Untangling Surface Codes: Bridging Braids and Lattice Surgery

Alexandru Paler[1, *]

[1] *Aalto University, 02150 Espoo, Finland*

We present a systematic method for translating fault-tolerant quantum circuits between their braiding and lattice surgery (LS) representations within the surface code. Our approach employs the ZX calculus to establish an equivalence between these two paradigms, enabling verified, bidirectional conversion of arbitrary surface-code-level circuits. We show that both braiding and LS operations can be uniformly expressed as compositions of multibody measurements and demonstrate that the Raussendorf compression rule encompasses all known braid and bridge optimizations. We also introduce a novel CNOT circuit with LS. Our framework provides a foundation for the automated verification, compilation, and benchmarking of large-scale surface code computations, advancing toward a unified formal language for topological quantum computation.

I. INTRODUCTION

Fault-tolerant quantum computation relies on error-correcting codes to protect logical operations from noise. Among these, the surface code is the most practical and widely studied architecture for large-scale quantum computers. Logical operations in the surface code can be implemented using two distinct paradigms: *braiding*, where topological defects are moved around each other, and *lattice surgery* (LS), where logical qubit patches are merged and split through multibody measurements.

Although both paradigms are known to be computationally equivalent, they differ significantly in geometry, scheduling, and resource cost. Existing comparisons between them have been limited to specific circuits, and no general method exists for automatically translating or verifying equivalence between arbitrary braided and LS implementations. This lack of a formal translation layer hinders automated benchmarking, compiler validation, and optimisation across topological architectures.

In this work we present the first systematic framework for translating between braided and LS representations. Our approach leverages the ZX calculus [1, 2] to capture both paradigms within a unified, circuit-agnostic semantics. The resulting translation rules allow high-level surface-code circuits to be converted, verified, and optimised automatically.

Beyond practical compiler applications, the framework also establishes a foundation for a formal language of surface-code computation, connecting ZX-calculus semantics with topological models of fault tolerance. A fully categorical treatment of these equivalences is deferred to future work, in which the underlying algebraic structures will be developed in detail.

A. Contributions

This work introduces a formally validated framework for translating fault-tolerant quantum computations between their *braided* and *lattice surgery* (LS) surface-code representations. The framework is based on circuit identities and the ZX calculus, which provides a semantics-preserving bridge between the two paradigms. To the best of our knowledge, this is the first systematic and bidirectional translation between arbitrary braided, LS, and quantum circuit descriptions.

1. We define explicit translation maps between braided and LS circuits and show that they are semantics-preserving.

2. We establish a systematic procedure for extracting high-level quantum circuits from arbitrary braided geometries, enabling automatic verification and benchmarking.

3. We show that the Raussendorf compression rule subsumes all known braid and bridge optimisation techniques, including the bridging operation originally introduced in [3].

4. We prove that bridges can be introduced and removed freely without altering the logical semantics, thereby enabling new classes of circuit transformations and simplifications.

5. We present a new LS circuit for implementing the logical CNOT and prove its equivalence, via ZX-calculus reasoning, to their braided counterparts.

While the constructions presented here can be expressed in a fully formal mathematical framework – e.g., by defining explicit categories for braided, lattice-surgery, and ZX representations, and proving functorial equivalences between them—we have intentionally opted for an intuitive exposition. A completely formal treatment, while possible, would considerably increase the technical density and reduce accessibility for readers primarily interested in the computational implications. We therefore defer a rigorous categorical and proof-theoretic development of these translations to a follow-up paper, which will include a detailed formalisation and correctness proofs.

* alexandru.paler@aalto.fi

B. Background

We will introduce the minimum definitions necessary to present the translation method between braids, LS and circuits. We will consider that arbitrary quantum circuits have been compiled into the ICM form [4, 5], which is built exclusively of: a) CNOTs, b) high fidelity non-Clifford states encoded in ancilla logical qubits, and c) logical qubit measurements. ICM circuits are fundamentally the way in which surface code computations are implemented. The lowest level surface code operations can be used only for CNOT gates, while universality is achieved through magic state ancilla and measurements.

Braided and LS circuits are executed in a measurement-based manner, and there is a Pauli frame that has to be tracked at the circuit's logical layer. The work of [6] has a complete treatment of how to track the corrections generated by the LS operations. In the following, we will not discuss the Pauli frame or how classically controlled gates have to be introduced into the quantum circuits or the ZX diagrams. We refer the reader to [6, 7] for how to track and update the Pauli frame.

1. *Lattice Surgery*

In LS the two-dimensional lattice of physical qubits is partitioned into patches. Each patch has four boundaries, two for each operator type: logical X (bit) and logical Z (phase).

In LS, patches are operated by merging and splitting them along the boundaries. The merge and split operations correspond to multibody measurement circuits (Fig. 1). The same operations are also used for implementing the CNOT gate (Fig. 2). Multibody measurements are operations in which multiple patches are merged and split simultaneously. The connection between lattice surgery and the ZX calculus has been formalised by [6], and one of the earliest works that has used this connection for expressing lattice surgery compilation tasks was [8].

The first CNOT implementation was presented in [9], but, as shown in [6], there are more possible constructions. Computational universality with LS is achieved by injection and distillation procedures which are mapped to logical qubit patches and CNOT gates implemented through merge and split operations.

2. *Braids*

Computations implemented by braiding are achieved after punching holes (also called defects) into the physical qubit lattice. The braids are the result of performing the following abstractions: a) filling out the holes with a solid; b) representing the time evolution (resulting from the discretised movement across the lattice) of the holes by strands; c) removing everything around the strands,

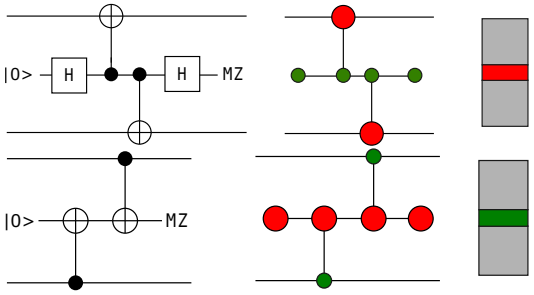

FIG. 1. The two-body measurement circuits used throughout this manuscript. The XX-measurement (top) uses an ancilla qubit initialized in $|+\rangle$ and measured in the X-basis and has an equivalent ZX-diagram where two red spiders are touching the measured wires. The XX-measurement can be abstracted in the language of lattice surgery (LS) as two grey patch interacting along their red boundaries. The ZZ-measurement (bottom) uses an ancilla initialized in $|0\rangle$ and measured in the Z-basis. The LS diagram uses two grey patches interacted along their green boundary.

