



HAL
open science

Compositionality of planar perfect matchings

Titouan Carette, Etienne Moutot, Thomas Perez, Renaud Vilmart

► **To cite this version:**

Titouan Carette, Etienne Moutot, Thomas Perez, Renaud Vilmart. Compositionality of planar perfect matchings: A universal and complete fragment of ZW-calculus. 50th International Colloquium on Automata, Languages, and Programming (ICALP 2023), Jul 2023, Paderborn, Germany. pp.120:1–120:17, 10.4230/LIPIcs.ICALP.2023.120 . hal-04002282

HAL Id: hal-04002282

<https://hal.science/hal-04002282>

Submitted on 23 Feb 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Compositionality of planar perfect matchings

A universal and complete fragment of ZW-calculus

Titouan Carette¹, Etienne Moutot², Thomas Perez³, and Renaud Vilmart⁴

¹titouan.carette@lu.lv, Centre for Quantum Computer Science, Faculty of Computing, University of Latvia, Raina 19, Riga, Latvia, LV-1586

²etienne.moutot@math.cnrs.fr, CNRS, I2M, Aix-Marseille Université, Marseille, France

³thomas.perez@ens-lyon.fr, Université de Lyon, ENS de Lyon, 69007 Lyon, France

⁴vilmart@lsv.fr Université Paris-Saclay, ENS Paris-Saclay, Inria, CNRS, LMF, 91190, Gif-sur-Yvette, France

Abstract

We exhibit a strong connection between the matchgate formalism introduced by Valiant and the ZW-calculus of Coecke and Kissinger. This connection provides a natural compositional framework for matchgate theory as well as a direct combinatorial interpretation of the diagrams of ZW-calculus through the perfect matchings of their underlying graphs.

We identify a precise fragment of ZW-calculus, the planar W-calculus, that we prove to be complete and universal for matchgates, that are linear maps satisfying the matchgate identities. Computing scalars of the planar W-calculus corresponds to counting perfect matchings of planar graphs, and so can be carried in polynomial time using the FKT algorithm, making the planar W-calculus an efficiently simulable fragment of the ZW-calculus, in a similar way that the Clifford fragment is for ZX-calculus. This work opens new directions for the investigation of the combinatorial properties of ZW-calculus as well as the study of perfect matching counting through compositional diagrammatical technics.

1 Introduction

A quantum computation mapping n qubits to m qubits corresponds to an isometric linear map $\mathbb{C}^{2^n} \rightarrow \mathbb{C}^{2^m}$. Due to the exponential size of their matrix representation, those linear maps are traditionally depicted as quantum circuits, an assemblage of elementary quantum gates similar to the more common boolean circuits. Given a quantum circuit $n \rightarrow m$, evaluating a coefficient of the corresponding $2^m \times 2^n$ matrix (i.e. evaluating the circuit with a given input) typically requires an exponential time. However, there are some specific classes of quantum circuits – or fragments –, that can be classically simulated in polynomial time. Examples are the Clifford fragment (as asserted by the Gottesman-Knill theorem) as well as the fragment that will particularly interest us in this paper, the nearest-neighbour matchgates [24]. Investigating those tractable fragments allows a better understanding of the computational advantage of quantum computing. The reference for all elementary results on quantum circuits is [19].

Taking the diagrammatical circuit representation seriously led to developing graphical languages for quantum computing [8]. Those languages are equational theories described by elementary gates and local identities between diagrams. Such languages come with an

interpretation into linear maps. A language is said universal for a class of linear maps if any linear map in the class is the interpretation of a diagram in the language. A language is said complete if two diagrams with the same interpretation are equivalent up to the equational theory, which means that they can be rewritten from one to the other using the local rewriting rules of the equational theory. In general, completeness is the most challenging property to prove.

The first quantum graphical language to appear was the ZX-calculus in 2008 [8]. It was rapidly known to be universal for all linear maps. However, providing a complete set of rewriting rules took another ten years (see [26] for an history of completeness) and first required a translation through another language, the ZW-calculus [18].

The ZW-calculus was introduced in [9] as a graphical representation of the two kinds of tripartite entanglement for qubits, namely the GHZ-states and W-states. It then appeared that this calculus had very nice algebraic properties allowing the internal encoding of arithmetical operations. Those properties allowed the ZW-calculus to be the first proven universal and complete language for linear maps [13]. Despite this historical importance, the ZW-calculus gathered less attention than other languages, seen as more connected to quantum computing. Still, we must mention interesting connections with fermionic quantum computing [12], and recent works importing some ZW-calculus primitives into ZX-calculus to exploit their algebraic properties [20, 28]. In this paper, we show that ZW-calculus has very strong connections with a specific family of quantum circuits: the matchgates.

Matchgates were introduced in 2002 by Valiant [24]. They are linear maps defined by counting the perfect matching of a graph from which we remove some vertices depending on the inputs. This underlying combinatorial structure allows to classically simulate the corresponding quantum circuits by using the Polynomial FKT algorithm for perfect matchings counting [1, 22]. The theory of matchgates was then developed further to the concept of holographic algorithms [25]. We can notice that if some connections between graphical languages and holographic algorithms have been investigated [3], we are not aware of any diagrammatical approach to the original concept of matchgate before the present work, except a mention in [12].




The main contribution of this paper is the introduction of a fragment of the ZW-calculus, that we call planar W-calculus. We show that this language is universal and complete for the planar matchgate fragment of quantum computation. The completeness proof relies on designing a normal form and a rewriting strategy to reach it. We also define a pro of matchgate computations by showing the compositionality of the matchgate identities introduced in [5]. The combinatorial characterisation of matchgate computations then directly follows from the correspondence with the graphical language. Hence one can see this paper as a reformulation of matchgate theory in a compositional framework.

The paper is structured as follows. Section 2 introduces our graphical primitives, their interpretation as linear maps and their combinatorial properties: the interpretation of a diagram can be deduced by counting the number of perfect matching of the underlying weighted graph. We present the generators and elementary rewrite rules of the language as well as an essential syntactic sugar: the fermionic swap that emulates the swap gate, which is not part of our language. Section 3 introduces the normal form and proves the completeness of the language. In Section 4, we properly define a pro of matchgates characterised as the linear maps satisfying the matchgate identities. We show that our language is universal for matchgates, *i.e.*, that the interpretation of a diagram is always a matchgate and that all matchgates correspond to a diagram. Finally, in Section 5, we sketch future directions of research suggested by the connection we identified between ZW-calculus and perfect matching counting.

