains less than three literals, then we use a dummy variable gadget to force a unused literal to be assigned false. These modifications do not affect the mechanism of each gadget. Once the placement of frames are fixed, we can compute the possible minimum height $h(\phi)$ of a packing and the number of vertical lines. Moreover, we can give a partial order on gates according to the placement of frames. Fig. 11 shows an example of the corresponding instance of ϕ_1. Since the correctness is straightforward from the construction, the following theorem is established.

Theorem 1 ($\mathcal{R}(\phi),\succeq(\phi),h(\phi)$): is a Yes-instance of Min-Braiding if and only if ϕ is satisfiable.

We should note that we can define a "2-D braiding circuit" representation and the problem of minimizing the volume of the 2-D braiding circuit in a similar way to the 1-D braiding circuit representation. The 1-D representation is also applicable to quasi 2-D arrangements, such as a bilinear array, in some extent, however, the conditions of the margins and overlap for the optimization problem become more complex when one attempts to utilize quantum gates with a more complex shape or to place qubits in a 2-D or 3-D arrangement. While the complexity of this problem remains an open question, we conjecture that optimization of 2-D circuits and higher dimensional circuits is intractable as it is a generalization of the 1-D braiding circuit representation.

IV. CONCLUSION

This article considers the computational complexity of the problem that must be solved to optimize a quantum circuit by changing the qubit order when using defect braiding, one of the FTQC methods needed for large-scale quantum information processing using the surface code. We proved that this problem is NP-complete by performing a polynomial-time reduction to PlanarRectilinear3SAT, which corresponds to the equivalent result for lattice surgery [28]. Taken together, these results suggest that optimizing FTQC circuits is costly and that heuristic methods, including approximation algorithms, are essential. As such, the development of such heuristic methods will be important for the realization of FTQC using the surface code, and for the comparison of other error-correction codes and fault-tolerant procedures to determine the most practical and lowest-overhead route to large-scale quantum information processing.

ACKNOWLEDGMENT

Shin Nishio would like to thank Thomas Scruby for useful discussions.

REFERENCES

[1] D. P. DiVincenzo, "Quantum computation," *Science*, vol. 270, no. 5234, pp. 255–261, 1995, doi: 10.1126/science.270.5234.255.

[2] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th Annu. ACM Symp. Theory Comput.*, 1996, pp. 212–219, doi: 10.1145/237814.237866.

[3] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th Annu. Symp. Found. Comput. Sci.*, 1994, pp. 124–134, doi: 10.1109/SFCS.1994.365700.

[4] H.-J. Briegel, W. Dür, J. I. Cirac, and P. Zoller, "Quantum repeaters: The role of imperfect local operations in quantum communication," *Phys. Rev. Lett.*, vol. 81, no. 26, 1998, Art. no. 5932, doi: 10.1103/PhysRevLett.81.5932.

[5] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," 2020, *arXiv:2003.06557*, doi: 10.48550/arXiv.2003.06557.

[6] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Phys. Rev. A*, vol. 52, no. 4, 1995, Art. no. R2493, doi: 10.1103/PhysRevA.52.R2493.

[7] D. Gottesman, "Stabilizer codes and quantum error correction," Ph.D. dissertations, California Inst. of Technol., Pasadena, CA, USA, 1997, doi: 10.48550/arXiv.quant-ph/9705052.

[8] A. Y. Kitaev, "Quantum error correction with imperfect gates," in *Quantum Communication, Computing, and Measurement*. Boston, MA, USA: Springer, 1997, pp. 181–188, doi: 10.1007/978-1-4615-5923-8_19.

[9] R. Raussendorf and J. Harrington, "Fault-tolerant quantum computation with high threshold in two dimensions," *Phys. Rev. Lett.*, vol. 98, no. 19, 2007, Art. no. 190504, doi: 10.1103/PhysRevLett.98.190504.

[10] A. M. Stephens, "Fault-tolerant thresholds for quantum error correction with the surface code," *Phys. Rev. A*, vol. 89, no. 2, 2014, Art. no. 022321, doi: 10.1103/PhysRevA.89.022321.

[11] S. Bravyi, D. Poulin, and B. Terhal, "Tradeoffs for reliable quantum information storage in 2D systems," *Phys. Rev. Lett.*, vol. 104, no. 5, 2010, Art. no. 050503, doi: 10.1103/PhysRevLett.104.050503.

[12] N. C. Jones et al., "Layered architecture for quantum computing," *Phys. Rev. X*, vol. 2, no. 3, 2012, Art. no. 031007, doi: 10.1103/PhysRevX.2.031007.