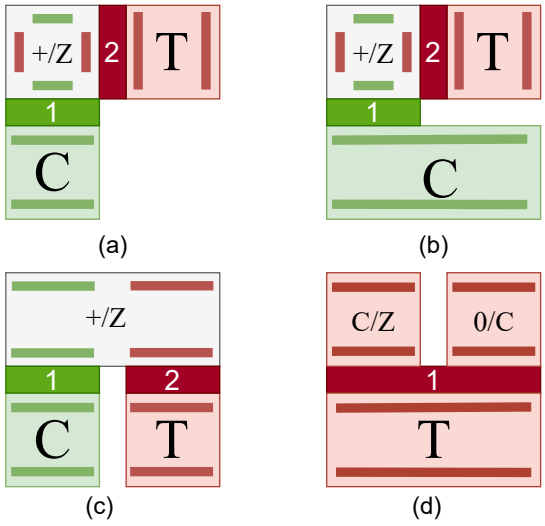

FIG. 2. There are different ways of implementing CNOTs with surface code lattice surgery. In the following the color of the square patches is indicative of the patch boundary that is about to be used. The $+/Z$ notation means that the patch will be initialized in $|+\rangle$ and measured in the Z-basis. The indices on the two-body measurements (cf. Fig. 1) indicates their order. For example, in (a) the green ZZ-measurements is performed first, and the red XX-measurement is executed afterwards. (a) The canonical patch arrangement for LS CNOT [9]; (b) extending one of the patches (e.g. the control C) does not change the computation; (c) after extending and rotating the ancilla patch such that it has two operator boundaries on the same side [10]; (d) the ancilla is now initialised in $|0\rangle$ and not measured, the control patch is measured in the Z basis after all three patches performed a three-body red X-measurement. The correctness of (d) is shown using ZX calculus in Fig. 9.

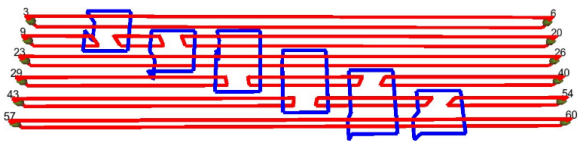

FIG. 3. Elements of braided circuits [4]: pairs of defects (here primal, red) form logical qubits and are braided with loops of the opposite type (here dual, blue) for performing CNOTs. The circuit herein does not include bridges.

including the lattice.

In the canonical compilation of braided circuits [11], logical qubits are defined by pairs of strands. There are two types of strands: primal and dual. Accordingly, depending on the strand types, there are primal and dual logical qubits.

Strands can be either *braided* (Figures 3 and 4) or *bridged* (Fig. 4a) . Braiding works for strands of the opposite type (primal-dual), while bridging is available for strands of the same type (primal-primal, dual-dual).

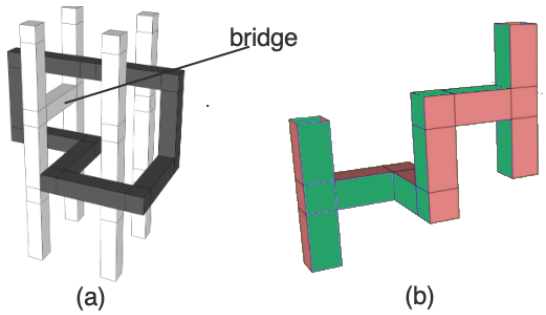

FIG. 4. CNOT implementations: (a) using braided surface codes [3] between two primary logical qubits (here white formed by pairs of defects) mediated by a dual loop (here dark grey); (b) lattice surgery. Time flows in both diagrams from the bottom to the top. The diagrams are visually extremely similar. However, this is just the result of the white *bridge* that is connecting the pair of vertically parallel defects in (a). In (b), we used green and red and to highlight the different correlation surfaces spanned along the patches.

3. *Correlation Surfaces*

Correlation surfaces have been initially proposed in the cluster-state implementation of the surface code [12]. A first method for automatically constructing correlation surfaces has been proposed in [13], and then a construction using Boolean variables in [11]. A SAT based approach to constructing LS correlation surfaces has been described by [14].

Definition I.1. A correlation surface as the set of qubits tracked along the path connecting the input logical operator with the output logical operator of a surface code computation.

In the case of braided computations, there are two types of correlation surfaces: tubes and sheets. A *tube* is formed by the logical operators encircling the strands. A *sheet* is formed by the logical operators spanned between pairs of strands (e.g. in a logical qubit formed by a pair of strands). The interpretation of the tubes and sheets, in terms of logical X and logical Z operators, alternates between the types of logical qubits (primal or dual).

Figs. 5 and6 illustrate the correct ways in which tubes and sheets can be connected: a) tubes connect to a sheet, when the supporting strands are of opposite type; b) tubes connect to tubes, or sheets to sheets, when the supporting strands are of the same type.

The construction from Fig. 5a exists wherever two strands are *bridged* [3]: the tubes have to be connected in order for the correlation surface to be well-defined. At the same time, the sheets spanned along the bridged strands will be overlapping (Fig. 5c) such that there are three valid constructions. The construction from Fig. 5b corresponds to a braid: tubes will be connected to sheets. Bridging and unbridging will be discussed in Sec. II E.

Definition I.2. In the case of lattice surgery (LS), we define correlation surfaces as the sheets covering the boundaries of the patches (when one considers these in 3D in space-time diagram-like manner like in Fig. 4).

In LS, we do not consider a dual lattice space, such that there are no primal or dual logical qubits. For this reason, the sheet correlation surfaces will correspond to logical X and logical Z operators at the boundaries of the patches. Due to the way how logical operators are initialised in braided computations, well-defined sheets exist only in loops, and well-defined tubes only in non-loops (i.e. chains).

