

One rig to control them all

Chris Heunen¹, Robin Kaarsgaard², and Louis Lemonnier¹

¹University of Edinburgh

²University of Southern Denmark

Abstract

We introduce a theory for computational control, consisting of eight naturally interpretable equations. Adding these to a prop of base circuits constructs controlled circuits, borne out in examples of reversible Boolean circuits and quantum circuits. We prove that this syntactic construction semantically corresponds to taking the free rig category on the base prop.

1 Introduction

Many computations contain *controlled* commands, that is, commands that are executed depending on the value of some memory cell. By *control* we mean the aspects of a computation that govern these dependencies.¹ Typically, the controlled command acts on one part of memory, and the controlling memory cell resides in another part of memory. To be more precise, for example, consider controlled negation in reversible Boolean circuits: the target bit is flipped depending on the value of the control bit. The goal of this article is to identify, separate, and study in isolation, this notion of computational control.

Traditional control flow or data flow is often mixed up with other computational aspects of circuits. For example, in reversible Boolean or quantum circuits, multi-controlled gates such as the Toffoli gate are integral to universality and not treated differently than other, uncontrolled, gates [43]. Yet separating out the controlled aspects of a computation as specified by a circuit has several benefits.

- (i) Multi-controlled gates are of foundational importance in many computational theories, including Boolean logic [45], reversible computation [43, 42], and quantum computation [39]. Isolating their control logic can help to better *understand* these theories.
- (ii) In quantum hardware, (multi-)controlled gates are among the most costly ones to perform physically [5, 47, 13]. Separating out the control aspects can help find better optimisation strategies [4]. In general, partitioning off control aspects can help to *optimise* computations in a generic way that is independent of the ‘base circuit theory’ and therefore more efficient to apply.
- (iii) Several recent results about logical completeness for quantum computation rely on elaborate families of equations [7, 6, 32, 15, 14, 16, 18, 17, 23]. Cordoning off control aspects can *simplify*, and thereby clarify the core status of some and make them more modular.

This article addresses these three challenges by introducing a theory of control governed by a handful of equations (see Figure 1). We argue that these equations completely capture control as follows.

- (i) The equations have clear computational interpretations, and several have appeared in the literature before [38, 42]. Additionally, we show that the equations are canonical in a strong way, by relating them to the natural mathematical notion of a *rig category* [31, 46]. Starting with an arbitrary ‘base circuit theory’, we syntactically construct a new ‘controlled circuit theory’. We do this in the most general setting possible, using *props* [33, 25, 9]. The construction has a universal property: roughly, the controlled prop is the free rig category on the base prop. This is borne out in examples: starting with the circuit theory consisting of a single NOT gate, the controlled theory is universal for reversible Boolean circuits. Starting with an additional Hadamard gate, the controlled theory is universal for quantum circuits [40]. Starting with a single $\sqrt{X}$ gate in fact suffices for the latter [29].

¹We do not mean ‘control theory’ as used in e.g. engineering [22].

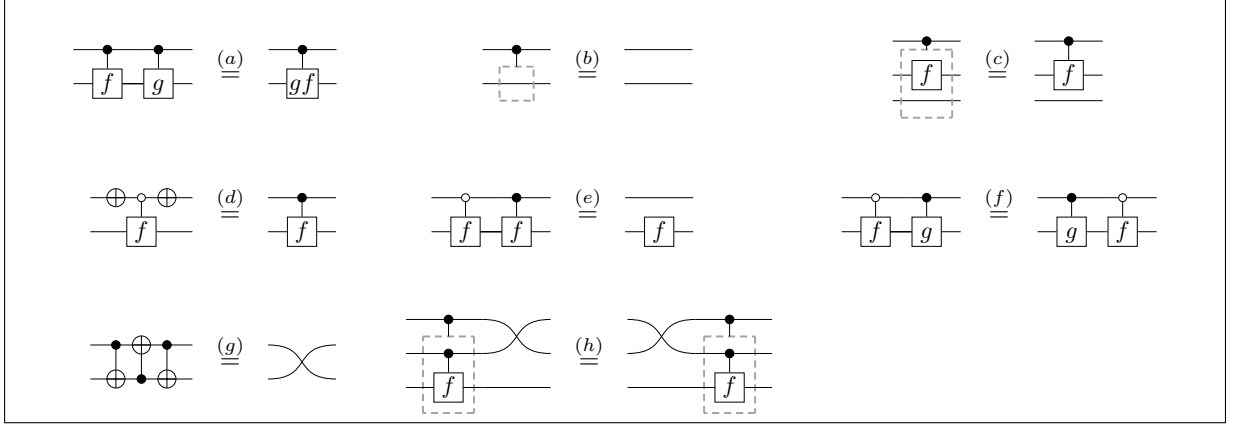


Figure 1: Control equations.

- (ii) The coherence theorems for rig categories [31, 46], and the fact that the control theory (of Figure 1) consists of only eight unquantified equations, give rise to many optimisation strategies. For example, we will show that the control theory suffices to syntactically derive equations like the following Sleator-Weinfurter decomposition quickly, that were before only known to hold semantically via matrix calculations in special cases [41, 11]:

- (iii) The equations simplify related work. The first works on complete equational theories for quantum circuits [16, 14, 15] contain a notion of *structural* equations, without a mathematical account of this notion. The same holds for [7, 6, 32]. The closest work to ours is a sister notion of ‘controlled prop’ [21], which is rather seen as a property of a category rather than a construction. Their definition is more general than ours, because they do consider NOT gates and colour changes. In their sense, our construction outputs a twice ‘controlled prop’, with a distinguished involution to change between the two controls. They also do not relate to the structure of rig categories, which makes their definition of ‘controlled prop’ incomplete in that sense. Similarly, template-based rewrite rules [42] for control in the literature, which are not complete in our sense.

This article fills that gap by showing that the structure of control is exactly that of rig categories. Our work similarly structures and elucidates a line of research on quantum programming languages taking semantics in rig categories [12, 26, 10, 11, 23]. Finally, our results obviate work on circumventing no-control theorems by restricting to specific gate sets [8].

These results also substantiate the claim in the title, that rig structure encapsulates controlled computation, and only controlled computation. Thus rig categories form the bare minimum model of computation: the ability to compose instructions sequentially (with $\circ$), to consider data in parallel (with $\otimes$), and to use one piece of data to condition computations on another (using $\oplus$). This may also explain the ubiquity of matrices.

We proceed as follows. First, we reviews background material on props, rig categories, permutations, and Gray codes (in Section 2). Then we introduce the main syntactic construction (in Section 3), and show that it semantically corresponds to rig completion (in Section 4). The latter proof uses in a pleasingly structural way the new technique of *Gray induction*, that proves a property for bitstrings by induction not on the normal successor of natural numbers, but by its Gray code. After that, we discuss several applications of the theory (in Section 5): a singly-generated universal theory for controlled reversible Boolean circuits, a generalised Sleator-Weinfurter decomposition [41], a relationship to modal quantum theory [37], and a universal singly-generated theory for controlled quantum circuits [29], and complete equational theories for quantum circuits over various gate sets. We conclude by indicating possible future work (in Section 6).

2 Background

2.1 Props, rig categories, and bipermutative categories

We assume familiarity with basic category theory, including (strict) symmetric monoidal categories, and recall here some basic facts about props, rig categories, and bipermutative categories. For a more in depth treatment on these topics, the interested reader is invited to consult, e.g., [46, 33].

A *prop* is a strict symmetric monoidal category whose objects are natural numbers, and where the monoidal product is given by addition. A *prop morphism* is an identity-on-objects strict symmetric monoidal functor between props. Props are naturally regarded as categories of diagrams, and play a similar role in the presentation of symmetric monoidal categories as free groups do in combinatorial group theory (see, e.g., [9]).

Rig categories are categories that are symmetric monoidal in two different ways, such that one distributes over the other, similar to how multiplication distributes over addition in arithmetic. More formally, a rig category $(\mathbf{C}, \otimes, \oplus, O, I)$ has two symmetric monoidal structures $(\mathbf{C}, \otimes, I)$ (the *tensor product*, with associator $\alpha^\otimes$, unitors $\lambda^\otimes$ and $\rho^\otimes$, and symmetry $\gamma^\otimes$) and $(\mathbf{C}, \oplus, O)$ (the *direct sum*, with associator $\alpha^\oplus$, unitors $\lambda^\oplus$ and $\rho^\oplus$, and symmetry $\gamma^\oplus$), and come with natural coherence isomorphisms witnessing distributivity on the right $\delta_R : (A \oplus B) \otimes C \rightarrow (A \otimes C) \oplus (B \otimes C)$ and $\delta_R^0 : O \otimes A \cong O$ and analogously on the left, subject to a significant number of coherence conditions (see [46]). When it is clear from the context, we will elide marking coherence isomorphisms with their monoidal structure, and simply write, e.g., α rather than $\alpha^\oplus$.

Bipermutative categories are the strictified version of rig categories: the two symmetric monoidal structures are strict, and right distributivity becomes an equality (such that $(A \oplus B) \otimes C = (A \otimes C) \oplus (B \otimes C)$), as do both nullary distributors (such that $A \otimes O = O = O \otimes A$), subject to three coherence conditions (see [34, 46]; the full definition of bipermutative category is also given in Appendix A). That this is the right notion of strict rig category is shown by the coherence theorem that every rig category is rig equivalent to a bipermutative one [34, Prop. 3.5]. We say that a bipermutative category is *semisimple* when its object are natural numbers, and the two strict symmetric monoidal structures are multiplication and addition of natural numbers on objects. Semisimple bipermutative categories are naturally regarded as props by forgetting the tensor product $(\otimes, I)$.