2 Perfect Matchings and Planar W-Calculus

We define our fragment of the ZW-calculus, the *planar W-calculus*, by defining its diagrams. Any diagram with n inputs and m outputs $D : n \rightarrow m$ is interpreted as a linear map $\llbracket D \rrbracket : \mathbb{C}^{2^n} \rightarrow \mathbb{C}^{2^m}$ inductively as follows:

$$\begin{aligned} \llbracket \begin{array}{|c|c|} \hline D_1 & D_2 \\ \hline \end{array} \rrbracket &:= \llbracket D_1 \rrbracket \otimes \llbracket D_2 \rrbracket & \llbracket \begin{array}{|c|} \hline D_1 \\ \hline D_2 \\ \hline \end{array} \rrbracket &:= \llbracket D_2 \rrbracket \circ \llbracket D_1 \rrbracket \\ \llbracket \square \rrbracket &:= (1) & \llbracket \begin{array}{|c|} \hline | \\ \hline \end{array} \rrbracket &:= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \llbracket \cup \rrbracket &:= (1 \ 0 \ 0 \ 1) & \llbracket \cap \rrbracket &:= \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \end{aligned}$$

In particular, note that we do *not* use the usual swap diagram , hence the name *planar*. We do have, however, the so-called cup  and cap  satisfying the ‘‘snake equations’’:

$$\begin{array}{c} \cup \\ \cup \end{array} = \begin{array}{|c|} \hline | \\ \hline \end{array} = \begin{array}{c} \cap \\ \cap \end{array}$$

In the following, with $D : n \rightarrow n$, we may use the following notation: $D^{\otimes \vec{b}}$ when \vec{b} is a bitstring, to represent $D^{b_1} \otimes \dots \otimes D^{b_n}$ with $D^0 = id_n$ and $D^1 = D$. We call a diagram D a scalar if it has no input and no output, i.e. $D : 0 \rightarrow 0$. In the category-theoretic terminology, such a collection of diagrams defines a pro, a strict monoidal category whose monoid of objects is generated by a unique element, and not a prop, which requires the category to be symmetric, *i.e.* to have swap diagrams. Furthermore, the presence of the cups and caps make the category a compact-closed pro. We define **Qubit** to be the prop whose $n \rightarrow m$ morphisms are linear maps $\mathbb{C}^{2^n} \rightarrow \mathbb{C}^{2^m}$. Hence $\llbracket \cdot \rrbracket : \mathbf{pW} \rightarrow \mathbf{Qubit}$ is a pro morphism.

We add the two following generators: the black spider and the binary white spider, whose interpretations are detailed in the next sub-sections.

2.1 Black Spider

To manipulate binary words $\alpha \in \{0, 1\}^n$ and $\beta \in \{0, 1\}^m$, we will denote $\alpha \oplus \beta \in \{0, 1\}^n$ the bitwise XOR (if $n = m$), $\alpha \cdot \beta \in \{0, 1\}^{n+m}$ the concatenation, $|\alpha| \in \{0, \dots, n\}$ the Hamming weight, *i.e.*, the number of ones in the word α , and $|\alpha|_2 \in \{0, 1\}$ the parity of this weight, 0 if even and 1 if odd. The *black spider* (or black node) is given by the following interpretation:

$$\llbracket \begin{array}{|c|} \hline \dots \\ \hline \cup \\ \hline \dots \\ \hline \end{array} \rrbracket := \sum_{\substack{u \in \{0,1\}^m \\ v \in \{0,1\}^n \\ |uv|=1}} |u\rangle\langle v|$$

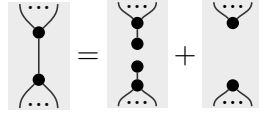
In other words, the black spiders gives an output 1 if and only if exactly one of its legs (either input or outputs) has value |1> and all the others |0>. As inputs and outputs behave exactly the same, one can use cup and caps in order to transform inputs into outputs and vice-versa:

$$\begin{array}{c} \cup \\ \cup \end{array} = \begin{array}{c} \cap \\ \cap \end{array}$$

Moreover, as input order do not matter, one can bend the wires and move black spiders around, without altering the resulting linear map, we say that the black nodes are flexsymmetric [7]. Flexsymmetry of the black spider allows us to see diagrams as graphs with fixed

inputs and outputs edges. Fixing the input and outputs edges, any graph isomorphism preserves the semantics.

With this graphical interpretation in mind, one can understand the interpretation of a scalar diagram, composed of only black spiders, as counting the number of perfect matchings in the underlying graph. To see this, one can use the interpretation of a single edge, which simply is the identity $|0\rangle\langle 0| + |1\rangle\langle 1|$. This interpretation gives a useful insight in the diagrams: given an edge, one can partition the set of perfect matchings between those that have this edge and those that don't:



In the case where the graph is an actual graph, without half edges, the resulting map is a scalar (no input or outputs). One can show by induction that this scalar corresponds to the number of ways of choosing a set of edges such that each vertex is covered by exactly one edge. In other words, *the number of perfect matchings* of the graph.

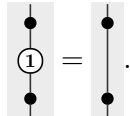
2.2 Binary White Spider

The last generator of the planar W-calculus is the *binary white spider*, given, for any $r \in \mathbb{C}$, by:

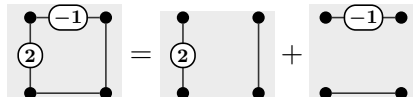
$$\left[\begin{array}{c} \text{---} \\ | \\ \textcircled{r} \\ | \\ \text{---} \end{array} \right] := \begin{pmatrix} 1 & 0 \\ 0 & r \end{pmatrix}$$

which corresponds to the usual binary white spider with weight r of the ZW-calculus. This binary spider corresponds to having a weight r on an edge of the graph. When $r \in \mathbb{N}$, the

interpretation is straightforward: the white spider can be replaced by r edges: .

And in particular, .

Let us interpret the white spiders as weights on the edges of a planar graph G with black spiders on their vertices. Consider one perfect matching of the same graph G' without weights and consider one perfect matching P of G' . If the edge e that belongs to P has a weight $r \in \mathbb{N}$, then it can be replaced by r edges. In other words, the single perfect matching P is replaced by r perfect matchings when e has weight r . By doing this for every edges, one can see that each perfect matching in G' corresponds to a perfect matching of G with a *weight* that is the product of all its edge weights, instead of weight 1 in G' . For $r \in \mathbb{C}$, one cannot replace a white spider by a given number of edges, but the interpretation is the same: the edge contribute to the perfect matchings that contain it with a *weight* r .

Example 2.1.  = 2 - 1 = 1

Diagrams generated by the black and binary white node, within the framework described at the beginning of the section, are called **pW**-diagrams.

2.3 The FKT Algorithm

In general, counting the number of perfect matchings in a graph is an #P-complete problem [23]. However, for planar graphs the same problem turns out to be surprisingly easy, as

Fisher, Temperley and Kastelyn showed that it is in P [14, 22]. The main idea behind the algorithm is that for planar graphs, it is possible to find a good orientation of the edges (called a Pfaffian orientation) in polynomial time such that the number of perfect matchings is the Pfaffian of the adjacency matrix A (actually its skew-symmetric version, called Tutte matrix) of the oriented graph. A result due to Cayley then shows that the Pfaffian is the square root of the determinant of A .