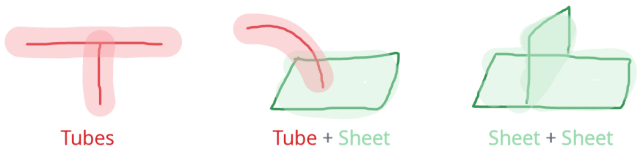

FIG. 5. Connecting correlation surfaces. Here we show how primal-red can be connected to dual-green. The reverse are also valid. a) two tubes of the same type; b) a tube and a sheet of opposite type; c) two sheets of the same type. Herein, the red strings correspond to the white defects from and the green loops to the grey defect loop from Fig. 4.

4. *Trivial Loops*

Our focus are loops, and we start by discussing a single loop surrounding a pair of strands (Fig. 7). The blue loop in Fig. 7b is just an identity from a computational point of view, but as shown in the Appendix it cannot be simply removed. That ring is an identity only if the two strands belong to the same logical qubit, otherwise, if the strands are from distinct qubits, the ring is a parity measurement.

FIG. 6. Two loops of opposite types have to be braided more than once, because otherwise the correlation surfaces would not be correctly connected with each other. This observation is a criteria for checking the correctness of braided circuits. (a) incorrect connection of a red tube to the green sheet, because the tube would need to connect twice at the black braiding point; (b) the red tube can connect correctly to the green sheet, because now there two braiding points and each of these points is touched only once by the tube.

Definition I.3. A loop is trivial when it acts like an identity operator (Fig. 7b).

Definition I.4. A loop is not trivial when it acts like an multibody operator (Fig. 7c).

Trivial loops are a necessity when optimising braided structures, because it allows eliminating other loops and strands (Sec. II F). We can introduce trivial loops using the ZX calculus (Fig. 16) and can decompose multibody measurements into pairs of non-trivial rings such as the one from Fig. 7c.

A ring around a loop of an opposite type (Fig. 7c) is a ZZ- or XX-measurement, depending on the type of the loop in the middle. The sheet of the outer loop can be used to measure the sign of the logical operators that are correlated along both tubes. This is effectively, in terms of LS, a multibody measurement between a qubit and an ancilla.

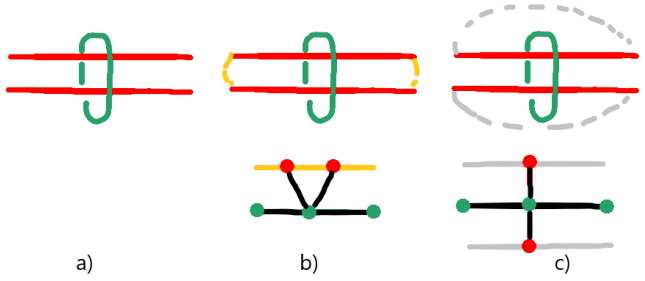

FIG. 7. a) A loop around a pair of opposite type strands; b) if the pair belongs to the same loop then it represents an identity as shown by the ZX diagram; c) if the strands belong to different loops, the result is a multibody measurement, in this example an XX.

II. METHODS

We offer a new perspective on LS and braided circuits, and will explain why the resemblance between bridged braided circuits and LS is more than a coincidence. We use the ZZ and XX measurement circuits (parity circuits, Fig. 1) and the ZX calculus to extract universal quantum circuits in order to show the equivalence between braided and LS diagrams (Algorithm 8). In the process of doing this, we introduce two reduced instructions sets: `LSIS` for LS circuits, and `RBIS` for braided circuits.

Our goal is to interpret braided and LS structures as ICM circuits. In an ICM circuit, the CNOT gate is used for achieving computational universality[4]: the T gates are performed by teleportation. We will not discuss the exact implementation of the T gates, or how non-Clifford states are encoded (i.e. injection) and their fidelity is increased (i.e. distillation), because those procedures use circuits formed exclusively of CNOT gates.

We use the CNOT gate to mediate the translation between LS and braids:

- braids-LS direction: we use the ZX interpretation of the braided structures to extract LS multibody measurements;
- LS-braids direction: LS computations are expressed as sequences of multibody measurements, and we will express the measurements using the circuits from Fig. 1 and then translate those circuits into braids.

For both translation directions, LS-braids and braids-LS, we assume, for the beginning, that the braided structures are formed entirely of simple loops and strands. Afterwards, we will show how to decompose arbitrarily complex braided structures into simple loops and strands.

We will follow a bottom-up approach.

- first, we introduce the straightforward bidirectional translation between LS and braids without bridges;
- second, we analyze, from the perspective of correlation surfaces, how multibody measurements are implemented in braided and LS circuits;
- finaly, we present methods for the automatic translation of arbitrary braided diagrams into LS and vice versa.

A. The braids-LS translation: Loops and Spiders

The braid-LS translation begins by selecting the inputs and the outputs of the braided structure. In general, a braided circuit is the diagrammatic equivalent of an ICM circuit, and the order in which the CNOT gates are executed is determined by the topological sort from diagram inputs towards outputs.

Strands and loops are the building blocks of braided circuits. However, without loss of generality, in order to keep the LS-braids interpretation straightforward, we use only loops for representing logical qubits. This approach is slightly different from the original [12], where pairs of strands were the support of a logical qubit: loops resulted depending on the initialisation and measurement basis.

Require: A surface-code circuit C, represented either as a braided diagram B or as a lattice-surgery (LS) diagram L
Ensure: The corresponding circuit in the alternate representation
if C is a braided circuit B **then**
  Identify all **loops** and **bridges** in B
  Decompose B into *elementary loops* using unbridging rules
  for all braids between loops of opposite type (primal–dual) **do**
    Replace each braid with an **XX** or **ZZ** multibody measurement
    Express the measurement as LS merge/split operations on logical patches
  end for
  Apply ZX simplifications to contract trivial spiders
  Return the LS circuit
else
  Identify all **merge/split** operations and their measurement bases in L
  for all multibody measurements **do**
    Express the measurement as a ZX diagram
    Translate each two-body ZX interaction into a braid between opposite-type strands
  end for
  Introduce bridges for same-type strand interactions if needed
  Apply loop compression (Raussendorf rule) to reduce redundant braids
  Return the braided circuit
end if

FIG. 8. Bidirectional Translation Between Braided and Lattice Surgery Circuits

For our purpose, logical qubits initialised into $|0\rangle$ will be primal loops, for $|+\rangle$ the loops will be duals. For these reasons: a) primal loops will correspond to logical qubits initialised and measured in the Z basis; b) dual loops correspond to logical qubits initialised and measured in the X basis.