Proposition 1. *Given a semisimple bipermutative category $\mathbf{C}$ and a natural number k , the full subcategory of $\mathbf{C}$ whose objects are powers of k is a prop $(\mathbf{C}_k, \otimes, 1)$.*

Remark 2. By using the proposition above to work on bits by choosing powers of 2, we have a natural involution with the symmetry $\gamma_{1,1}$. If all morphisms are endomorphisms, then we obtain a controllable prop $(\mathbf{C}_2, \otimes, 1, \gamma_{1,1})$ as defined below.

We provide the most natural example of such a category in the next section.

2.2 Permutations

An important example of a semisimple bipermutative category is the category of permutations **Perm**: its objects are natural numbers, and morphisms $n \rightarrow n$ are permutations $[n] \rightarrow [n]$. The *direct sum* $\oplus$, on objects, simply sums the numbers, and on morphisms, concatenates them.

$$f \oplus g = \begin{cases} [m+n] & \rightarrow [m+n] \\ i & \mapsto f(i) & \text{if } i < m, \\ i & \mapsto g(i-m) + m & \text{otherwise.} \end{cases}$$

The symmetry γ for $\oplus$ is defined as follows.

$$\gamma_{m,n} = \begin{cases} [m+n] & \rightarrow [n+m] \\ i & \mapsto n+i & \text{if } i < m, \\ i & \mapsto i-m & \text{otherwise.} \end{cases}$$

The *tensor product* $\otimes$ multiplies the two numbers on objects, and acts on morphisms as follows.

$$f \otimes g = \begin{cases} [mn] & \rightarrow [mn] \\ an+b & \mapsto f(a)n + g(b) \end{cases}$$

The symmetry s for $\otimes$ is defined as:

$$s_{m,n} = \begin{cases} [mn] & \rightarrow [nm] \\ an + b & \mapsto bm + a \end{cases}$$

In fact, **Perm** is both the free prop and the free bipermutative category. This means that the tensor product and its symmetries can entirely be described in terms of $\oplus$ and $\gamma^\oplus$. In particular, for all $f: m \rightarrow m$ and $g: n \rightarrow n$:

$$f \otimes g = s_{n,m} \circ \underbrace{(f \oplus f \oplus \cdots \oplus f)}_{n \text{ times}} \circ s_{m,n} \circ \underbrace{(g \oplus g \oplus \cdots \oplus g)}_{m \text{ times}} \quad (1)$$

$$= \underbrace{(g \oplus g \oplus \cdots \oplus g)}_{m \text{ times}} \circ s_{n,m} \circ \underbrace{(f \oplus f \oplus \cdots \oplus f)}_{n \text{ times}} \circ s_{m,n} \quad (2)$$

Note that permutations are generated by transpositions $\tau_i = (i \ i+1)$, subject to equations $\tau_i^2 = \text{id}$ and $(\tau_i \tau_{i+1})^3 = \text{id}$ and $\tau_i \tau_j = \tau_j \tau_i$ when $i+1 < j$.

2.3 Gray codes

In this paper, we work with a similar but different way of generating permutations with transpositions on Gray codes. Gray codes are a way of ordering bit strings such that the transition from any bit string to its successor consists of a single bit flip—i.e., such that the Hamming distance between adjacent bit strings is 1. Gray codes play an important role in the synthesis of reversible circuits (see, e.g., [35]).

For example, a Gray code of bit strings of length 3 is shown in Figure 2.

More formally, we define a function $h_n: [2^n] \rightarrow \{0, 1\}^n$ as:

$$h_0(0) = \varepsilon \quad h_n(i) = \begin{cases} 0 \cdot h_{n-1}(i) & \text{if } i < 2^{n-1}, \\ 1 \cdot h_{n-1}(2^n - 1 - i) & \text{otherwise.} \end{cases}$$

We then fix *Gray transpositions* $\theta_{n,i}$ to swap $h_n(i)$ with $h_n(i+1)$. The two latter words have common prefix $w_n(i)$ and common suffix $w'_n(i)$. We also define permutations $r_n: [2^n] \rightarrow [2^n]$ that translate between the lexicographic order and the Gray order:

$$r_0(0) = \text{id}_1 \quad r_n(i) = \begin{cases} r_{n-1}(i) & \text{if } i < 2^{n-1}, \\ 2^{n-1} + r_{n-1}(2^n - 1 - i) & \text{otherwise.} \end{cases}$$

In other words, the transposition $\theta_{n,i}$ swaps $r_n(i)$ and $r_n(i+1)$.

Proposition 3. *For all n and i , we have: $\theta_{n,i} = r_n^{-1} \circ (\text{id}_i \oplus \gamma_{1,1} \oplus \text{id}_{2^n-2-i}) \circ r_n$.*

Since the translation permutations are defined by induction on the number of (qu)bits, we then obtain an inductive definition for the Gray transpositions.

$$\theta_{n+1,i} = \begin{cases} \theta_{n,i} \oplus \text{id}_{2^n} & \text{if } i < 2^n - 1, \\ s_{2^n,2} \circ (\text{id}_{2^n} \oplus \gamma_{1,1} \oplus \text{id}_{2^n-2}) \circ s_{2,2^n} & \text{if } i = 2^n - 1, \\ \text{id}_{2^n} \oplus \theta_{n,2^{n+1}-1-i} & \text{otherwise.} \end{cases} \quad (3)$$

3 Controlled props

In this section, we will show how to freely adjoin control to a prop, such that arbitrarily controlled versions of all morphisms of the original prop are morphisms in the controlled prop. For this to make sense, the prop must satisfy some lightweight requirements, namely those of a *controllable prop*.

Definition 4. A *controllable prop* $(\mathbf{P}, +, 0, x)$, or *crop* for short, is a prop $(\mathbf{P}, +, 0)$ whose morphisms are endomorphisms and in which one morphism $x: 1 \rightarrow 1$ is a distinguished involution. We sometimes refer to this involution as the *NOT gate*. A crop morphism is a prop morphism of the underlying props that preserves the chosen involution.

Binary	Gray
000	000
001	001
010	011
011	010
100	110
101	111
110	101
111	100

Figure 2: A Gray code on bit strings of length 3.

One important example of controllable prop is the prop $\mathbf{X}$ generated only by the NOT gate x .

We introduce a construction on controllable props, that consists in adding a control operator. In order to have readable equations and better presentation, we allow ourselves to write both a *positive* control (on the left) and a *negative* control (on the right). In fact, only one of them is necessary, since the other can be obtained by conjugating with the NOT morphism (see (d) in Definition 5).

$$C^1(f): \begin{array}{c} \bullet \\ \hline \boxed{f} \\ \vdots \end{array} \quad C^0(f): \begin{array}{c} \circ \\ \hline \boxed{f} \\ \vdots \end{array}$$

Both these operators are defined as functors. This means that they preserve both composition and identities.

$$\begin{array}{c} \bullet \\ \hline \boxed{f} \boxed{g} \\ \vdots \end{array} = \begin{array}{c} \bullet \quad \bullet \\ \hline \boxed{f} \boxed{g} \\ \vdots \end{array} \quad \begin{array}{c} \bullet \\ \hline \boxed{\vdots} \\ \vdots \end{array} = \begin{array}{c} \hline \vdots \end{array}$$

We represent the chosen involution $x: 1 \rightarrow 1$ with the usual NOT gate.

$$\text{---} \oplus \text{---}$$

Because props are also equipped with a symmetry for the tensor, we can also obtain controlled operation with a control wire at the bottom.

$$\begin{array}{c} \boxed{f} \\ \vdots \\ \bullet \end{array} \stackrel{\text{def}}{=} \begin{array}{c} \bullet \\ \hline \text{---} \oplus \text{---} \\ \vdots \end{array}$$

In the rest of the paper, we draw a single wire to picture any object n .

Definition 5 (Controlled prop). Given a controllable prop $(\mathbf{P}, +, 0, x)$, we extend it to a new prop with endofunctors C^0 and C^1 such that if $f: n \rightarrow n$, then $C^b(f): 1 + n \rightarrow 1 + n$, and that, for all n and $f, f_1, f_2: n \rightarrow n$, we have equations:

- (a) composition: $C^1(g \circ f) = C^1(g) \circ C^1(f)$;
- (b) identity: $C^1(\text{id}_n) = \text{id}_{n+1}$;
- (c) strength: $C^1(f + \text{id}_m) = C^1(f) + \text{id}_m$;
- (d) colour change: $(x + \text{id}_n) \circ C^0(f) \circ (x + \text{id}_n) = C^1(f)$;
- (e) complementarity: $C^0(f) \circ C^1(f) = \text{id}_1 + f$;
- (f) commutativity: $C^0(f_1) \circ C^1(f_2) = C^1(f_2) \circ C^0(f_1)$;
- (g) “swap”: $C^1(x) \circ \text{swap}_{1,1} \circ C^1(x) \circ \text{swap}_{1,1} \circ C^1(x) = \text{swap}_{1,1}$;
- (h) “swap” coherence: $(\text{swap}_{1,1} \otimes \text{id}_n) \circ C^1(C^1(f)) = C^1(C^1(f)) \circ (\text{swap}_{1,1} \otimes \text{id}_n)$.

Figure 1 gives a diagrammatic account of the above equations. Write $\mathbf{cP}$ for this new prop, that we call the *controlled prop* of $\mathbf{P}$. Note that $(\mathbf{cP}, +, 0, x)$ is itself a controllable prop.