Note that one can find such an orientation for any planar graph, even weighted with complex weights, and the equality $pf(A) = \sqrt{\det(A)}$ still holds. Therefore, computing the total *weight* of perfect matchings in a complex-weighted graph is in P.

Proposition 2.2. *Let D be a scalar pW-diagram. Then $\llbracket D \rrbracket$ is computable in polynomial time in the number of black nodes.*

2.4 Fermionic Swap

The usual ZW-calculus does have another generator that we did not explicitly include in our fragment, called the *fermionic swap*:

$$\llbracket \text{fermionic swap} \rrbracket := \sum_{x,y \in \{0,1\}} (-1)^{xy} |x\rangle\langle y|$$

However, it turns out that the fermionic swap is just syntactic sugar, and it is actually in our fragment:

Notice that the previous equation also appears in [6] to relate planar and non-planar matchgates. It is very useful to treat this piece of diagram as a generator of its own, especially as a particular kind of swap, which shares a lot of (but not all) properties of the symmetric braiding of props. In particular:

Where $|D|$ is the number of black nodes in the diagram D .

3 Completeness

The planar W-calculus is introduced with an equational theory, given in Figure 1, relating together diagrams with the same semantics. We write $\text{pW} \vdash D_1 = D_2$ when one can turn diagram D_1 into diagram D_2 by applying the equations of Figure 1 locally.

Proposition 3.1. *The equational theory of Figure 1 preserves the semantics:*

$$\text{pW} \vdash D_1 = D_2 \implies \llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket$$

In the following, we will show that the converse also holds, that is, that whenever two diagrams have the same semantics, they can be turned into one another using the equational theory. Intuitively, this implies that the equational theory completely captures the interaction of generators with one another in the fragment.

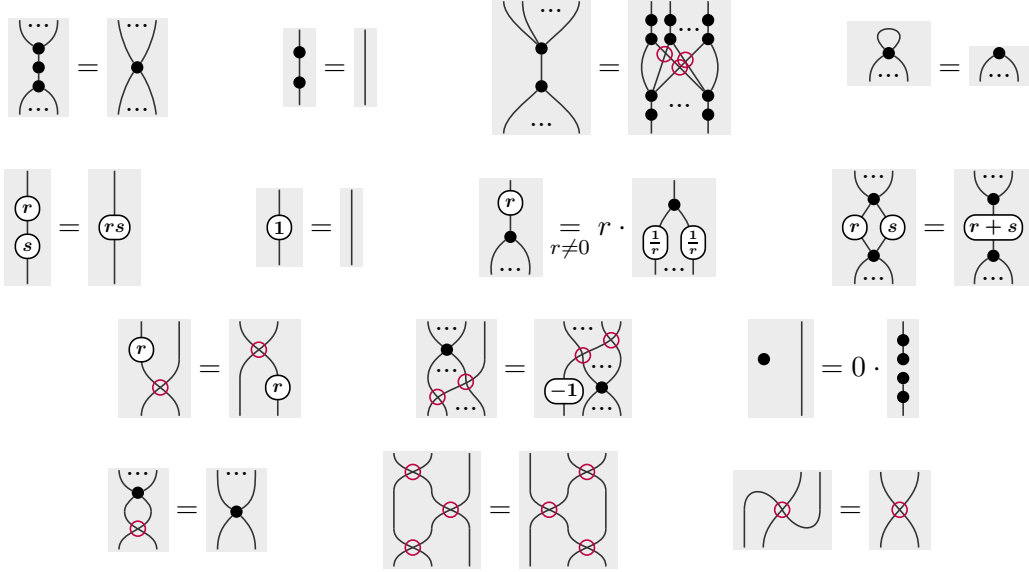


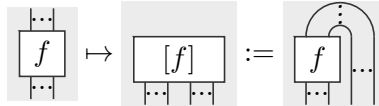
Figure 1: Axioms of the planar W-calculus.

To show this result, we give a notion of normal form, which we call W-graph-state with X-gates (WGS-X for short), then a refinement of that normal form (reduced WGS-X form) which can be shown to be unique, and we give a rewrite strategy (derivable from the equational theory) to turn any \mathbf{pW} -diagram into this form.

3.1 Normal Form

The first step we take towards defining a normal form is a simplification, making use of the compact structure of the underlying pro, where we relate maps and states:

Proposition 3.2. *There is an isomorphism between $\mathbf{pW}(n, m)$ and $\mathbf{pW}(0, n + m)$ defined as such:*




This isomorphism allows us only to consider states rather than maps in the following. Then, we define W-graph-states, by first defining ordered weighted graphs:



Definition 3.1 (Ordered R -Weighted Graph). $G = (V, E, w)$ is called an ordered R -weighed graph if:

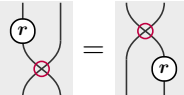

- ▷ V is a set endowed with a total order \prec (or equivalently a sequence)
- ▷ $E \subset V \times V$ is such that $(u, v) \in E \implies u \prec v$
- ▷ $w : E \rightarrow R \setminus \{0\}$ maps each edge to its weight

Definition 3.2 (W-Graph-State). Let $G = (V, E, w)$ be an ordered weighted graph. Then, $\text{WGS}(G)$ is defined as the \mathbf{pW} -diagram where:

- ▷ Each vertex in V gives a W-spider linked to an output through an additional \bullet (the order on V gives the order of the outputs)
- ▷ Each (weighted) edge (u, v) gives a white dot with parameter $w((u, v))$ linked to the W-spiders obtained from u and v

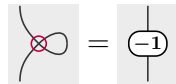
- ▷ All wire crossings in $\text{WGS}(G)$ are fermionic swaps 
- ▷ No output wire crosses another wire
- ▷ There are no self-intersecting wires

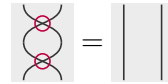
When an edge has weight 1 we may ignore the white dot and represent the edge as a simple wire, since  = . Notice that there are several ways to build $\text{WGS}(G)$, but all of

them are equivalent thanks to  and the axioms on the fermionic swap , together with the provable identities in Lemmas 3.3 and 3.4:



Lemma 3.3.

Lemma 3.4.