There are similarities between loops and spiders. For example, it has been observed by [15] that the double strand approach to defining braided logical qubits is similar to constructing GHZ states. The above example highlights the applicability of the ZX calculus for braided circuit analysis [15]. This has been also very recently observed in [16], but that paper does not discuss bridges, because these are not effectively braids.

The main difference between ZX wires and strands is that the first do not have a type (colour), while the latter do. For this reason, it is useful to imagine that, for a spider with arity larger or equal to two, two of the spider's legs correspond to the qubit-wire in the circuit diagram, while the other legs are for CNOTs. This observation uses the fact that a ZX diagram is a bipartite graph: there are no adjacent same-colour spiders, and, even if such a situation would exist, the spiders can be merged (Fig. 20). In other words, a ZX spider can be translated into a braided loop, by: 1) having two of the spider's legs as a loop; 2) turning the remaining legs into braids (CNOTs).

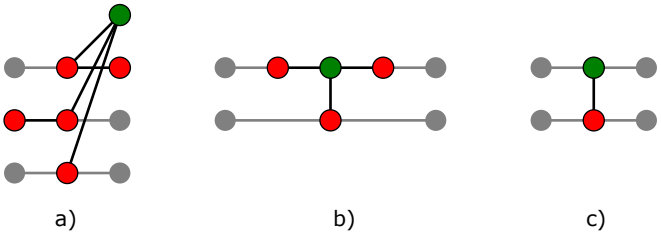

FIG. 9. The correctness of Fig. 2d): a) the three patches are horizontal wires connected by the ZX diagram of a 3-body measurement; b) after contracting all red spiders; c) eliminating the trivial red spiders results in the ZX diagram of a CNOT.

B. The LS-braids translation: Rotations and Braids

The LS-braids translation is based on the observation that the multibody measurements for the LS CNOT are structurally (and functionally) similar to the braiding pattern of the primal-primal CNOT. Fig. 9d highlights the connection between the LS CNOT and the braid CNOT: both are the result of implementing the same three-body X measurement and then measuring the control qubit/patch while maintaining the qubit/patch of the ancilla initialised in the Z basis.

Similarly to how [10] introduced a set of rules in order to express arbitrary LS computations, we define a `LS reduced instruction set (LSIS)` in order to translate between LS and braids:

- *initialisation into* $|0\rangle$ *or* $|+\rangle$ and *measurement into X or Z basis* (e.g. the $0/Z$ notation on the ancilla patches from Fig. 5);
- *extend and contract* – a patch is extended to the next empty patch space, or contracted to leave an empty patch space (e.g. Fig. 2b);
- *rotate* – the orientation of the boundaries is changed; for example, the X boundary is moved from the top to the right (e.g. the effect on the T-patch when comparing Fig. 2a and Fig. 2c);
- *merge and split* – necessary for implementing multibody measurements (e.g. the operations from .

The LS patch rotation operation is switching the position of the logical operators. The rotation [9, 10] is used to position logical operators such that the necessary multibody measurement can be implemented. In [17], a rotation was assumed to take twice the code distance, such that rotations had to be performed at half distance – a procedure assumed to be *sufficient* for practical purposes. In [10] a rotation takes three times the code distance. Rotations could also be implemented with

Hadamard gates, which in the case of the surface code are transversal [9]. This would reduce the depth of the compiled circuit. If sufficent hardware resources are available, it is not necessary to rotate a patch in order to access its logical operator boundary [10, 18].

When braiding, this LS rotation is equivalent to changing the type of the correlation surface corresponding to the logical operator: a tube becomes a sheet, and a sheet is turned into a tube (Fig. 5).

Proposition II.1. *A LS rotation corresponds to a braid between strands of different types.*

We will now illustrate how Fig. 2a can be automatically translated into the standard functionally equivalent braided circuit [12]. We start from the LS CNOT circuit from Fig. 2a. The latter includes two multi-body measurements, XX and ZZ, which correspond to the circuits from Fig. 1. The step-by-step constructive translation of a CNOT from LS to braids is illustrated in Fig. 10.

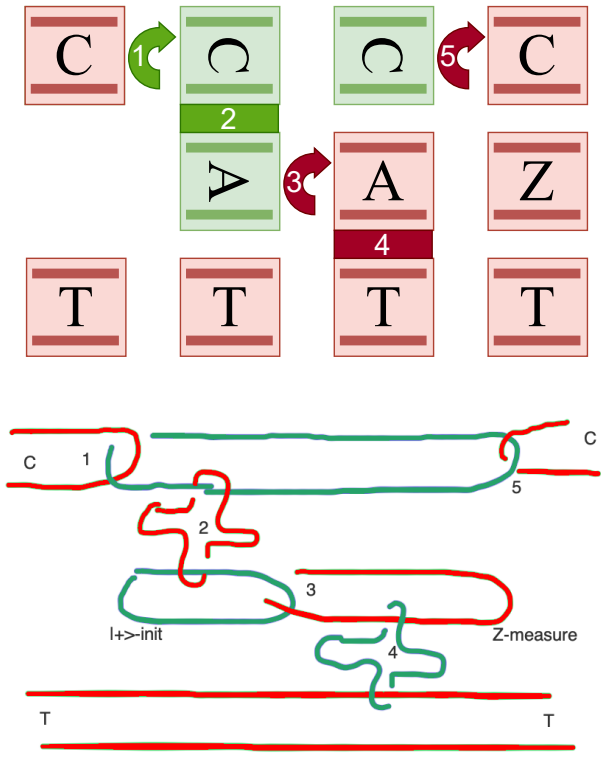

FIG. 10. Parallel comparison between the LS CNOT and the braided CNOT using XX and ZZ measurements. The control and dual qubits are primals, the ancilla because it is initialised in $|+\rangle$ it is directly a dual. The rotation arrows correspond to braiding a primal with a dual. This one-to-one comparison between circuits is practically a visual translation of `LSIS` (Sec. II B) to `RBIS` (Sec. II E).

C. Multibody Measurements and the Bialgebra Rule

We translated a LS CNOT into braids, and used the ZX language for expressing two-body measurements. We will use two-body measurements to compose multi-body measurements compatible with LS and braids. The two-body measurements are effectively the key of our translation methods.