Remark 6. The strength and composition equations above imply that, for all $f: m \rightarrow m$ and $g: n \rightarrow n$, we have:

$$\begin{array}{c} \bullet \\ \hline \boxed{f} \\ \vdots \\ \bullet \\ \hline \boxed{g} \\ \vdots \end{array} = \begin{array}{c} \bullet \quad \bullet \\ \hline \boxed{f} \quad \boxed{g} \\ \vdots \end{array} = \begin{array}{c} \bullet \quad \bullet \\ \hline \boxed{g} \quad \boxed{f} \\ \vdots \end{array}$$

Throughout the paper, we make extensive use of multi-controlled gates, such as

$$\begin{array}{c} \bullet \\ \hline \oplus \\ \vdots \end{array} = \begin{array}{c} \bullet \\ \hline \oplus \\ \vdots \end{array} : (\text{id}_2 + \text{swap}_{1,1}) \circ C^1(C^0(C^0(x))) \circ (\text{id}_2 + \text{swap}_{1,1}),$$

that we denote by $C_0^{10}(x)$ for readability, and which are well-defined thanks to Equation (h). More generally, given two Boolean words w and w' , we write $C_{w'}^w(f)$ for the gate f controlled on $|w|$ top wires, positively or negatively according to w , and controlled on $|w'|$ bottom wires, according to w' .

Proposition 7. *A controllable prop $\mathbf{P}$ is isomorphic to the wide subcategory of $\mathbf{cP}$ whose morphisms are only the non-controlled ones. This also means that there is an embedding $\mathbf{P} \hookrightarrow \mathbf{cP}$.*

There are two forms of multiple controlled morphisms. Already in $\mathbf{cP}$, one can control morphisms from $\mathbf{P}$ on multiple wires. In $\mathbf{ccP}$, one can additionally formally control morphisms from $\mathbf{cP}$. In fact, the assignment $\mathbf{c}-$ is a monad on the category of crops and crop morphisms. The monad multiplication flattens the latter type of multiple control, but because the two forms of multiple control are distinct, the monad is not idempotent. The monad unit is the embedding from the previous proposition.

Relation to permutations. The simplest example of a controlled prop is already quite meaningful, as it captures all classical computing. To justify this statement, we prove that the controlled prop $\mathbf{cX}$ of the prop $\mathbf{X}$ generated by a single involution is isomorphic to the category of permutations between finite sets of cardinality a power of 2.

Theorem 8. *The controlled prop $\mathbf{cX}$ is crop isomorphic to $\mathbf{Perm}_2$.*

To prove this, we define functors in both directions and show that they are inverse of each other. Let us first define a crop morphism $\alpha: \mathbf{cX} \rightarrow \mathbf{Perm}_2$.

$$\begin{aligned} \alpha: \quad \mathbf{cX} &\rightarrow \mathbf{Perm}_2 \\ n &\mapsto 2^n \\ x &\mapsto \gamma_{1,1} \\ \text{swap}_{m,n} &\mapsto s_{2^m, 2^n} \\ C^1(f) &\mapsto \text{id}_{2^n} \oplus \alpha(f) \\ C^0(f) &\mapsto \alpha(f) \oplus \text{id}_{2^n} \\ \text{id}_m + f &\mapsto \text{id}_{2^m} \otimes \alpha(f) \end{aligned}$$

Lemma 9. *The crop morphism $\alpha: \mathbf{cX} \rightarrow \mathbf{Perm}_2$ is well-defined.*

The proof consists in showing that any equation that holds in $\mathbf{cX}$ also holds in $\mathbf{Perm}_2$ after the application of α . In summary, the equations (a) and (b) hold in $\mathbf{Perm}_2$ because $\oplus$ is a (bi)functor; (c) follows by distributivity coherence; (d) by naturality of the symmetry γ for $\oplus$; (e) and (f) by bifunctoriality of $\oplus$; (g) by naturality of γ and associative coherence.

We now define a functor $\beta: \mathbf{Perm}_2 \rightarrow \mathbf{cX}$ which maps a Gray code transposition $\theta_{n,i}$ to its associated multi-controlled NOT:

$$\begin{aligned} \beta: \quad \mathbf{Perm}_2 &\rightarrow \mathbf{cX} \\ 2^n &\mapsto n \\ \theta_{n,i} &\mapsto C_{w'_n(i)}^{w_n(i)}(x) \end{aligned}$$

where $w_n(i)$ and $w'_n(i)$ are respectively the common prefix and suffix of the i^{th} word in the Gray code $h_n(i)$. In a Gray code order, two successive Boolean words have only one bit that differs, for example 1011 and 1001 when $n = 4$ and $i = 13$. The operation that permutes these two words is a NOT gate on the third bit, triggered when the first bit is 1, the second 0 and the fourth 1. This particular example corresponds to the multi-controlled gate $C_1^{10}(x)$ in $\mathbf{cX}$:



We use the induction formula for Gray code transpositions (3) and the fact that all permutations are generated by (Gray code) transpositions to prove that $\beta: \mathbf{Perm}_2 \rightarrow \mathbf{cX}$ is a well-defined functor, and is in fact a crop morphism.

Lemma 10. *The functor $\beta: \mathbf{Perm}_2 \rightarrow \mathbf{cX}$ is a well-defined crop morphism, and*

$$\beta(\text{id}_{2^n} \oplus f) = C^1(\beta(f)), \quad \beta(f \oplus \text{id}_{2^n}) = C^0(\beta(f)),$$

for all morphisms $f: 2^n \rightarrow 2^n$.

Proof. We recall that any permutation $2^n \rightarrow 2^n$ is generated by Gray code transpositions $\theta_{n,i}$, with the equations below, respectively referred to as involution, Yang-Baxter [36], and commutativity.

$$\forall i < 2^n - 1, \theta_{n,i}^2 = \text{id}_{2^n}, \quad \forall i < 2^n - 2, (\theta_{n,i} \theta_{n,i+1})^3 = \text{id}_{2^n}, \quad \forall i + 1 < j, \theta_{n,i} \theta_{n,j} = \theta_{n,j} \theta_{n,i}.$$

The functor $\beta: \mathbf{Perm}_2 \rightarrow \mathbf{cX}$ is well-defined: the involution equation follows from (a) and (b); Yang-Baxter is proven by induction, the base case on two (q)bits is a direct consequence of (g), alongside (d) and (e), and the rest follows thanks to the induction formula for Gray codes and (h); the commutation is an easy consequence of (f) in Definition 5.

This functor is monoidal: first note that in $\mathbf{Perm}_2$, we have $s_{2,2} = \theta_{2,1} \circ \theta_{2,2} \circ \theta_{2,1}$ (and we have $h_2(1) = 01$, $h_2(2) = 11$ and $h_2(3) = 10$), whose image through β is then:

$$\begin{aligned} \beta(\theta_{2,1} \circ \theta_{2,2} \circ \theta_{2,1}) &= \beta(\theta_{2,1}) \circ \beta(\theta_{2,2}) \circ \beta(\theta_{2,1}) \\ &= C_1(x) \circ C^1(x) \circ C_1(x) \\ &\stackrel{(\text{def})}{=} \text{swap}_{1,1} \circ C^1(x) \circ \text{swap}_{1,1} \circ C^1(x) \circ \text{swap}_{1,1} \circ C^1(x) \circ \text{swap}_{1,1} \\ &\stackrel{(g)}{=} \text{swap}_{1,1} \circ \text{swap}_{1,1} \circ \text{swap}_{1,1} = \text{swap}_{1,1} \end{aligned}$$

thus β preserves the symmetry of the tensor; additionally, we know that:

$$\begin{aligned} \text{id}_2 \otimes \theta_{n,i} &= \theta_{n,i} \oplus \theta_{n,i} \\ &= (\theta_{n,i} \oplus \text{id}_{2^n}) \circ (\text{id}_{2^n} \oplus \theta_{n,i}) \\ &\stackrel{(3)}{=} \theta_{n+1,i} \circ \theta_{n+1,2^{n+1}-1-i} \end{aligned}$$

and therefore, since $i < 2^n - 2$, we have:

$$\begin{aligned} \beta(\text{id}_2 \otimes \theta_{n,i}) &= \beta(\theta_{n+1,i}) \circ \beta(\theta_{n+1,2^{n+1}-1-i}) \\ &= C_{w'_{n+1}(i)}^{w_{n+1}(i)}(x) \circ C_{w'_{n+1}(2^{n+1}-1-i)}^{w_{n+1}(2^{n+1}-1-i)}(x) \\ &= C^0 \left(C_{w'_n(i)}^{w_n(i)}(x) \right) \circ C^1 \left(C_{w'_n(i)}^{w_n(i)}(x) \right) \\ &\stackrel{(e)}{=} \text{id}_1 + C_{w'_n(i)}^{w_n(i)}(x) = \text{id}_1 + \beta(\theta_{n,i}) \end{aligned}$$

which concludes that the functor β is monoidal, since any permutations is generated by the $\theta_{n,i}$. Moreover, since we have $\theta_{1,0} = \gamma_{1,1}$, and since its image by β is x , the functor β preserves the NOT gate. Therefore, β is a crop morphism.

Additionally, we prove that for all f , we have $\beta(\text{id}_{2^n} \oplus f) = C^1(\beta(f))$. Once again, it suffices to prove it for the generators, therefore:

$$\begin{aligned} \beta(\text{id} \oplus \theta_{n,i}) &\stackrel{(3)}{=} \beta(\theta_{n+1,2^{n+1}-1-i}) \\ &= C_{w'_{n+1}(2^{n+1}-1-i)}^{w_{n+1}(2^{n+1}-1-i)}(x) \\ &= C^1 \left(C_{w'_n(i)}^{w_n(i)}(x) \right) \\ &= C^1(\beta(f)) \end{aligned}$$

and similarly, we have $\beta(f \oplus \text{id}_{2^n}) = C^0(\beta(f))$. □

This implies that β is the inverse of α , which suffices to prove Theorem 8.