Definition 3.3 (WGS-X form). We say that a pW-state D on n qubits is in:

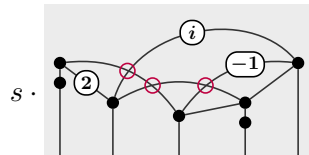
- ▷ **WGS-X form** if there exist $s \in \mathbb{C}$, $G = ([1, n], E, w)$ an ordered graph, and $\vec{b} \in \{0, 1\}^n$ such that $D = s \cdot \left(\bullet^{\otimes \vec{b}} \right) \circ \text{WGS}(G)$.
- ▷ **pseudo-WGS-X form** if it is in WGS-X form with potentially vertices linked to several outputs, additional  ($r \neq 0$) on wires that do not correspond to edges in the graph, and potentially fermionic swaps  between outputs.
- ▷ **reduced WGS-X form** (rWGS-X) if it is in WGS-X form and:



$$\forall i, (b_i = 0 \implies \nexists j, (i, j) \in E)$$

i.e. $b_i = 0$ is only possible if vertex i has no neighbour on its right.

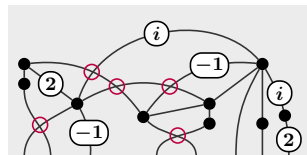
Example 3.5. $\text{WGS} \left(\begin{array}{c} \text{graph with vertices } 1, 2, \dots, i, \dots \\ \text{edges with weights } 2, -1, \dots \end{array} \right) = \text{WGS-X diagram}$ where in the starting graph, vertices are ordered left to right, and edges with no indication of weight have weight 1.

If $\vec{b} = (0, 1, 1, 0, 1)$, then the obtained WGS-X state is:




where we used the fact that  is an involution to simplify the diagram. The WGS-X state is however not reduced, as both the first and fourth qubits have additional  applied to them, but still have neighbours on their right.

Finally, the following diagram is an example of a pseudo-WGS-X state:



3.2 Rewrite Strategy

We define in this section a rewrite strategy, derived from the equational theory, that will terminate in a normal form (WGS-X). Doing this naively is made difficult by the potential presence of fermionic swaps  wherever we are looking for patterns to rewrite. Thankfully, the last 5 equations in Figure 1, together with the above Lemmas 3.3 and 3.4 essentially tell us that we can treat those as usual swaps with the only catch that removing self loops or moving wires past black nodes adds a -1 weight to the wires.

In the upcoming rewrite strategy, we will hence only specify the patterns without potential fermionic swaps inside. Should there be some present, it is understood that they will be moved out of the pattern, before the rewrite occurs. The rules necessary for the rewrite strategy are given in Figure 2.

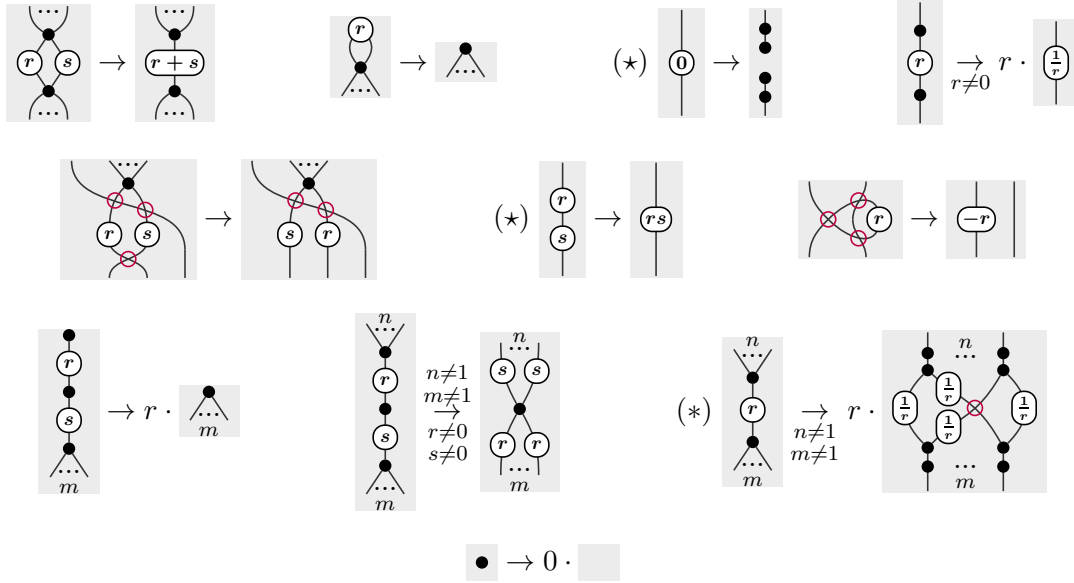


Figure 2: Rewrite rules. All these rules except (\star) are supposed to apply when any of the white nodes are replaced by identity (i.e. when their weight is 1). Rule $(*)$ can only be applied if at least one of the black nodes is internal, and if none of the other rules applies.

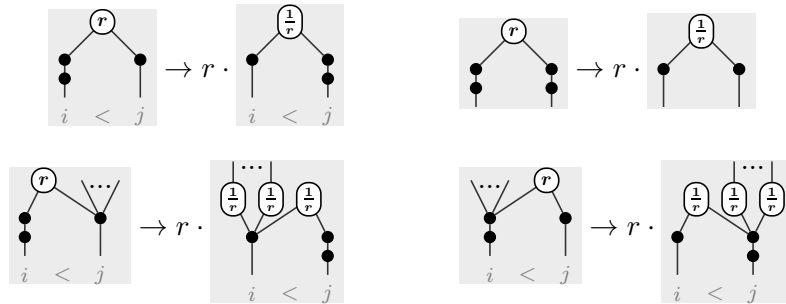


Figure 3: Rules for reduced WGS-X form, together with rule $(*)$ when the leftmost black node is a type-0 boundary node.

Proposition 3.6. *The rewrite rules of Figure 2 are derivable from the equational theory of Figure 1 and hence are sound.*

For the rewrite strategy to terminate, we need to distinguish between different types of nodes:




Definition 3.4 (Boundary Node / Internal Node). A node is a *boundary node of type 1* if it is linked directly to an output. A node is a *boundary node of type 0* if it is connected to a binary boundary node of type 1.



We say that a black node of D is *internal* if it is not a boundary node.

The rewrite strategy is then laid out as follows:

Definition 3.5 (Rewrite Strategy). The rewrite strategy is defined in 3 steps:

1. Apply the rewrites of Figure 2 in any order but following constraints, until none apply anymore. The diagram ends up in pseudo-WGS-X form.

2. First, whenever a type-1 boundary is linked to $n > 1$ outputs directly, apply  \rightarrow  to the $n - 1$ rightmost such outputs (the top black node then becomes a type-0 boundary node, the bottom one a type-1 boundary node). Then, push all potential fermionic swaps  between outputs inside the graph part. Finally, move boundary weights up



into the edges of the WGS using  $\rightarrow r \cdot$ . The diagram ends up in WGS-X form.

3. Whenever a type-0 vertex in the graph has a right neighbour, depending on the arity of the nodes, apply rule (*) or one of the rules of Fig. 3 between the two nodes (and apply any other possible rule before going on).

A claim was made in Definition 3.5 about the form of the diagram at the end of each step. Those claims are going to be proven in the following (Proposition 3.7). At the same time, we are going to show that the rewrite terminates.