In braided circuits, the simplest multibody measurement patterns appear, for example, when a single loop is braided with multiple other loops of the opposite type. In LS, multibody measurements appear each time that merge and split operations are executed (e.g. Fig. 2).

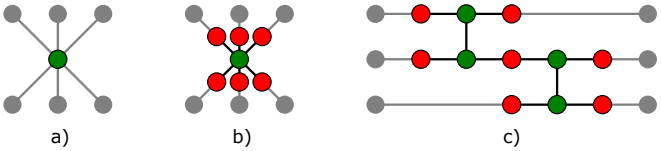

FIG. 11. Manipulating spiders: a) a green spider connecting multiple wires; b) trivial red spiders are introduced; c) the green spider is split, another trivial red spider is included between the two greens, and the wires are arranged to look like a quantum circuit.

Proposition II.2. *In the ZX calculus, spiders are manifestations of a multibody measurement [1].*

Starting from Fig. 11a, we use the ZX calculus to split a spider into a sequence of two-body measurements illustrated in Fig. 11c: the pairwise green spiders are two-body measurements which can be easily translated into braids if the red spiders are not trivially eliminated. If the red spiders are trivially removed, the green spiders can be easily translated into LS merge and split operations.

In Fig. 12 we derive the braided primal-primal CNOT step-by-step by starting from the ZX diagram of a CNOT. We chose a seemingly more complex route to arrive at the multibody measurement implemented by the red spider: applied the bialgebra rule. Moreover, the bialgebra rule shows that it is possible to achieve a braided CNOT using four loops (Fig. 14).

We can generalise the construction of multibody measurements. From the perspective of LS/braids, these measurements appear by using an ancilla qubit (Fig. 14) initialised in a given basis, interacting it with patch-operators/strands of the opposite type, and measuring the ancilla in the opposite basis.

D. Spiders and Bridges

Multibody measurements can be analysed also by their effect on correlation surfaces. The latter are helpful to determine a method to decompose seemingly complex braided structures into elementary braided loops. In the

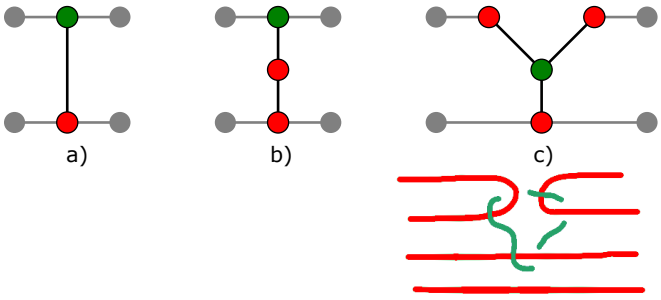

FIG. 12. An example of how to use the bialgebra rule to build a multibody measurement mediated by a green loop. Effectively, this is a demonstration of how a CNOT is implemented in braided diagrams: a) a CNOT in the ZX calculus; b) after introducing a trivial red spider; c) after applying the bialgebra rule and translating to braids.

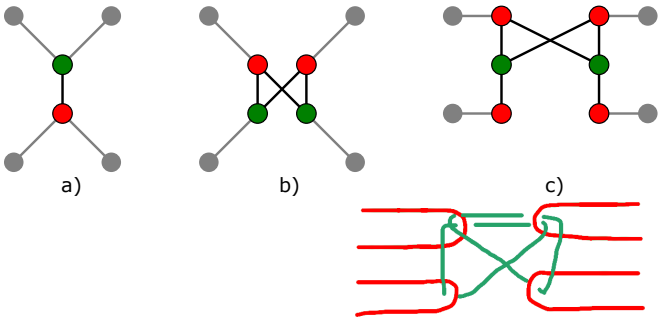

FIG. 13. A slightly more complex application of the bialgebra rule that results in two 3-body measurements: a) a CNOT; b) applying the bialgebra rule; c) after rearranging the wires and translating into braids after inserting trivial green spiders on the lower wires.

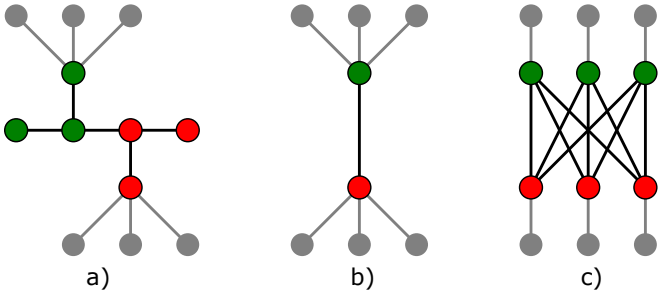

FIG. 14. Two spiders of opposite colours can be interpreted as sharing an ancilla initialised and measured in opposite basis: a) the ancilla wire is highlighted; b) the general input to the bialgebra rule; c) after applying the bialgebra rule. The last subfigure can be transformed, for example, into braids in a similar way to Fig. 13.

translation from Sec. II A, we used only loops in braided circuits. Due to this decision, when using ZX diagrams to interpret braids, the colour of the spiders will play a significant role with respect to where and how the correlation surfaces will be interacted.

Proposition II.3. *In braided circuits, every two-body measurement can be replaced with a bridge.*

Assuming, without loss of generality, that all the wires of a ZX diagram are translated into braids as primal-red loops:

- ZZ-measurements are used to interact sheets (logical Z operators);
- XX-measurements are used to interact tubes (logical X operators).

Although XX- and ZZ-measurements are used for different tasks, internally the circuits perform the same thing: they correlate sheets, the difference being only the space where this operation takes place (primal vs. dual).

Proposition II.4. *Bridging does not change the computation and does not affect the functional correctness of the diagrams.*

From the perspective of the ZX calculus, the type of the spider determines the type of the bridge. Thus, bridges between strands of the same type/color can be arbitrarily introduced into the braided structure. We show this by case distinction:

1. The bridge is between strands belonging to the same loop (Fig. 15b). This is a special case in the step by step derivation of the Raussendorf rule: 1) only two primal loops are present; 2) no dual moustaches; 3) the central strand is not removed. None of the above change the computation after bridging;
2. The bridge is between two distinct loops (Fig. 15c). If there is no chain of braids connecting the loops then the entire structure consists of two disjoint sheets and their trivial product that results after the bridge product – effectively there is no tube than enforces the sheet to be connected to it (cf. Figures 5 and 6)

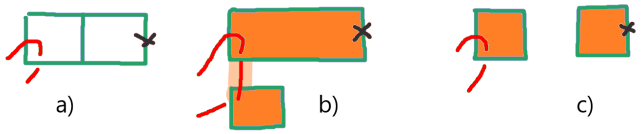

FIG. 15. Example of loops after unbridging: a) the original 3D structure, b) disconnecting loops along braided defect (red); c) disconnecting disjoint loops (i.e. no other defect is braiding them). The black cross marks that the associated loop may continue and be braided with another dual.