4 Rigged props thanks to Kronecker

As shown in the previous section, in particular in the definition of the functor α , control is linked to the additive tensor $\oplus$, which we loosely call *direct sum*. This means that if we can extend a ‘base circuit theory’ with direct sums, we should get its controlled circuit theory. In fact, the better way is to start from a prop generated by the direct sum, and then recapture the ‘base circuit theory’ on the tensor product $\otimes$ from there.

In the context of matrices, the Kronecker product can be computed with direct sums only. Given a matrix M , the matrix $I_n \otimes M$ is obtained as the block-diagonal matrix

$$\left[\begin{array}{ccc} M & & \\ & \ddots & \\ & & M \end{array} \right] \Bigg\} n \text{ times},$$

or more precisely, $M \oplus \cdots \oplus M$. Additionally, the symmetry for the tensor product $\otimes$ can be expressed in terms of composition of symmetries for the direct sum. This shows that any theory around the direct sum can simulate the one of a tensor product, given some coherence. One of these is the bifactoriality of the tensor product, or in pictures, the following equality:

$$\begin{array}{c} \boxed{f} \\ \hline \end{array} = \begin{array}{c} \boxed{f} \\ \hline \end{array}$$

In a bipermutative category, with $f: k \rightarrow l$ and $g: m \rightarrow n$, we have:

$$\begin{aligned} f \otimes g &= (f \otimes \text{id}_n) \circ (\text{id}_k \otimes g) &= (\text{id}_l \otimes g) \circ (f \otimes \text{id}_m) \\ &= s_{n,l} \circ (\text{id}_n \otimes f) \circ s_{k,n} \circ (\text{id}_k \otimes g) &= (\text{id}_l \otimes g) \circ s_{m,l} \circ (\text{id}_m \otimes f) \circ s_{k,m} \\ &= s_{n,l} \circ \underbrace{(f \oplus \cdots \oplus f)}_{n \text{ times}} \circ s_{k,n} \circ \underbrace{(g \oplus \cdots \oplus g)}_{k \text{ times}} &= \underbrace{(g \oplus \cdots \oplus g)}_{l \text{ times}} \circ s_{m,l} \circ \underbrace{(f \oplus \cdots \oplus f)}_{m \text{ times}} \circ s_{k,m} \end{aligned}$$

To reflect this in a tensor product $\otimes$ generated by a direct sum $\oplus$, the last line above needs to be enforced, leading to our definition of rigged prop below.

4.1 Rigged props

We are now able to define rigged props as described above: a new prop whose monoidal structure is meant as a direct sum. Its generators are the ones from the base prop, for which we adjust the domains and codomains to a matrix point of view. The relations for this new prop are the ones of the base prop, in addition to the relations (5) that ensure that the newly obtained tensor product is bifactorial.

Definition 11 (Rigged prop). Given a prop $(\mathbf{P}, +, 0)$ with a set of generators G , and a set of relations R , its *rigged prop* $(\tilde{\mathbf{P}}, \oplus, 0)$ is the prop generated by

$$\tilde{G} = \{\tilde{g}: 2^k \rightarrow 2^l \mid g: k \rightarrow l \in G\} \quad (4)$$

with equations

$$\tilde{R} = \left\{ \widetilde{f + \text{id} \circ \text{id} + h} = \widetilde{\text{id} + h \circ f + \text{id}} \mid f, h \in G \right\} \quad (5)$$

$$\cup \left\{ \tilde{f} = \tilde{h} \mid (f = h) \in R \right\} \quad (6)$$

where $\widetilde{}$ is defined as follows for $f: k \rightarrow l$, $g: l \rightarrow m$, and $h: m \rightarrow n$.

$$\widetilde{\text{id}_n} = \text{id}_{2^n} \quad (7)$$

$$\widetilde{g \circ f} = \tilde{g} \circ \tilde{f} \quad (8)$$

$$\tilde{f} \otimes \tilde{h} \stackrel{\text{def}}{=} \widetilde{f + h} = s_{2^n, 2^l} \circ \underbrace{(\tilde{f} \oplus \tilde{f} \oplus \cdots \oplus \tilde{f})}_{2^n \text{ times}} \circ s_{2^k, 2^n} \circ \underbrace{(\tilde{h} \oplus \tilde{h} \oplus \cdots \oplus \tilde{h})}_{2^k \text{ times}} \quad (9)$$

Relations (5) and (6) ensure that $\tilde{\mathbf{P}}$ is larger than $\mathbf{P}$, meaning that there is an embedding $\mathbf{P} \hookrightarrow \tilde{\mathbf{P}}$.

Theorem 12. *Given a prop $\mathbf{P}$, its rigged prop $\tilde{\mathbf{P}}$ is semisimple bipermutative.*

Proof. There is a prop morphism $E: \mathbf{Perm} \rightarrow \tilde{\mathbf{P}}$ since $\mathbf{Perm}$ is the free prop. The $\otimes$ monoidal structure on $\tilde{\mathbf{P}}$ can then be defined the same way as in $\mathbf{Perm}$. The tensor $\otimes$ in $\tilde{\mathbf{P}}$ is indeed a bifunctor, mainly thanks to (5). Equation (9) ensures that the symmetry of the tensor $\oplus$ is natural. The coherence equations are inherited from $\mathbf{Perm}$ through E . Therefore, $\tilde{\mathbf{P}}$ is semisimple bipermutative. $\square$

This construction $\widetilde{}$ provides not merely a semisimple bipermutative category, but the universal one.

Theorem 13. *Given a prop $\mathbf{P}$ and a strict monoidal functor to a semisimple bipermutative category $F: \mathbf{P} \rightarrow \mathbf{Q}$ such that $F(1) = 2$, there is a unique strict bipermutative functor $\tilde{F}: \tilde{\mathbf{P}} \rightarrow \mathbf{Q}$ such that the following diagram commutes:*

$$\begin{array}{ccc} \mathbf{P} & \xrightarrow{\quad} & \tilde{\mathbf{P}} \\ & \searrow F & \downarrow \tilde{F} \\ & & \mathbf{Q} \end{array}$$

Proof. Let us first define the resulting functor. Since $F(1) = 2$, we have $F(n) = 2^n$. Thus we can define:

$$\tilde{F}(n) = n \quad \tilde{F}(\tilde{g}) = F(g)$$

and this suffices to define the functor $\tilde{F}$. All the equations in $\tilde{P}$ derive from P or from the rig structure, and therefore $\tilde{F}$ is well-defined because it is strict rig and because F is well-defined. A strict rig functor between two semisimple bipermutative categories is necessarily the identity on objects. Since the functor is obtained by the image of F , it is necessarily unique. $\square$

The domains and codomains of the generators for $\tilde{\mathbf{P}}$, in (4), can in fact be the power of any arbitrary natural number. We choose 2 to fit the story of circuits on bits. Moreover, the definition of rigged prop does not account for the NOT gate, which plays an central role in our theory of control. This is the focus of the next section.

4.2 Rigged crops

We now present a definition of rigged props that takes into account the choice of NOT gate. Whether it is classical reversible circuits or quantum circuits, the NOT gate is represented as the following matrix:

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

which is also the symmetry $\gamma_{1,1}$ for the direct sum $\oplus$. Given a crop with a choice of NOT gate, we further identify the chosen NOT gate with the symmetry $\gamma_{1,1}$ in its constructed rigged prop, to obtain its rigged crop.

Definition 14 (Rigged crop). Given a controllable prop $(\mathbf{P}, +, 0, x)$ with a set of generators G , and a set of relations R , its *rigged crop* $(\overline{\mathbf{P}}, \oplus, 0)$ is the prop generated by

$$\overline{G} = \tilde{G}$$

subject to equations

$$\overline{R} = \{\tilde{x} = \gamma_{1,1}\} \cup \tilde{R}.$$

The rigged crop is semisimple bipermutative in the same way as the rigged prop.

Theorem 15. *The category $(\overline{\mathbf{P}}, \otimes, 1, \oplus, 0)$ is semisimple bipermutative.*

Furthermore, rigged crops enjoy a similar universality as rigged props.

Theorem 16. *Given a controllable prop $(\mathbf{P}, +, 0, x)$, and a strict monoidal functor $F: \mathbf{P} \rightarrow \mathbf{Q}$ to a semisimple bipermutative category with $F(x) = \gamma_{1,1}$, there is a unique strict rig functor $\overline{F}: \overline{\mathbf{P}} \rightarrow \mathbf{Q}$ such that the following diagram commutes.*

$$\begin{array}{ccc} \mathbf{P} & \hookrightarrow & \overline{\mathbf{P}} \\ & \searrow F & \downarrow \overline{F} \\ & & \mathbf{Q} \end{array}$$

In the following, we are interested in objects in $\overline{\mathbf{P}}$ that are powers of 2. Given how generators in $\mathbf{P}$ are mapped into $\overline{\mathbf{P}}$, the embedding $\mathbf{P} \hookrightarrow \overline{\mathbf{P}}$ corestricts to an embedding $\mathbf{P} \hookrightarrow \overline{\mathbf{P}}_2$.

Corollary 17. *Given a controllable prop $(\mathbf{P}, +, 0, x)$, and a strict monoidal functor $F: \mathbf{P} \rightarrow \mathbf{Q}$ to a semisimple bipermutative category with $F(x) = \gamma_{1,1}$, there is a unique prop morphism $\overline{F}_2: \overline{\mathbf{P}}_2 \rightarrow \mathbf{Q}_2$ such that $\overline{F}_2$ preserves the NOT gate and the following diagram commutes.*

$$\begin{array}{ccc} \mathbf{P} & \hookrightarrow & \overline{\mathbf{P}}_2 \\ & \searrow F & \downarrow \overline{F}_2 \\ & & \mathbf{Q}_2 \end{array}$$

Relation to permutations. Observe that, similarly to the controlled prop on a single involution, the rigged crop on a single involution is the category of permutations.