Proposition 3.7 (Termination in rWGS-X form). *The rewrite strategy terminates in polynomial time. Moreover, after Step 1 of the rewrite, the diagram is indeed in pseudo-WGS-X form, after Step 2, it is in WGS-X form, and after Step 3, it is in rWGS-X form.*

An important operation on WGS-X states that has a simple graphical interpretation is the following:

Lemma 3.8. *For any diagram D in WGS-X form (s, G, \vec{b}) , applying  \circ  ^{$\otimes(b_i \oplus 1)$} on the i th output can be turned into the WGS-X form $(s, G \setminus \{i\}, \vec{b} \setminus b_i)$, where $G \setminus \{i\}$ is defined as the graph G from which vertex i is removed (together with all edges linked to i and their weights), and similarly $\vec{b} \setminus b_i$ is defined as the sequence \vec{b} from which i th element is removed.*


This allows us to prove the following:

Lemma 3.9. *For any diagram D in WGS-X form (s, G, \vec{b}) :*

$$\llbracket D \rrbracket = 0 \iff s = 0$$

We may then prove that 0-diagrams can be put in a very well-defined form:

Lemma 3.10. *Let D be a WGS-X state such that $\llbracket D \rrbracket = 0$. Then D can be put in the WGS-X form $(0, G = ([1, n], \emptyset, -), \vec{0})$, i.e.:*

$$\text{pW} \vdash D = 0 \cdot \text{$$

We are now able to prove the completeness of the equational theory.

Theorem 3.11. *Let D_1 and D_2 be two \mathbf{pW} -diagrams. Then:*

$$\llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket \iff \mathbf{pW} \vdash D_1 = D_2$$

This last theorem, together with the fact that the rewriting in $\mathbf{rWGS-X}$ form is polynomial (Proposition 3.7) makes the problem of deciding whether two \mathbf{pW} -diagrams are semantically equivalent a P problem.

4 Matchgates

This section aims at characterising exactly the linear maps that W -diagrams represent.

4.1 Matchgate Identities

Valiant first introduced matchgate identities to characterise $2 \rightarrow 2$ matchgates, a family of linear maps described in a combinatorial way [24]. In [5], the matchgate identities have been extended to characterise matchgates of any size. In the literature, there is a close link between matchgate identities and the Grassman-Plucker identities applied to Pfaffians. It is not the case here, as the diagrammatic technics allow us to directly link matchgate identities to matchings without the intermediate of the Pfaffian. We can fully recover the connection with Pfaffians through the Fetcher-Kasteleyn-Temperley algorithm for counting perfect matchings [1,22], more details on this are outlined in Section 5. Many of the proofs of this section are inspired by the very useful clarification of matchgate theory presented in [6]. Notice that contrary to the literature that differentiates between matchgrids, matchcircuits or matchnets, we will only use the term matchgate for any linear map satisfying the matchgate identities.

Recall that for binary words $\alpha \in \{0, 1\}^n$ and $\beta \in \{0, 1\}^m$, $\alpha \oplus \beta \in \{0, 1\}^n$ is the bitwise XOR (if $n = m$), $\alpha \cdot \beta \in \{0, 1\}^{n+m}$ the concatenation, $|\alpha| \in \{0, \dots, n\}$ the Hamming weight, *i.e.*, the number of ones in the word α , and $|\alpha|_2 \in \{0, 1\}$ the parity of this weight, 0 if even and 1 if odd.

Definition 4.1 (Matchgate Identities). A tensor $\Gamma \in \mathbb{C}^{2^n}$ satisfies the **matchgate identities** (MGIs) if for all $\alpha, \beta \in \{0, 1\}^n$:

$$\sum_{k=1}^{|\alpha \oplus \beta|} (-1)^k \Gamma_{\alpha \oplus e_{p_k}} \Gamma_{\beta \oplus e_{p_k}} = 0$$

Where $e_{p_k} \in \{0, 1\}^n$ is the binary word which is zero everywhere except in position p_k , which is the k th position in the set $\{p_1, \dots, p_{|\alpha \oplus \beta|}\} \subseteq \{1, \dots, n\}$ of positions in which the words α and β differs.

The matchgate identities are not linear, so the set of matchgates is not a subspace of the vector space \mathbb{C}^{2^n} but an algebraic variety [5]. In general, those identities are not algebraically independent, *i.e.* are not all strictly necessary to describe match-tensors.

Indeed, there are numerous symmetries in those identities. For example, the case $\alpha = \beta$ directly gives empty sums and exchanging α and β gives the same identity. Interestingly, one can replace half of the identities with a parity condition.

Proposition 4.1 (Parity condition [6]). *If Γ satisfies the matchgate identities then it satisfies the **parity condition**: for all $\alpha, \beta \in \{0, 1\}^n$, $|\alpha|_2 \neq |\beta|_2 \Rightarrow \Gamma_\alpha \Gamma_\beta = 0$.*

The parity condition splits match-tensors into two groups, the one with odd parity, such that $|\alpha|$ even implies $\Gamma_\alpha = 0$, and the one of even parity, such that $|\alpha|$ odd implies $\Gamma_\alpha = 0$. In particular, the parity condition directly implies that all terms in identities with $|\alpha|_2 \neq |\beta|_2$ are zero. Notice that the parity condition is not sufficient. We still need matchgate identities in general.

However, the parity condition is sufficient for $n \leq 3$, but not anymore for $n = 4$, the original case considered by Valiant [24]. In particular, for $n = 0$, the matchgate identities are trivially true; hence they are satisfied by all scalars (processes $0 \rightarrow 0$).

4.2 The Pro of Matchgates

We will now use the matchgates to define a pro. So far, matchgate identities have been used to characterise vectors seen as tensors, without consideration of inputs and outputs. To apply them to linear maps $f : n \rightarrow m$, we will use the state form: $[f] : 0 \rightarrow n + m$ described in Proposition 3.2. It allows us to define matchgates.

Definition 4.2 (Matchgates). A **matchgate** is a linear map $f : \mathbb{C}^{2^n} \rightarrow \mathbb{C}^{2^m}$ such that $[f]$ satisfies the matchgate identities.

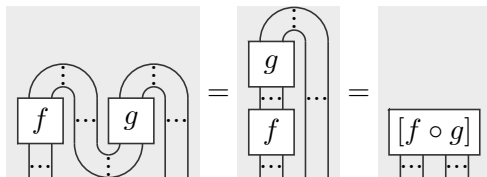
We would like to define a sub-pro of **Qubit** whose processes are matchgates, however, there are a few properties to check before that. We start by showing stability by the tensor product.