E. Unbridging: Decomposing Complex Braided Structures

LS circuits are straightforward to translate into to braided circuits. Depending on the type of LS multibody measurement the corresponding braided structure can be automatically generated. Translating a braided

circuit to LS does not seem as easy, because braided circuits might include bridges. We need independent, not bridged [3], loops (e.g. Fig. 5a). If there are bridged loops in a diagram, the bridges representation have to be undone, meaning that each bridged structure is decomposed into elementary loops.

Definition II.5. We call *unbridging* the procedure by which a braided circuit is transformed such that all correlation surface combinations of the type Fig. 5a and Fig. 5c are transformed into the type from Fig. 5b.

Our goal is to deconstruct arbitrary (e.g. Fig. 17), but valid, braided representations into loops - elementary building blocks compatible with LS. To this end, we introduce, similarly to Sec. II B, a `Reduced Braids Instruction Set (RBIS)`, which consists of the braiding relation and diagrammatic rewrites which leave the computation unchanged (effectively, identity operations):

1. *braid* strings of the same type (the identity logical operation) or of opposite type (the logical CNOT);
2. *introduce* trivial loops;
3. *remove* trivial loops (e.g. in Fig. 19, the loop in the middle and the strand that connects the two junctions);
4. *bridge* (e.g. in Fig. 19b, the primal loops that are braided with the large dual);
5. *unbridge* (e.g. in Fig. 19c, the dual strands that look like a moustache).

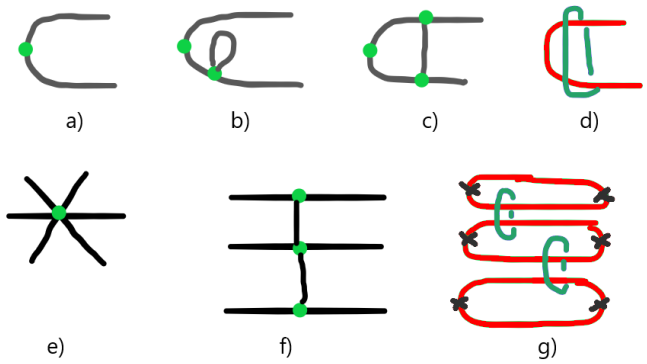

FIG. 16. a) GHZ state with a green spider; b) another green spider connected to a loop can be trivially introduced; c) the loop is decomposed such that the two legs are connected; d) the braiding equivalent of the previous ZX diagram; e) a green spider with multiple legs; f) pairing the legs of the spider and decomposing the green spider; g) one the multiple possible braiding equivalents of the previous ZX diagram includes two loops braided with three logical-qubit-loops.

Proposition II.6. *Unbridging does not affect the correctness of the diagrams.*

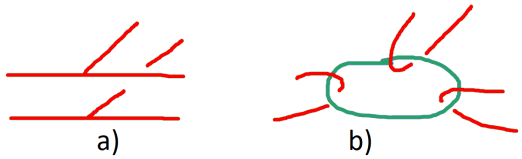

FIG. 17. Structures like in a) can be decomposed into a ring of opposite type (e.g. dual) that is braided like in b) with loops. The correctness of this construction is shown by enumerating the supported correlation surfaces.

Bridges are manifestations of the underlying CNOTs used in multibody measurement circuits. Consequently, unbridging can be used at each strand junction of a braided diagram. Trivial rings do not need to be included: these can be contracted through the ZX calculus (Fig. 16).

F. The Raussendorf rule: The Instruction Set of Braids

The Raussendorf rule (Fig. 18) encompasses all known operations about how correlation surfaces can be constructed and transformed from one into another (Fig. 7). In Fig. 19 we derive step-by-step the Raussendorf rule.

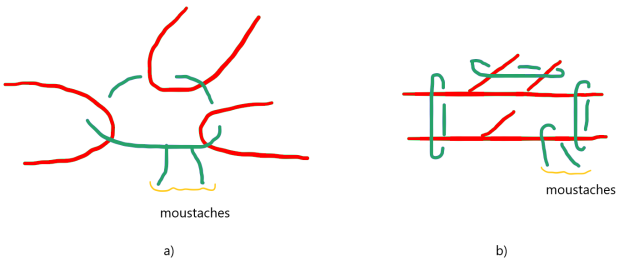

FIG. 18. The compression rule as presented by Raussendorf: a) multiple loops braided with a loop of opposite type and the opposite type loop has some moustaches; b) the result of the rule is that the primals are bridged, the moustaches are unbridged and trivial dual loops are introduced.

In terms of braided structures the Raussendorf rule can be show using the steps from Fig. 19.

In Fig. 22 we used the ZX calculus and rely heavily on the bialgebra rule to perform the individual steps. We will not show the generality of the transformations formally: we will show it only for three loops. The diagrams are too complicated when more loops are involved. We have chosen to arrange the ZX diagram to look very similar to the braided structure.

III. DISCUSSION AND CONCLUSION

This work provides the first step towards a unified formal language for surface-code computation. We introduced two reduced instruction sets, `LSIS` for lattice

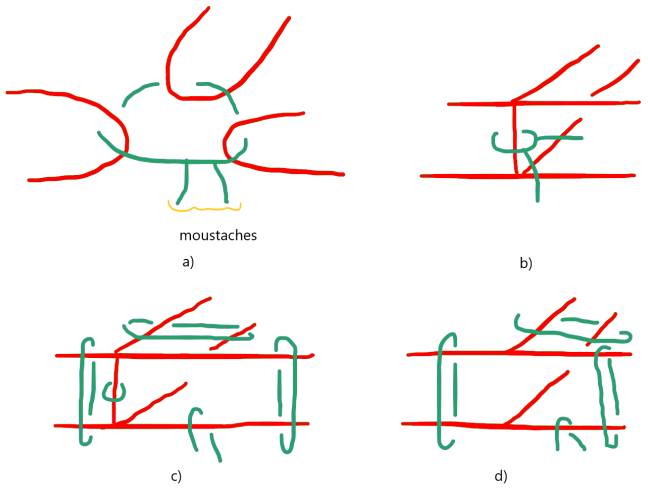

FIG. 19. Step-by-step derivation of the Raussendorf rule: a) multiple loops braided with a loop of opposite type and the opposite type loop has some moustaches; b) because the blue loop correlates the green tubes, these can be bridged; c) trivial blue loops are introduced and the moustaches can be unbridged from the central blue loop; d) the central loop and the braided strand can be removed because their role in terms of correlation surfaces has been taken over by the (previously trivial) blue loops.

surgery computations, and RBIS for braided computations, and showed how to translated between these instructions.