Theorem 18. *The category $\overline{\mathbf{X}}$ is exactly the category of permutations.*

Proof. Permutations are generated by transpositions $\tau_i = (i \ i+1)$, which can be mapped in $\overline{\mathbf{X}}$ to the involutions $\text{id}_i \oplus \gamma_{1,1} \oplus \text{id}_{n-3-i}$. The latter commute if they act on separate parts of the object. Moreover, the Yang-Baxter equation $(\tau_i \ \tau_{i+1})^3 = \text{id}$ follows from naturality of γ and associative coherence. Thus the functor $\mathbf{Perm} \rightarrow \overline{\mathbf{X}}$ is well-defined and the identity on objects. Because it maps coherence morphisms to coherence morphisms, it is in fact a strict bipermutative functor. As $\overline{\mathbf{X}}$, just like $\mathbf{Perm}$, only contains coherence morphisms, the functor is an isomorphism. $\square$

We now redefine the functors $\alpha: \mathbf{cX} \rightarrow \overline{\mathbf{X}}_2$ and $\beta: \overline{\mathbf{X}}_2 \rightarrow \mathbf{cX}$ to fit the topic at hand.

Remark 19. In principle, the notion of control makes sense for any prop consisting only of endomorphisms. However, we focus here on groupoids, since this is where controlled gates originated. In the more general, irreversible setting, control certainly makes sense, though more space-efficient gadgets such as multiplexers are typically used instead.

4.3 Rig is control

In this section, we fix a controllable prop $(\mathbf{P}, +, 0, x)$. It is now a matter of proving that $\mathbf{cP}$ and $\overline{\mathbf{P}}_2$ are isomorphic. First let us define a functor $\mathbf{cP}$ to $\overline{\mathbf{P}}_2$.

$$\begin{aligned} A: \quad \mathbf{cP} &\rightarrow \overline{\mathbf{P}}_2 \\ n &\mapsto 2^n \\ f \in \mathbf{cX} &\mapsto \alpha(f) \\ g \in G &\mapsto \tilde{g} \\ C^1(f) &\mapsto \text{id}_{2^n} \oplus A(f) \\ C^0(f) &\mapsto A(f) \oplus \text{id}_{2^n} \\ \text{id}_m + f &\mapsto \text{id}_{2^m} \otimes A(f) \end{aligned}$$

Lemma 20. *The prop morphism $A: \mathbf{cP} \rightarrow \overline{\mathbf{P}}_2$ is well-defined.*

Proof. We need to prove that all equations that hold in $\mathbf{cP}$ also hold in $\overline{\mathbf{P}}_2$ after taking the image under A . The equations of monoidal coherence hold because $\overline{\mathbf{P}}_2$ is a prop. We further observe that if f does not contain any control, then $A(f) = \tilde{f}$; therefore, all original equations from $\mathbf{P}$ hold by (6). Additionally, the equations (a) and (b) hold in $\mathbf{Perm}_2$ because $\oplus$ is a (bi)functor; (c) follows by coherence of distributivity; (d) by naturality of the symmetry γ for $\oplus$; (e) and (f) by bifunctoriality of $\oplus$; and (g) by naturality of γ and associative coherence. $\square$

Similarly, we can define a functor $\overline{\mathbf{P}}_2 \rightarrow \mathbf{cP}$.

$$\begin{aligned} B: \quad \overline{\mathbf{P}}_2 &\rightarrow \mathbf{cP} \\ 2^n &\mapsto n \\ f \in \overline{\mathbf{X}}_2 &\mapsto \beta(f) \\ \tilde{g} \in \overline{G} &\mapsto g \\ \text{id}_{2^n} \oplus f &\mapsto C^1(B(f)) \\ f \oplus \text{id}_{2^n} &\mapsto C^0(B(f)) \end{aligned}$$

Lemma 21. *The functor $B: \overline{\mathbf{P}}_2 \rightarrow \mathbf{cP}$ is a well-defined prop morphism.*

Proof. Note that $\overline{\mathbf{P}}$ is generated by $\overline{G}$ and permutations γ . Therefore, morphisms are of the form:

$$\gamma_0 \circ (\text{id} \oplus \tilde{g}_1 \oplus \text{id}) \circ \gamma_1 \circ \cdots \circ \gamma_{n-1} \circ (\text{id} \oplus \tilde{g}_n \oplus \text{id}) \circ \gamma_n$$

where γ_i are short for compositions of symmetries for the direct sum $\oplus$. Since the generator terms are surrounded by permutations, we can write them as follows, without loss of generality:

$$\gamma_0 \circ (\text{id} \oplus \tilde{g}_1) \circ \gamma_1 \circ \cdots \circ \gamma_{n-1} \circ (\text{id} \oplus \tilde{g}_n) \circ \gamma_n$$

If this is a morphism $2^m \rightarrow 2^m$ in $\overline{\mathbf{P}}_2$, then the terms containing generators $\tilde{g}: 2^k \rightarrow 2^k$ are of the form:

$$\text{id}_{2^m-2^k} \oplus \tilde{g} = \text{id}_{2^{m-1}} \oplus (\text{id}_{2^{m-2}} \oplus (\cdots \oplus (\text{id}_{2^k} \oplus \tilde{g}) \cdots))$$

Therefore the definition of B above defines an image of all morphism in $\overline{\mathbf{P}}_2$. We now prove that B is well-defined. All coherence equations hold because β is well-defined. The only equation specific to $\overline{\mathbf{P}}_2$ is (5), which is obviously preserved since $\otimes$ is a monoidal structure, provided that B is indeed a prop morphism. We thus need to prove that $B(f \otimes g) = B(f) + B(g)$ for all f and g . We know, through β , that $B(s_{2^m, 2^n}) = \text{swap}_{m,n}$, so that it suffices to show that $B(\text{id}_2 \otimes f) = \text{id}_1 + B(f)$.

$$\begin{aligned} B(\text{id}_2 \otimes f) &= B(f \oplus f) \\ &= B(f \oplus \text{id}_n) \circ B(\text{id}_n \oplus f) \\ &= C^0(B(f)) \circ C^1(B(f)) \\ &\stackrel{(e)}{=} \text{id}_1 + B(f) \end{aligned}$$

Thus B is a prop morphism. □

Both functors are well-defined and each other's inverse, yielding an isomorphism of props.

Theorem 22. *The props $\mathbf{cP}$ and $\overline{\mathbf{P}}_2$ are isomorphic.*

Thus the control equations of Figure 1 are exactly the ones that turn a crop into a rigged crop. In this sense, we conclude they are structural: they govern exactly the rig structure. A fortiori, Corollary 17 shows that the control equations are minimal: our definition of controlled prop (see Definition 5) is universal among such structural categories.

5 Applications

To showcase the usefulness of the control equations of Figure 1, this section discusses five applications drawn from circuit theory, both Boolean and quantum, that can be handled easily using our control theory.

5.1 Reversible Boolean circuits

We have already seen that controlling the degenerate prop consisting of a single NOT gate generates the full prop of permutations between sets of size power of two, that is, of reversible Boolean circuits. To make this more concrete, let us derive the circuit identity

expressing that two NOT gates that are controlled on the same wire but have different targets always commute. This may seem obvious, but usual proofs resort to truth tables. In contrast, the proof below is equational, and demonstrates how the control equations interact. First, observe the following.

$$\begin{array}{c} \bullet \\ \oplus \end{array} \stackrel{(\text{inv.})}{=} \begin{array}{c} \bullet \\ \oplus \oplus \oplus \end{array} \stackrel{(e)}{=} \begin{array}{c} \bullet \bullet \circ \\ \oplus \oplus \oplus \oplus \end{array} \stackrel{(a)}{=} \begin{array}{c} \circ \\ \oplus \oplus \end{array} \quad (10)$$

The desired equation now follows.

$$\begin{array}{c} \oplus \\ \bullet \bullet \\ \oplus \end{array} \stackrel{(10)}{=} \begin{array}{c} \oplus \\ \bullet \circ \\ \oplus \oplus \end{array} \stackrel{(f)}{=} \begin{array}{c} \oplus \\ \circ \bullet \\ \oplus \oplus \end{array} \stackrel{(10)}{=} \begin{array}{c} \oplus \\ \bullet \bullet \\ \oplus \end{array} \quad (11)$$

5.2 Sleator-Weinfurter

The Sleator-Weinfurter construction is a general construction for forming a doubly controlled f^2 -gate using nothing but controlled f -gates and controlled NOT gates (see Figure 3). This property has led to the quantum synthesis of reversible circuits being studied by means of the NCV gate set $\{NOT, CV, CNOT\}$ (see, e.g., [1]). Originally shown through linear algebra [5], the construction has a simple proof in terms of the control equations.

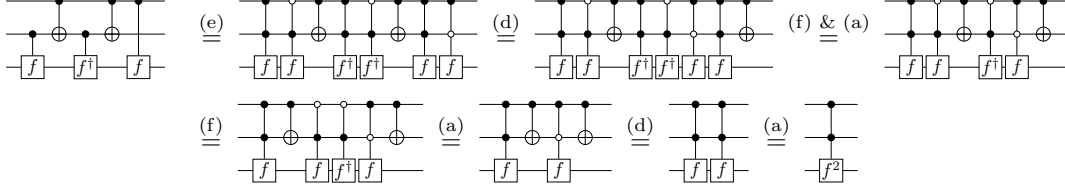


Figure 3: Sleator-Weinfurter identity through control equations.