Lemma 4.2. *Given two linear maps $f : a \rightarrow b$ and $g : c \rightarrow d$ whose state forms $[f] \in \mathbb{C}^{2^{a+b}}$ and $[g] \in \mathbb{C}^{2^{c+d}}$ satisfy the matchgate identities, then $[f \otimes g] \in \mathbb{C}^{2^{a+c+b+d}}$ satisfies the matchgate identities.*

The next thing to check is stability by composition; this follows from the following result:

Lemma 4.3. *If $\Gamma \in \mathbb{C}^{2^{n+2}}$ satisfies the matchgate identities, then the tensor obtained by contracting two consecutive indices satisfies the matchgate identities.*

Notice that the consecutive indices assumption is essential here. Without it, we could easily construct the swap gate that does not satisfy the matchgate identities. To be able to contract consecutive indices is enough to show the stability by composition. The idea is to iterate contraction on consecutive indices until we obtain enough cups to use the snake equation, pictorially:



Now that we have stability by tensor and composition, it only remains to show the identities are matchgates. id_0 is a scalar, so directly a matchgate. The state-form of id_1 is the cap which is a matchgate as it satisfies the parity condition (sufficient for $n = 2$). The fact that all id_n are matchgates follows from stability by the tensor product. We can now state the main theorem of this subsection.

Theorem 4.4 (Match). *The matchgates form a pro **Match**, which is a sub-pro of **Qubit**.*

Notice that **Match** is compact closed since the cup and the cap are both matchgates. Hence we can also use process/state duality in **Match** without any worry. As expected, all W -diagrams represent matchgates.

Lemma 4.5. *The functor $\llbracket - \rrbracket : \mathbf{pW} \rightarrow \mathbf{Qubit}$ factorises through \mathbf{Match} , i.e., the interpretations of diagrams in W are matchgates.*

$$\begin{array}{ccc} \mathbf{pW} & \xrightarrow{\llbracket - \rrbracket} & \mathbf{Qubit} \\ & \dashrightarrow & \uparrow \\ & & \mathbf{Match} \end{array}$$

Proof. We have to prove that the interpretation of any \mathbf{pW} diagram is a matchgate. To do so, as matchgates are stable by composition and tensor product we only have to check that the interpretations of the generators are matchgates. The state forms of the generators have at most three outputs (n -ary spiders can be decomposed into binary and ternary spiders), so it is sufficient to check the parity condition, which is indeed satisfied by the interpretations of the generators. \square

4.3 Universality

Now that we proved that all \mathbf{pW} -diagrams represent matchgates, it remains to show that all matchgates can be represented by a \mathbf{pW} diagram, in other words, that \mathbf{pW} is universal for \mathbf{Match} . This will require a few additional properties of matchgates, adapting some results of [6].

Lemma 4.6. *If Γ satisfies the matchgate identities and $\Gamma_{\mathbf{0}} = 1$, where $\mathbf{0}$ is binary word full of 0, then it is uniquely determined by its coefficients Γ_{α} where $|\alpha| = 2$.*

Proof. If $|\alpha| = 0$ then we already know that $\Gamma_{\alpha} = 1$ and the parity condition implies that $\Gamma_{\alpha} = 0$ if $|\alpha| = 1$. We show that for all α with $3 \leq |\alpha|$, we can express Γ_{α} from coefficients Γ_{β} where all β s have strictly smaller Hamming weights. Let i be the first position where α and $\mathbf{0}$ differ, the matchgate identity corresponding to $\alpha \oplus e_i$ and $\mathbf{0} \oplus e_i$ is:

$$\sum_{k=1}^{|\alpha|} (-1)^k \Gamma_{\alpha \oplus e_i \oplus e_{p_k}} \Gamma_{e_i \oplus e_{p_k}} = 0$$

Here the p_k are exactly the position where α is 1, in particular $i = p_1$ so:

$$\Gamma_{\alpha} = \Gamma_{\alpha} \Gamma_{\mathbf{0}} = \sum_{k=2}^{|\alpha|} (-1)^k \Gamma_{\alpha \oplus e_i \oplus e_{p_k}} \Gamma_{e_i \oplus e_{p_k}}$$

For $k \geq 2$, We have $|e_i \oplus e_{p_k}| = 2$ and $|\alpha \oplus e_i \oplus e_{p_k}| = |\alpha| - 2$ so Γ_{α} is completely determined by coefficients corresponding to strictly smaller Hamming weight. It follows that all Γ_{α} can be expressed from the Γ_{β} s with $|\beta| = 2$. \square

We will now be able to reuse the normal form from Section 3 to construct diagrams representing any matchgate.

Lemma 4.7 (Universality). *\mathbf{pW} is universal for \mathbf{Match} .*

Proof. Relying on process/state duality, we only consider states $0 \rightarrow n$. Given Γ satisfying the matchgate identities, we will construct a W diagram D such that $\llbracket D \rrbracket = \Gamma$. We start by considering the case where $\Gamma_{\mathbf{0}} = 1$. Then we construct a weighted graph G on n vertices setting the weight of the edge (i, j) to $\Gamma_{e_i \oplus e_j}$. We then take D to be the diagram in graph form corresponding to G . By construction we then have $\llbracket D \rrbracket_{\mathbf{0}} = 1$ and $\llbracket D \rrbracket_{e_i \oplus e_j} = \Gamma_{e_i \oplus e_j}$ for all $i \neq j$. Furthermore, by Lemma 4.5, $\llbracket D \rrbracket$ is a matchgate so by Lemma 4.6, $\llbracket D \rrbracket = \Gamma$.

Now if $\Gamma_0 \neq 1$: First if $\Gamma_0 \neq 0$ then $\Gamma' = \frac{1}{\Gamma_0}\Gamma$ is of the right form so we can obtain D by adding a floating edge of weight Γ_0 to the diagram D' representing Γ' . The last case is $\Gamma_0 = 0$, then if $\Gamma = 0$ we can represent Γ by any diagram and a floating black node, else let β be such that $\Gamma_\beta \neq 0$, then Γ' defined as $\Gamma'_\alpha = \Gamma_{\alpha \oplus \beta}$ satisfies $\Gamma'_0 \neq 0$ and there is a diagram D' representing Γ' . A diagram D representing Γ is then obtained by plugging binary black nodes to the outputs of D' corresponding to the positions where β is 1. \square

Notice that since **Match** is a sub-pro of **Qubit**, the completeness proof of Section 3 still holds in **Match**. It provides us with a universal and complete graphical language for matchgates.

Theorem 4.8. *pW is universal and complete for Match.*

5 Further Work

The proper definition and axiomatisation of the pW-calculus pave the way to diverse investigations of the connection between combinatorics and quantum computing. We briefly outline in this last section some very promising directions that are the subjects of ongoing research.