We have shown that arbitrary braided structure can be easily decomposed into loops of two types and that the resulting structures have the form of multibody measurements. These measurements can be translated into LS operations. From the other direction, LS operations can be easily transformed into simple braided loops. The entire process is intermediated by the ZX calculus.

In order to show that braided structures can be decomposed into simple loops, we showed that the Raussendorf rule is in fact a compressed representation of RBIS. The Raussendorf rule includes bridging and unbridging at the same time, as well as the introduction and removal of trivial loops.

ACKNOWLEDGEMENTS

I am grateful for the discussions with Austin Fowler and Simon Devitt. I developed most of these ideas and translations, and large parts of the manuscript, after the long discussions I had with Austin about compilation and decoding during my enjoyable Los Angeles 2020 pandemic stay. This research was developed in part with two Google Faculty Awards in 2019 and 2020 and a Fulbright scholarship in 2020. This research was enabled in part with funding from the Defense Advanced Research Projects Agency [under the Quantum Benchmarking (QB) program under award no. HR00112230007 and HR001121S0026 contracts], and was supported by the QuantERA grant EQUIP through the Academy of Finland, decision number 352188. The views, opinions and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

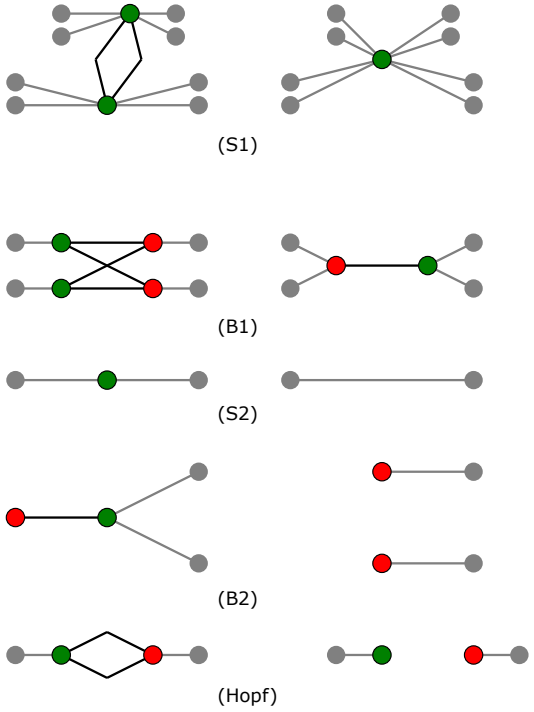

FIG. 20. The transformation rules of the ZX calculus [1]. The rules are valid also when spiders are switching their color. For example, the (S2) rule says that non-phased spider can be removed or trivially added to a wire irrespective of its color (green or blue).

IV. APPENDIX

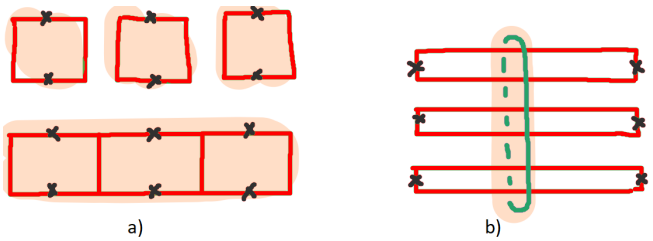

FIG. 21. Multibody measurements with braids while leaving the computation unchanged: a) bridging allows the joint measurement of sheets; b) the dual ring allows the joint measurement of tubes.

We discussed how to bridge and unbridge braided geometries. In Fig. 15 we illustrated how in some unbridged interpretations of braids, the sheets are similar to products of logical operators. Measuring for example the large

sheet in Fig. 15b) is equivalent to measuring the pair of the sheet operators from Fig. 15c).

It is possible to use a ring to surround multiple loops of opposite type. In order to keep the discussion consistent, in Fig. 21 we use primals to illustrate how to implement XX- and ZZ- measurements.

Therefore, in braided geometries, one introduces a tube correlation between two loops in order to measure sheet operators tuples(i.e. for primals Z-operators), and introduces support for a sheet in order to measure tube operator tuples (i.e. for primals X-operators).

[1] J. van de Wetering, Zx-calculus for the working quantum computer scientist, arXiv preprint arXiv:2012.13966 (2020).

[2] H. Bombin, D. Litinski, N. Nickerson, F. Pastawski, and S. Roberts, Unifying flavors of fault tolerance with the zx calculus, Quantum **8**, 1379 (2024).

[3] A. G. Fowler and S. J. Devitt, A bridge to lower overhead quantum computation, arXiv preprint arXiv:1209.0510 (2012).

[4] A. Paler, I. Polian, K. Nemoto, and S. J. Devitt, Fault-tolerant, high-level quantum circuits: form, compilation and description, Quantum Science and Technology **2**, 025003 (2017).

[5] M. K. Vijayan, A. Paler, J. Gavriel, C. R. Myers, P. P. Rohde, and S. J. Devitt, Compilation of algorithm-specific graph states for quantum circuits, Quantum Science and Technology **9**, 025005 (2024).

[6] N. de Beaudrap and D. Horsman, The zx calculus is a language for surface code lattice surgery, Quantum **4**, 218 (2020).

[7] A. Paler, S. Devitt, K. Nemoto, and I. Polian, Software-based pauli tracking in fault-tolerant quantum circuits, in *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)* (IEEE, 2014) pp. 1–4.