5.3 Schumacher-Westmoreland: modal quantum theory

Starting from a theory of matrices, it is now easy to generate a circuit equational theory with our control completion of circuit props. We give an example of this on modal quantum theory [37]. The latter is a toy theory for quantum mechanics where scalars are not complex numbers but drawn from a finite field, for example the two-element field $\mathbb{Z}_2$. Of course this cannot express all of quantum theory, but it still retains key notions such as entanglement, mixed states, superdense coding and teleportation [37]. This makes modal quantum theory quite a rich toy model, which makes modal quantum circuits, acting on *mobits* (modal bits), an interesting case study.

As pointed out by Lafont [30], reversible matrices on $\mathbb{Z}_2$ have two generators

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad J = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

governed by the following complete [30, Theorem 6] set of matrix equations.

$$X^2 = \text{id}_2 \tag{12}$$

$$J^3 = \text{id}_2 \tag{13}$$

$$JXJ = X \tag{14}$$

$$(\text{id}_1 \oplus X)(X \oplus \text{id}_1)(\text{id}_1 \oplus X) = (X \oplus \text{id}_1)(\text{id}_1 \oplus X)(X \oplus \text{id}_1) \tag{15}$$

$$(\text{id}_1 \oplus X)(J \oplus \text{id}_1)(\text{id}_1 \oplus X) = (X \oplus \text{id}_1)(\text{id}_1 \oplus J)(X \oplus \text{id}_1) \tag{16}$$

$$(\text{id}_1 \oplus J)(X \oplus \text{id}_1)(\text{id}_1 \oplus J) = (J \oplus \text{id}_1)(\text{id}_1 \oplus J)(J \oplus \text{id}_1) \tag{17}$$

The first three equations are simple mobit equations, (15) is Yang-Baxter, and (16) is the naturality of the symmetry. The last one is not structural, and translates to the following circuit equation.

$$\begin{array}{c} \bullet \\ \diagdown \\ \boxed{J} \end{array} \begin{array}{c} \bullet \\ \diagup \\ \boxed{J} \end{array} = \begin{array}{c} \bullet \\ \diagdown \\ \boxed{J} \end{array} \begin{array}{c} \bullet \\ \diagup \\ \boxed{J} \end{array} \begin{array}{c} \bullet \\ \diagdown \\ \boxed{J} \end{array} \begin{array}{c} \bullet \\ \diagup \\ \boxed{J} \end{array} \tag{18}$$

Proposition 23. Equation (18) is independent of the control equations of Figure 1 and equations (12), (13), and (14).

Proof. Interpret diagrams in X and J in permutation matrices, by mapping X to the usual NOT gate and J to the identity. All control equations and (12), (13), and (14) hold directly. However, equation (18) does not hold in this model: the left-hand side is a swap, whereas the right-hand side is the identity. $\square$

We show that this independent equation (18) is the unique one necessary for completeness in addition to the mobit equations and control equations.

Theorem 24. The control equations of Figure 1 and equations (12), (13), (14) and (18) are complete for mobit circuits interpreted in $\mathbf{GL}(\mathbb{Z}_2)$.

The same proof strategy as before works. Define functors between the circuit prop obtained with mobit equations as well as control equations, and the prop $\mathbf{GL}(\mathbb{Z}_2)_2$. Observe that $\mathbf{GL}(\mathbb{Z}_2)$ is a bipermutative category, and thus $\mathbf{GL}(\mathbb{Z}_2)_2$ is a crop. Write $\mathbf{J}$ for the crop generated by x and j such that $x^2 = \text{id}_1$, $j^3 = \text{id}_1$, and $jxj = x$. Write $\mathbf{cJ}_{/\sim}$ for the controlled prop out of $\mathbf{J}$ further quotiented by (18).

Similar to Gray code transpositions (see §2.3), we define Gray code J 's to be:

$$\zeta_{n,i} = r_n^{-1} \circ (\text{id}_i \oplus J \oplus \text{id}_{2^n-2-i}) \circ r_n.$$

for which we can also obtain an inductive formulation.

$$\zeta_{n+1,i} = \begin{cases} \zeta_{n,i} \oplus \text{id}_{2^n} & \text{if } i < 2^n - 1, \\ s_{2^n,2} \circ (\text{id}_{2^n} \oplus J \oplus \text{id}_{2^n-2}) \circ s_{2,2^n} & \text{if } i = 2^n - 1, \\ \text{id}_{2^n} \oplus (\theta_{n,2^{n+1}-1-i} \circ \zeta_{n,2^{n+1}-1-i} \circ \theta_{n,2^{n+1}-1-i}) & \text{otherwise.} \end{cases} \quad (19)$$

This allows us to define the following prop morphisms, one direction obtained through the prop and control structure, and the other because morphisms in $\mathbf{GL}(\mathbb{Z}_2)_2$ are generated by X and J , this time in Gray code: $\theta_{n,i}$ and $\zeta_{n,i}$. In the definitions below, the order $<_\ell$ describes the lexicographic order.

$$\begin{array}{ll} A_J: & \mathbf{cJ}_{/\simeq} \rightarrow \mathbf{GL}(\mathbb{Z}_2)_2 \\ & n \mapsto 2^n \\ f \in \mathbf{cX} & \mapsto \alpha(f) \\ j & \mapsto J \\ C^1(f) & \mapsto \text{id}_{2^n} \oplus A_J(f) \\ C^0(f) & \mapsto A_J(f) \oplus \text{id}_{2^n} \\ \text{id}_m + f & \mapsto \text{id}_{2^m} \otimes A_J(f) \end{array} \quad \begin{array}{ll} B_J: & \mathbf{GL}(\mathbb{Z}_2)_2 \rightarrow \mathbf{cJ}_{/\simeq} \\ & 2^n \mapsto n \\ \theta_{n,i} & \mapsto C_{w'_n(i)}^{w_n(i)}(x) \\ \zeta_{n,i} & \mapsto C_{w'_n(i)}^{w_n(i)}(j) \quad \text{if } h_n(i) <_\ell h_n(i+1), \\ \zeta_{n,i} & \mapsto C_{w'_n(i)}^{w_n(i)}(j^2) \quad \text{otherwise.} \end{array}$$

The functor A_J is obviously well-defined for the same reason as Lemma 20 and because (18) reflects (17). We then treat the functor B_J the same way as $\beta: \mathbf{Perm}_2 \rightarrow \mathbf{cX}$.

Lemma 25. *The functor $B_J: \mathbf{GL}(\mathbb{Z}_2)_2 \rightarrow \mathbf{cJ}_{/\simeq}$ is a well-defined crop morphism, and*

$$B_J(\text{id}_{2^n} \oplus f) = C^1(B_J(f)), \quad B_J(f \oplus \text{id}_{2^n}) = C^0(B_J(f)),$$

for all morphisms $f: 2^n \rightarrow 2^n$.

Proof. In order to show that B_J is well-defined, we need to prove that all equations that hold in $\mathbf{GL}(\mathbb{Z}_2)_2$ also hold after image by B_J . To do so, we work with the Lafont representation given earlier in the section. The equations (12), (13), and (14) are straightforward. Equation (15) is the usual Yang-Baxter, therefore holds after applying B_J for the same reason it does for β (see Lemma 10). Similarly to Yang-Baxter for β , well-definedness at (16) follows by induction, where the base case (on two mobits) is:

and the rest of the induction is direct because β is well-defined. A similar reasoning applies to (17), however the base case of the induction cannot be proven simply by the rules of $\mathbf{cJ}$, this is why (18) is added. Additionally, We use the same strategy than in Lemma 10 to prove that B_J is a prop morphism and maps the direct sums to control. $\square$

Because A_J and B_J are well-defined crop morphisms and have inverse images, the categories $\mathbf{cJ}_{/\simeq}$ and $\mathbf{GL}(\mathbb{Z}_2)_2$ are crop isomorphic.

5.4 Controlled-V

An active area of research in quantum computing deals with determining the precise resources necessary for universal quantum computing to emerge [27, 44, 28]. As we have seen, in the classical case, universality emerges from the simplest possible theory of control, namely the one generated by a crop with a single involution $NOT: 1 \rightarrow 1$. A natural question to ask is how this prop must be extended to give a model of universal quantum computation.

Recent work [29] shows that a controlled- V gate, where V is a *square root* of the NOT gate (i.e., satisfies $V \circ V = NOT$), suffices to perform universal quantum computing. This has the surprising consequence that a model of universal quantum computation can be generated from the theory of control for a single operation of order 4.

Theorem 26. Let $(\mathbf{V}, +, 0, V^2)$ be the crop given by the prop presented by a single generator $V : 1 \rightarrow 1$ and relation $V^4 = \text{id}$. Then there exists a prop morphism $\mathbf{cV} \rightarrow \mathbf{Unitary}$ whose image is universal for quantum computing.

Proof. Send V to the usual V -gate $\frac{1}{2} \begin{pmatrix} 1+i & 1-i \\ 1-i & 1+i \end{pmatrix}$, and its controlled variants to the usual (iterated) controlled V -gate. $\square$

While this gives a model of universal quantum computing, its equational theory is not complete (i.e., the functor is not faithful); this can be seen from the fact that the image of V could have been chosen to be the S -gate, which is not universal for quantum computing even given arbitrary control (it generates a theory of *monomial matrices*). The precise relations needed to pin down the theory of controlled- V is an open question.

5.5 Quantum circuits

A complete equational theory for quantum circuits was an open question that was solved only recently [16] with several later improvements [15, 14, 21]. These papers already refer to some equations as *structural*. We show here that these equations are structural in a formal and categorical sense: they are only about control, and follow directly from the structure of a rig category.