5.1 New Simulation Techniques for Quantum Circuits

The identification of a fragment of the ZX-calculus exactly corresponding to the efficiently simulable Clifford gate [4] allows to design new rewrite-based simulation technics for quantum circuits introduced in [15]. Those algorithms have a parametrised complexity which is polynomial in the number of Clifford gated but exponential in the number of T -gates (a gate outside of the Clifford fragment sufficient to reach approximate universality).

Similarly, we have identified an efficiently simulable fragment of ZW-calculus: the pW-calculus exactly corresponding to matchgates. Adding the swap gate to pW we obtain another fragment of ZW which is exactly the fermionic ZW-calculus introduced in [12]. This calculus is universal for **Qubit** modulo an encoding trick: the dual-rail encoding. Equivalently, LFM is ZW where white nodes are contrived to have even arities, so adding arity one white nodes (corresponding to preparing $|+\rangle$ states) is enough to recover the full ZW-calculus, which is universal for **Qubits**. This situation suggests the possibility of designing rewrite-based simulation algorithms with complexities parametrised by the number of swap gates and/or $|+\rangle$ preparation. It would lead to a brand new kind of quantum simulation technics exploiting the combinatorial structure of matchgate and directly connected to classical perfect matching counting algorithms.

5.2 Combinatorial Interpretation of Full ZW-Calculus

In Section 2, we provided a combinatorial interpretation of pW-diagrams *via* perfect matchings in planar graphs. This combinatorial approach directly extends to LFM-calculus *via* perfect matchings in arbitrary graphs (which is #P-complete). Furthermore, we can also extend the interpretation to the full ZW-calculus, where white nodes can have arbitrary arities. To do so, we must consider hypergraph matchings, *i.e.*, subsets of hyperedges covering each vertex exactly once. The arbitrary arity white nodes here play the role of hyperedges, and the black nodes, the role of vertices. Thus, the interpretation of ZW-scalars is the number of hypergraph matchings of the hypergraph underlying the diagram. Notice that hypergraph matching is also presented as the set cover problem in the literature. The full ZW-calculus

could offer new perspectives on set cover in the same way that \mathbf{pW} does for perfect matchings. In particular, some reduction results appear to have very clear diagrammatical proofs.

Aside from perfect matchings, it seems that graphical languages can encode other counting problems on graphs or hypergraphs. Designing such languages could shed a new tensorial/diagrammatical light on the corresponding combinatorial problems. Those approaches are reminiscent of the recent ZH-based algorithm for $\#\text{SAT}$, introduced in [16] and related works linking graphical languages and counting complexity [10, 11]. Conversely, the question of applying similar combinatorial interpretations to other graphical languages as ZX-calculus [8], or ZH-calculus [2] is also worth being investigated.

5.3 Towards a Diagrammatic Approach of Perfect Matching Counting

In Section 2, we used the Fletcher-Kasteleyn-Temperley algorithm as a black box to compute the interpretation of \mathbf{pW} -scalars in polynomial time. However, it seems possible to achieve the same result with purely diagrammatical technics. In fact, applying the rewriting strategy described in Section 3 to a scalar reduces it to a normal form from which we can directly read the interpretation. It seems very probable that this requires only a polynomial number of rewrites.

This provides a way to count perfect matchings without referring to Pfaffian computation, and conversely, it gives a new algorithm to compute Pfaffians based on rewriting.

The FKT algorithm only applies to a specific class of graphs, called Pfaffian graphs, *i.e.*, the graphs admitting a Pfaffian orientation. In particular, all planar graphs are Pfaffian [14]. It seems that Pfaffian orientation are directly connected to the behavior of fermionic swap and their lack of naturality which introduces -1 weights in the edges. More generally, all graphs not containing $K_{3,3}$ are Pfaffian [17, 27] (we recall that planar graphs are precisely the graphs not containing neither $K_{3,3}$ nor K_5 as minors). Moreover, there also exists a polynomial time algorithm for K_5 -minor-free graphs [21] based on graph decomposition. There is a large amount of work in perspective, re-expressing in diagrammatic terms those different variations and understanding adequately how our rewriting rules could encode the minor constraints.

Formalising and implementing those different algorithms is the object of ongoing work. The main difficulty is to identify the suitable data structures to manipulate the topological data of a given diagram, equivalently, the specific planar embedding of the corresponding graph.

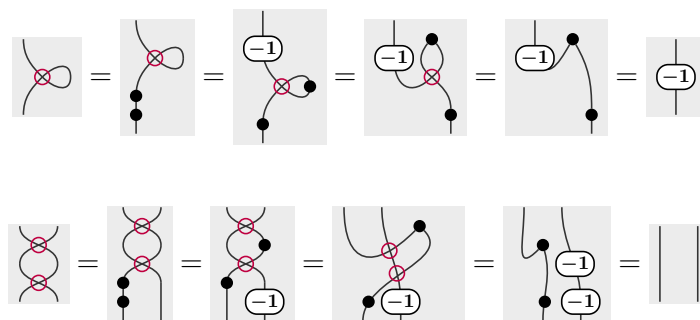
References

- [1] The statistics of dimers on a lattice: I. the number of dimer arrangements on a quadratic lattice. *Physica*, 27(12):1209–1225, 1961.
- [2] M Backens and A Kissinger. Zh: A complete graphical calculus for quantum computations involving classical non-linearity. *Electronic Proceedings in Theoretical Computer Science*, 287:23–42, 2019.
- [3] Miriam Backens. A new holant dichotomy inspired by quantum computation. *arXiv preprint arXiv:1702.00767*, 2017.
- [4] Miriam Backens, Simon Perdrix, and Quanlong Wang. Towards a minimal stabilizer zx-calculus. *Log. Methods Comput. Sci.*, 16(4), 2020. doi:10.23638/LMCS-16(4:19)2020.
- [5] Jin-yi Cai, Vinay Choudhary, and Pinyan Lu. On the theory of matchgate computations. *Theory Comput. Syst.*, 45(1):108–132, 2009. doi:10.1007/s00224-007-9092-8.