[13] C. Gidney and M. Nielsen, "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits," *Quantum*, vol. 5, 2021, Art. no. 433, doi: 10.22331/q-2021-04-15-433.

[14] C. M. Dawson and M. A. Nielsen, "The Solovay-Kitaev algorithm," *Quantum Inf. Comput.*, vol. 6, pp. 81–95, 2006, doi: 10.5555/2011679.2011685.

[15] D. E. Deutsch, "Quantum computational networks," *Proc. Roy. Soc. London. A. Math. Phys. Sci.*, vol. 425, no. 1868, pp. 73–90, 1989, doi: 10.1098/rspa.1989.0099.

[16] A. Barenco et al., "Elementary gates for quantum computation," *Phys. Rev. A*, vol. 52, no. 5, 1995, Art. no. 3457, doi: 10.1103/PhysRevA.52.3457.

[17] D. P. DiVincenzo, "Two-bit gates are universal for quantum computation," *Phys. Rev. A*, vol. 51, no. 2, 1995, Art. no. 1015, doi: 10.1103/PhysRevA.51.1015.

[18] A. Barenco, "A universal two-bit gate for quantum computation," *Proc. Roy. Soc. London. Ser. A: Math. Phys. Sci.*, vol. 449, no. 1937, pp. 679–683, 1995, doi: 10.1098/rspa.1995.0066.

[19] S. Lloyd, "Almost any quantum logic gate is universal," *Phys. Rev. Lett.*, vol. 75, no. 2, 1995, Art. no. 346, doi: 10.1103/PhysRevLett.75.346.

[20] D. E. Deutsch, A. Barenco, and A. Ekert, "Universality in quantum computation," *Proc. Roy. Soc. London. Ser. A: Math. Phys. Sci.*, vol. 449, no. 1937, pp. 669–677, 1995, doi: 10.1098/rspa.1995.0065.

[21] M. Reck, A. Zeilinger, H. J. Bernstein, and P. Bertani, "Experimental realization of any discrete unitary operator," *Phys. Rev. Lett.*, vol. 73, no. 1, 1994, Art. no. 58, doi: 10.1103/PhysRevLett.73.58.

[22] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Phys. Rev. A*, vol. 86, no. 3, 2012, Art. no. 032324, doi: 10.1103/PhysRevA.86.032324.

[23] C. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter, "Surface code quantum computing by lattice surgery," *New J. Phys.*, vol. 14, no. 12, 2012, Art. no. 123011, doi: 10.1088/1367-2630/14/12/123011.

[24] A. Paetznick and A. G. Fowler, "Quantum circuit optimization by topological compaction in the surface code," 2013, *arXiv:1304.2807*, doi: 10.48550/arXiv.1304.2807.

[25] A. Paler, I. Polian, K. Nemoto, and S. J. Devitt, "Fault-tolerant, high-level quantum circuits: Form, compilation and description," *Quantum Sci. Technol.*, vol. 2, no. 2, 2017, Art. no. 025003, doi: 10.1088/2058-9565/aa66eb.

[26] F. Hua et al., "Autobraid: A framework for enabling efficient surface code communication in quantum computing," in *Proc. IEEE/ACM 54th Annu. Int. Symp. Microarchitect.*, 2021, pp. 925–936, doi: 10.1145/3466752.3480072.

[27] M. Hanks, M. P. Estarellas, W. J. Munro, and K. Nemoto, "Effective compression of quantum braided circuits aided by ZX-calculus," *Phys. Rev. X*, vol. 10, no. 4, 2020, Art. no. 041030, doi: 10.1103/PhysRevX.10.041030.

[28] D. Herr, F. Nori, and S. J. Devitt, "Optimization of lattice surgery is NP-hard," *Npj Quantum Inf.*, vol. 3, no. 1, pp. 1–5, 2017, doi: 10.1038/s41534-017-0035-1.

[29] A. G. Fowler and S. J. Devitt, "A bridge to lower overhead quantum computation," 2012, *arXiv:1209.0510*, doi: 10.48550/arXiv.1209.0510.

[30] S. J. Devitt, A. M. Stephens, W. J. Munro, and K. Nemoto, "Requirements for fault-tolerant factoring on an atom-optics quantum computer," *Nature Commun.*, vol. 4, no. 1, pp. 1–8, 2013, doi: 10.1038/ncomms3524.

[31] D. E. Knuth and A. Raghunathan, "The problem of compatible representatives," *SIAM J. Discr. Math.*, vol. 5, pp. 422–427, Aug. 1992, doi: 10.1137/0405033.

[32] D. Lichtenstein, "Planar formulae and their uses," *SIAM J. Comput.*, vol. 11, pp. 329–343, May 1982, doi: 10.1137/0211025.