Conjugation plays an important role in controlled quantum circuits [21], because of the following equation.

$$\begin{array}{c} \bullet \\ | \\ \text{---} h^{-1} \text{---} f \text{---} h \text{---} \\ | \\ \bullet \end{array} = \begin{array}{c} \text{---} h^{-1} \text{---} f \text{---} h \text{---} \\ | \\ \bullet \end{array} \quad (20)$$

Note that this conjugation equation follows from our control theory as follows.

$$\begin{array}{c} \bullet \\ | \\ \text{---} h^{-1} \text{---} f \text{---} h \text{---} \\ | \\ \bullet \end{array} \stackrel{(a)}{=} \begin{array}{c} \bullet \quad \bullet \quad \circ \quad \circ \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \text{---} h^{-1} \text{---} f \text{---} h^{-1} \text{---} h \text{---} h \text{---} \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \circ \quad \circ \quad \bullet \end{array} \stackrel{(f)}{=} \begin{array}{c} \bullet \quad \circ \quad \bullet \quad \circ \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \text{---} h^{-1} \text{---} h^{-1} \text{---} f \text{---} h \text{---} h \text{---} \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \circ \quad \circ \quad \bullet \end{array} \stackrel{(e)}{=} \begin{array}{c} \bullet \\ | \\ \text{---} h^{-1} \text{---} f \text{---} h \text{---} \\ | \\ \bullet \end{array}$$

Let $(\mathbf{Z}, +, 0)$ be the prop generated by $\alpha : 0 \rightarrow 0$ for $\alpha \in \mathbb{R}$, and $Z(\alpha) : 1 \rightarrow 1$ for $\alpha \in \mathbb{R}$, and $H : 1 \rightarrow 1$ satisfying

$$\textcircled{2\pi} = \square \quad (21)$$

$$\textcircled{\alpha_1} \textcircled{\alpha_2} = \textcircled{\alpha_1 + \alpha_2} \quad (22)$$

$$\text{---} \boxed{Z(\alpha_1)} \text{---} \boxed{Z(\alpha_2)} \text{---} = \text{---} \boxed{Z(\alpha_1 + \alpha_2)} \text{---} \quad (23)$$

$$\text{---} \boxed{H} \text{---} \boxed{H} \text{---} = \text{---} \quad (24)$$

$$\text{---} \boxed{H} \text{---} \boxed{Z(\alpha_1)} \text{---} \boxed{H} \text{---} \boxed{Z(\alpha_2)} \text{---} \boxed{H} \text{---} = \text{---} \boxed{Z(\beta_1)} \text{---} \boxed{H} \text{---} \boxed{Z(\beta_2)} \text{---} \boxed{H} \text{---} \boxed{Z(\beta_3)} \text{---} \textcircled{\beta_0} \quad (25)$$

where (25) is the well-known Euler decomposition, in which $\beta_0, \beta_1, \beta_2$ and β_4 can be computed from α_1 and α_2 deterministically. We choose the crop $(\mathbf{Z}, +, 0, HZ(\pi)H)$ and can now form its controlled prop $\mathbf{cZ}$. Interestingly, the prop $\mathbf{cZ}$ is not immediately the prop of unitary operations: for a given α , the morphisms $C^1(\alpha)$ and $Z(\alpha)$ have the same semantics in unitaries but are still different in $\mathbf{cZ}$. We need to quotient further with the following equation:

$$\text{---} \boxed{Z(\alpha)} \text{---} = \text{---} \bullet \text{---} \textcircled{\alpha} \quad (26)$$

and we write $\mathbf{cZ}_{/\simeq}$ for this category. From this point, it is simple to show that we have equivalent equations to the complete equational theory for quantum circuits [21], and therefore $\mathbf{cZ}_{/\simeq}$ is isomorphic to the category of unitaries on qubits.

Theorem 27. Equations (21), (22), (23), (24), (25) together with the control equations of Figure 1 and equation (26), are sound and complete for quantum circuits.

Proof. The set of generators is universal, and it suffices to show that all the required equations follow from ours. The circuit equations from the literature [21] contain similar equations to (21), (22), (23), (24), and (25), given that equation (26) holds. Moreover, we have proven that controlled conjugation (20) holds in our control theory. This is enough to cover the set of equations for quantum circuits [21]. $\square$

In fact, the same general strategy works to obtain a complete equational theory for quantum circuits formed from certain discrete gate sets by exploiting the rig structure shown in [11]. Let $(\mathbf{\Pi}, +, 0)$ be the prop generated by $\omega : 0 \rightarrow 0$, $V : 1 \rightarrow 1$, and $S : 1 \rightarrow 1$ satisfying

$$\omega \omega \omega \omega \omega \omega \omega \omega = \boxed{} \quad (27)$$

$$-\boxed{V}-\boxed{V}-\boxed{V}-\boxed{V}- = - \quad (28)$$

$$-\boxed{S}-\boxed{V}-\boxed{S}- = -\boxed{V}-\boxed{S}-\boxed{V}- \quad (29)$$

Choose now the crop $(\mathbf{\Pi}, +, 0, V^2)$, form its controlled prop $\mathbf{c\Pi}$, and quotient it further by the equation

$$-\boxed{S}- = -\omega \omega \quad (30)$$

to obtain the category $\mathbf{c\Pi}/\simeq$. Recall that the *Gaussian Clifford+T* gate set [2] consists of the gates S , K , X , CX , and CCX , where $K = \frac{1-i}{\sqrt{2}}H$, and H is the usual Hadamard gate. We then obtain the desired result:

Theorem 28. *Equations (27), (28), (29), together with the control equations of Figure 1 and equation (30), are sound and complete for Clifford circuits, Clifford+T circuits with ≤ 2 qubits, and Gaussian Clifford+T circuits.*

Proof. Establishing the required completeness results in [11] only required $\omega^8 = \text{id}$ (which follows by (27)), $V^2 = \gamma_{1,1}$ (which follows by (28) and that fact that V^2 is the chosen involution), and $SVS = VSV$ where $S = \text{id} \oplus \omega^2$ (which follow from (29) and (30)), as well as the axioms of rig categories, which are implied by the control equations of Figure 1 by Theorem 22. $\square$

6 Conclusion

Our results raise several interesting questions, that we leave to future work.

- The control equations of Figure 1 have natural interpretations that seem to indicate they are independent. But is this set of equations indeed minimal? Can we find models that satisfy all but one of the control equations?
- Several circuit optimisations in the literature rely on auxiliary wires in a circuit to enact controlled operations [14, 15], and other works also make use of ancillas to obtain complete equational theories for CNOTs [20] and Toffolis [19]. How do these relate to rigged crops?
- There is no quantum circuit implementation of a controlled unitary where the unitary is a black box input [3]. Yet there are many physical implementations bypassing the strict confines of this no-go theorem [24, 8]. The latter rely on the ability to identify subspaces of qubits by adding auxiliary dimensions, harmonising with our results. Is there a version of the construction of Theorem 5 that allows for auxiliary wires?
- We have isolated the control aspects from the rest of the computation. Can this be used to analyse causal dependencies within the data flow of the computation?
- We have worked out examples adding computational control to several props such as a single X gate, a single $\sqrt{X}$ gate, and an X gate together with a Hadamard gate. What does the controlled prop look like for other concrete base circuit theories?
- The control equations of Figure 1 implicitly assume only two possibilities on each wire: positive and negative control. Can we extend the theory from two-valued data like bits or qubits to data with multiple values like qutrits?

- We have worked with bipermutative categories, where the fact that $\oplus$ is symmetric corresponds to the fact $x^2 = 1$ that the NOT gate is an involution. Are there interesting theories when we generalise to $\oplus$ being merely braided?
- Boolean functions are fundamental to bounded decision diagrams and SAT-solving. Do our results imply anything practically useful in those applications?

Acknowledgements

The authors would like to thank the members of the quantum programming group, specifically Wang Fang, and of the category theory group at Edinburgh for their support and feedback. We extend our thanks to Robert Booth, Kostia Chardonnet, Noé Delorme, Nicolas Heurtel, Simon Perdrix, and Peter Selinger for comments and fruitful discussions that helped us improve the paper. This research was funded by the Engineering and Physical Sciences Research Council (EPSRC) under project EP/X025551/1 “Rubber DUQ: Flexible Dynamic Universal Quantum programming”. Robin Kaarsgaard was supported by Sapere Aude: DFF-Research Leader grant 5251-00024B.

References

- [1] N. Abdessaied, M. Amy, R. Drechsler, and M. Soeken. Complexity of reversible circuits and their quantum implementations. *Theoretical Computer Science*, 618:85–106, 2016.
- [2] M. Amy, A. N. Glaudell, and N. J. Ross. Number-theoretic characterizations of some restricted clifford+T circuits. *Quantum*, 4:252, 2020.
- [3] M. Araújo, F. Adrien, F. Costa, and Č. Brukner. Quantum circuits cannot control unknown operations. *New Journal of Physics*, 16(9):093026, 2014.
- [4] S. Balaucă and A. Arusoae. Efficient constructions for simulating multi controlled quantum gates. In *International Conference on Computer Science*, volume 13353 of *Lecture Notes in Computer Science*, pages 179–194. Springer, 2022.
- [5] A. Barenco, C. H. Bennett, R. Cleve, D. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Physical Review A*, 5:3457–3467, 1995.
- [6] X. Bian and P. Selinger. Generators and relations for $U_n(\mathbb{Z}[\frac{1}{2}, i])$. In *Quantum Physics and Logic*, volume 343 of *Electronic Proceedings in Theoretical Computer Science*, pages 145–164, 2021.
- [7] X. Bian and P. Selinger. Generators and relations for 2-qubit Clifford+T operators. *Electronic Proceedings in Theoretical Computer Science*, 394:13–28, 2023.
- [8] A. Bisio, M. Dall’Arno, and P. Perinotti. Quantum conditional operations. *Physical Review A*, 94:022340, 2016.
- [9] F. Bonchi, B. Sobociński, and F. Zanasi. Interacting Hopf algebras. *Journal of Pure and Applied Algebra*, 221(1):144–184, 2017.
- [10] J. Carette, C. Heunen, R. Kaarsgaard, and A. Sabry. The quantum effect: A recipe for quantum-II. In *Proceedings of the ACM on Programming Languages*, volume 8, pages 1–29, 2024.
- [11] J. Carette, C. Heunen, R. Kaarsgaard, and A. Sabry. With a few square roots, quantum computing is as easy as II. *Proceedings of the ACM on Programming Languages*, 8, 2024.
- [12] J. Carette and A. Sabry. Computing with semirings and weak rig groupoids. In *European Symposium on Programming*, volume 9632 of *Lecture Notes in Theoretical Computer Science*, pages 123–148. Springer, 2016.
- [13] Z. Chen, W. Liu, Y. Ma, W. Sun, R. Wang, H. Wang, H. Xu, G. Xue, H. Yan, Z. Yang, J. Ding, Y. Gao, F. Li, Y. Zhang, Z. Zhang, Y. Jin, H. Yu, J. Chen, and F. Yan. Efficient implementation of arbitrary two-qubit gates using unified control. *Nature Physics*, 21:1489–1496, 2025.