[8] C. Gidney and A. G. Fowler, Flexible layout of surface code computations using autoccz states, arXiv preprint arXiv:1905.08916 (2019).

[9] C. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter, Surface code quantum computing by lattice surgery, New Journal of Physics **14**, 123011 (2012).

[10] D. Litinski, A game of surface codes: Large-scale quantum computing with lattice surgery, Quantum **3**, 128 (2019).

[11] A. Paler, *Design methods for reliable quantum circuits*, Ph.D. thesis, Universität Passau (2014).

[12] R. Raussendorf, J. Harrington, and K. Goyal, Topological fault-tolerance in cluster state quantum computation, New Journal of Physics **9**, 199 (2007).

[13] A. Paler, S. Devitt, K. Nemoto, and I. Polian, Synthesis of topological quantum circuits, in *2012 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)* (IEEE, 2012) pp. 181–187.

[14] D. B. Tan, M. Y. Niu, and C. Gidney, A sat scalpel for lattice surgery: Representation and synthesis of subroutines for surface-code fault-tolerant quantum computing, in *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)* (IEEE, 2024) pp. 325–339.

[15] M. Hanks, M. P. Estarellas, W. J. Munro, and K. Nemoto, Effective compression of quantum braided circuits aided by zx-calculus, Physical Review X **10**, 041030 (2020).

[16] M. Kupper, D. Horsman, C. Heunen, and N. De Beaudrap, String diagrams for defect-based surface code computing, arXiv preprint arXiv:2508.14672 (2025).

[17] A. G. Fowler and C. Gidney, Low overhead quantum computation using lattice surgery, arXiv preprint arXiv:1808.06709 (2018).

[18] G. Watkins, H. M. Nguyen, K. Watkins, S. Pearce, H.-K. Lau, and A. Paler, A high performance compiler for very large scale surface code computations, Quantum **8**, 1354 (2024).

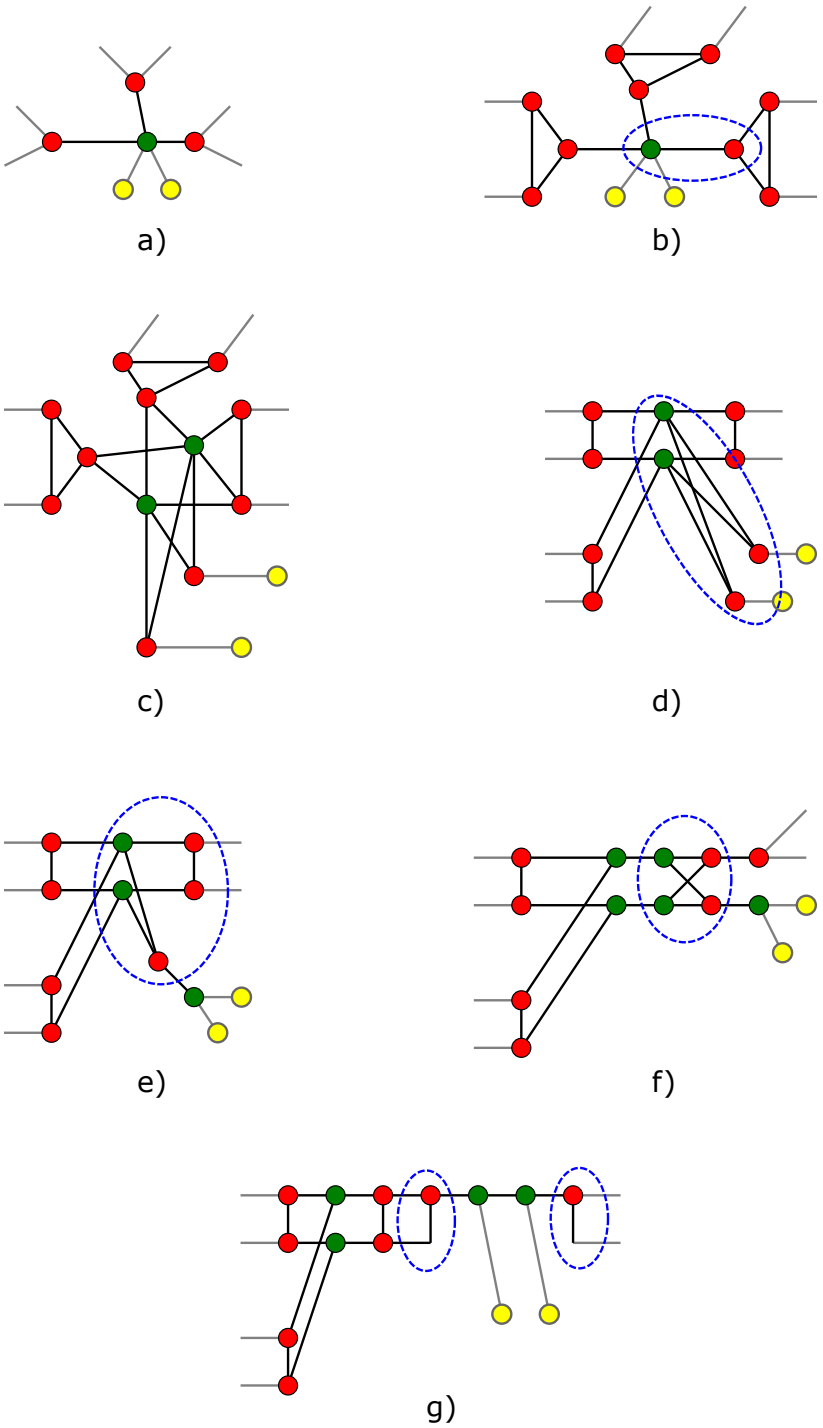

FIG. 22. The steps of obtaining the Raussendorf rule with the ZX calculus: a) the original ZX diagram; b) trivial loops are introduced; c) the result of applying the bialgebra ZX rule on the spiders previously encircled blue; d) reorganising the diagram by moving the top loop to the bottom; e) the result of applying the bialgebra ZX rule on the spiders previously encircled blue; f) splitting green spiders and merging red spiders; g) the result of applying the bialgebra ZX rule on the spiders previously encircled blue. After the last step there are red encircled spiders, which represent the fact that the GHZ-like-loop logical qubit is uncomputed to a single wire. This wire is braided-CNOT with the moustaches, and then the wire is returned again to a GHZ-like-loop.