- [14] A. Clément, N. Delorme, and S. Perdrix. Minimal equational theories for quantum circuits. In *Logic in Computer Science*, pages 27:1–27:14. ACM/IEEE, 2024.
- [15] A. Clément, N. Delorme, S. Perdrix, and R. Vilmart. Quantum circuit completeness: Extensions and simplifications. In *Computer Science Logic*, volume 288 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:23, 2024.
- [16] A. Clément, N. Heurtel, S. Mansfield, S. Perdrix, and B. Valiron. A complete equational theory for quantum circuits. In *Logic in Computer Science*, pages 1–13. ACM/IEEE, 2023.
- [17] A. Clément and S. Perdrix. PBS-calculus: A graphical language for coherent control of quantum computations. In *Mathematical Foundations of Computer Science*, volume 170 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:14, 2020.
- [18] A. Clément and S. Perdrix. Resource optimisation of coherently controlled quantum computations with the pbs-calculus. In *Mathematical Foundations of Computer Science*, volume 241 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 36:1–36:15, 2022.
- [19] J.R.B. Cockett and C. Comfort. The category TOF. *Electronic Proceedings in Theoretical Computer Science*, 287:67–84, 2019.
- [20] R. Cockett, C. Comfort, and P. Srinivasan. The category cnot. *Electronic Proceedings in Theoretical Computer Science*, 266:258–293, 2018.
- [21] N. Delorme and S. Perdrix. Diagrammatic reasoning with control as a constructor, applications to quantum circuits, 2025. arXiv:2508.21756.
- [22] J. M. Erbele. *Categories in control: applied PROPs*. PhD thesis, University of California, Riverside, 2016.
- [23] W. Fang, C. Heunen, and R. Kaarsgaard. Hadamard-II: Equational quantum programming. arXiv:2506.06835, 2025.
- [24] N. Friis, V. Dunjko, W. Dür, and H. J. Briegel. Implementing quantum control for unknown subroutines. *Physical Review A*, 89:030303, 2014.
- [25] D. R. Ghica and A. Jung. Categorical semantics of digital circuits. In *Formal Methods in Computer-Aided Design*, pages 41–48, 2016.
- [26] C. Heunen and R. Kaarsgaard. Quantum information effects. *Proceedings of the ACM on Programming Languages*, 6, 2022.
- [27] M. Howard, J. Wallman, V. Veitch, and J. Emerson. Contextuality supplies the ‘magic’ for quantum computation. *Nature*, 510:351–355, 2014.
- [28] R. Jozsa and N. Linden. On the role of entanglement in quantum-computational speed-up. *Proceedings of the Royal Society of London A*, 459:2011–2032, 2003.
- [29] R. Kaarsgaard. All you need is controlled-V: universality of a standard two-qubit gate by catalytic embedding. arXiv:2509.07578, 2025.
- [30] Y. Lafont. Towards an algebraic theory of boolean circuits. *Journal of Pure and Applied Algebra*, 184(2):257–310, 2003.
- [31] M. L. Laplaza. Coherence for distributivity. In G. M. Kelly, M. Laplaza, G. Lewis, and Saunders Mac Lane, editors, *Coherence in Categories*, pages 29–65. Springer, 1972.
- [32] S. M. Li, N. J. Ross, and P. Selinger. Generators and relations for the group $O_n(\mathbb{Z}[1/2])$. In *Quantum Physics and Logic*, volume 343 of *Electronic Proceedings in Theoretical Computer Science*, pages 210–264, 2021.
- [33] S. Mac Lane. Categorical algebra. *Bulletin of the American Mathematical Society*, 71:40–106, 1965.
- [34] J. P. May. *E_∞ Ring Spaces and E_∞ Ring Spectra*, volume 577 of *Lecture Notes in Mathematics*. Springer-Verlag, 1977.

- [35] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.
- [36] J. H. H. Perk and H. Au-Yang. Yang–Baxter equations. In *Encyclopedia of Mathematical Physics*, pages 465–473. Academic Press, 2006.
- [37] B. Schumacher and M. D. Westmoreland. Modal quantum theory. In *Quantum Physics and Logic*, pages 145–151, 2010.
- [38] R. Sharma and S. Archour. Optimizing ancilla-based quantum circuits with SPARE. *Proceedings of the ACM on Programming Languages*, 9, 2025.
- [39] V. V. Shende, S. S. Bullock, and I. L. Markov. Synthesis of quantum logic circuits. In *Asia and South Pacific Design Automation Conference*, pages 272–275. ACM, 2005.
- [40] Y. Shi. Both Toffoli and controlled-NOT need little help to do universal quantum computing. *Quantum Information and Computation*, 3(1):84–92, 2003.
- [41] T. Sleator and H. Weinfurter. Realizable universal quantum logic gates. *Physical Review Letters*, 74:4087–4090, 1995.
- [42] M. K. Thomsen, R. Kaarsgaard, and M. Soeken. Ricercar: a language for describing and rewriting reversible circuits with ancillae and its permutation semantics. In *Reversible Computing*, pages 200–215, 2015.
- [43] T. Toffoli. Reversible computing. In *International Colloquium on Automata, Languages, and Programming*, Lecture Notes in Computer Science, pages 632–644. Springer, 1980.
- [44] V. Vedral. The elusive source of quantum speedup. *Foundations of Physics*, 40:1141–1154, 2010.
- [45] H. Vollmer. *Introduction to circuit complexity*. Springer, 1999.
- [46] D. Yau. *Bimonoidal Categories, E_n -Monoidal Categories, and Algebraic K-Theory: Volume I: Symmetric Bimonoidal Categories and Monoidal Bicategories*, volume 283 of *Mathematical Surveys and Monographs*. American Mathematical Society, 2024.
- [47] N. Yu, R. Duan, and M. Ying. Five two-qubit gates are necessary for implementing the Toffoli gate. *Physical Review A*, 88:010304, 2013.

A Definitions

Definition 29 (Bipermutative category). A *bipermutative category* $(\mathbf{C}, \otimes, I, \oplus, O)$ is a category such that $(\mathbf{C}, \otimes, I)$ and $(\mathbf{C}, \oplus, O)$ are strict symmetric monoidal categories, and the following conditions are satisfied:

- for all objects A , we have $O \otimes A = O = A \otimes O$; for all morphisms $f: A \rightarrow B$, we have $f \otimes \text{id}_O = \text{id}_O \otimes f$;
- for all objects A, B, C and morphisms f, g, h , we have that

$$(A \oplus B) \otimes C = (A \otimes C) \oplus (B \otimes C) \quad (31)$$

and

$$(f \oplus g) \otimes h = (f \otimes h) \oplus (g \otimes h) \quad (32)$$

and the following diagram commutes:

$$\begin{array}{ccc} (A \oplus B) \otimes C & \xrightarrow{=} & (A \otimes C) \oplus (B \otimes C) \\ \downarrow \gamma \otimes \text{id} & & \downarrow \gamma \\ (B \oplus A) \otimes C & \xrightarrow{=} & (B \otimes C) \oplus (A \otimes C) \end{array} \quad (33)$$

- if we define a natural left distributor δ as:

$$A \otimes (B \oplus C) \xrightarrow{s} (B \oplus C) \otimes A \xrightarrow{=} (B \otimes A) \oplus (C \otimes A) \xrightarrow{s \oplus s} (A \otimes B) \oplus (A \otimes C) \quad (34)$$

then the following diagram commutes for all objects A, B, C, D :

$$\begin{array}{ccc} (A \oplus B) \otimes (C \oplus D) & \xrightarrow{=} & (A \otimes (C \oplus D)) \oplus (B \otimes (C \oplus D)) \\ \downarrow \delta & & \downarrow \delta \oplus \delta \\ ((A \oplus B) \otimes C) \oplus ((A \oplus B) \otimes D) & \xrightarrow{=} & (A \otimes C) \oplus (A \otimes D) \oplus (B \otimes C) \oplus (B \otimes D) \\ & \searrow & \downarrow \text{id} \oplus \gamma \oplus \text{id} \\ & & (A \otimes C) \oplus (B \otimes C) \oplus (A \otimes D) \oplus (B \otimes D) \end{array} \quad (35)$$

A *strict bipermutative functor* is a functor which is a strict symmetric monoidal functor with respect to both symmetric monoidal structures $(C, \otimes, I)$ and $(C, \oplus, O)$.