- [6] Jin-Yi Cai and Aaron Gorenstein. Matchgates revisited. *Theory Comput.*, 10:167–197, 2014. doi:10.4086/toc.2014.v010a007.
- [7] Titouan Carette. *Wielding the ZX-calculus, Flexsymmetry, Mixed States, and Scalable Notations. (Manier le ZX-calcul, flexsymétrie, systèmes ouverts et li-mandes)*. PhD thesis, University of Lorraine, Nancy, France, 2021. URL: <https://tel.archives-ouvertes.fr/tel-03468027>.
- [8] Bob Coecke and Ross Duncan. Interacting quantum observables. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfssdóttir, and Igor Walukiewicz, editors, *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, volume 5126 of *Lecture Notes in Computer Science*, pages 298–310. Springer, 2008. doi:10.1007/978-3-540-70583-3_25.
- [9] Bob Coecke and Aleks Kissinger. The compositional structure of multipartite quantum entanglement. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part II*, volume 6199 of *Lecture Notes in Computer Science*, pages 297–308. Springer, 2010. doi:10.1007/978-3-642-14162-1_25.
- [10] Niel de Beaudrap, Aleks Kissinger, and Konstantinos Meichanetzidis. Tensor network rewriting strategies for satisfiability and counting. In Benoît Valiron, Shane Mansfield, Pablo Arrighi, and Prakash Panangaden, editors, *Proceedings 17th International Conference on Quantum Physics and Logic, QPL 2020, Paris, France, June 2 - 6, 2020*, volume 340 of *EPTCS*, pages 46–59, 2020. doi:10.4204/EPTCS.340.3.
- [11] Niel de Beaudrap, Aleks Kissinger, and John van de Wetering. Circuit extraction for zx-diagrams can be #p-hard. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPICs*, pages 119:1–119:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.ICALP.2022.119.
- [12] Giovanni de Felice, Amar Hadzihasanovic, and Kang Feng Ng. A diagrammatic calculus of fermionic quantum circuits. *Log. Methods Comput. Sci.*, 15(3), 2019. doi:10.23638/LMCS-15(3:26)2019.
- [13] Amar Hadzihasanovic. A diagrammatic axiomatisation for qubit entanglement. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*, pages 573–584. IEEE Computer Society, 2015. doi:10.1109/LICS.2015.59.
- [14] P. W. Kasteleyn. Graph theory and crystal physics. *Graph Theory and Theoretical Physics*, 1967.
- [15] Aleks Kissinger, John van de Wetering, and Renaud Vilmart. Classical simulation of quantum circuits with partial and graphical stabiliser decompositions. In François Le Gall and Tomoyuki Morimae, editors, *17th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2022, July 11-15, 2022, Urbana Champaign, Illinois, USA*, volume 232 of *LIPICs*, pages 5:1–5:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.TQC.2022.5.

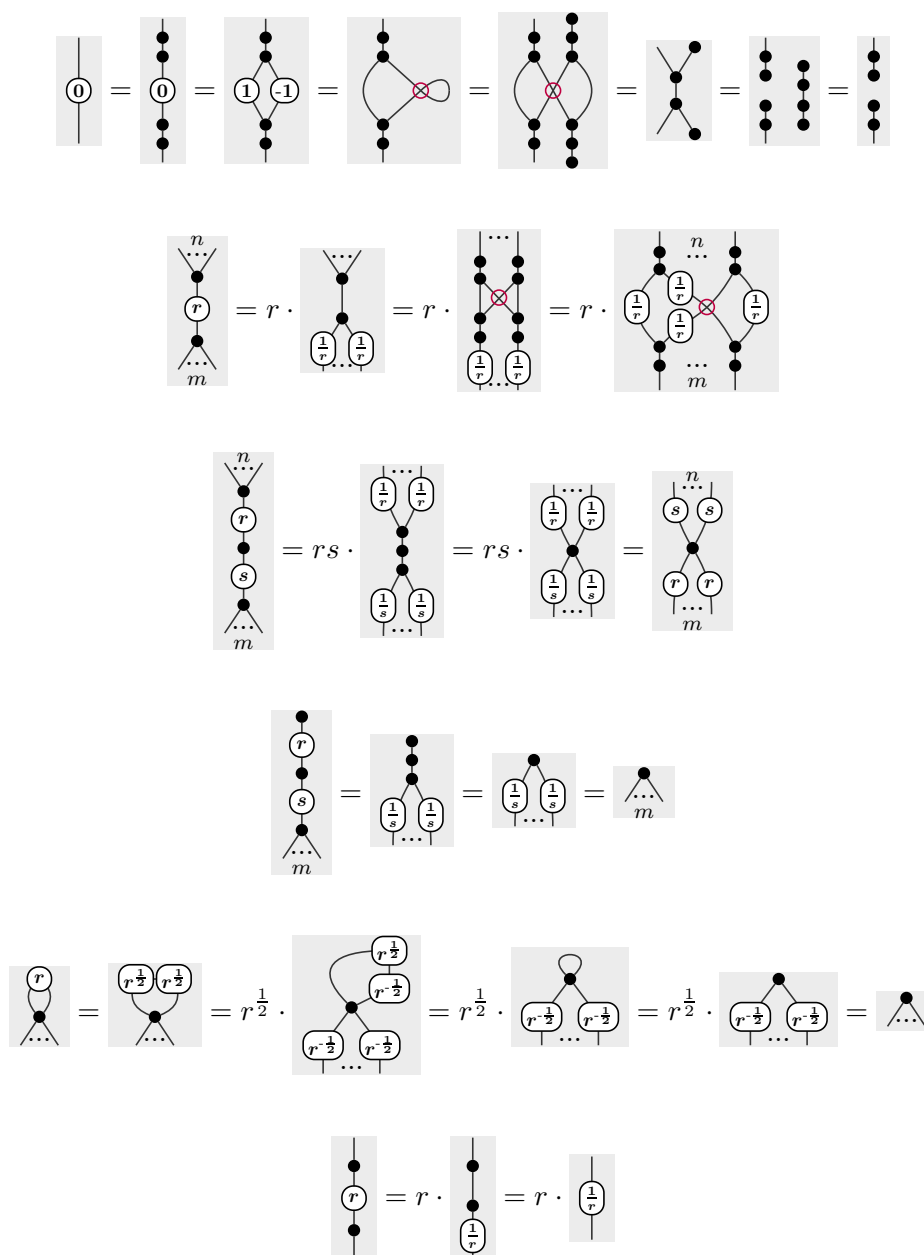
- [16] Tuomas Laakkonen, Konstantinos Meichanetzidis, and John van de Wetering. A graphical #sat algorithm for formulae with small clause density. *CoRR*, abs/2212.08048, 2022. [arXiv:2212.08048](https://arxiv.org/abs/2212.08048), doi:10.48550/arXiv.2212.08048.
- [17] Charles HC Little. An extension of kasteleyn’s method of enumerating the 1-factors of planar graphs. In *Combinatorial Mathematics: Proceedings of the Second Australian Conference*, pages 63–72. Springer, 1974.
- [18] Kang Feng Ng and Quanlong Wang. A universal completion of the zx-calculus. *arXiv preprint arXiv:1706.09877*, 2017.
- [19] Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.
- [20] Razin A Shaikh, Quanlong Wang, and Richie Yeung. How to sum and exponentiate hamiltonians in zxw calculus. *arXiv preprint arXiv:2212.04462*, 2022.
- [21] Simon Straub, Thomas Thierauf, and Fabian Wagner. Counting the number of perfect matchings in k5-free graphs. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 66–77. IEEE Computer Society, 2014. doi:10.1109/CCC.2014.15.
- [22] H. N. V. Temperley and Michael E. Fisher. Dimer problem in statistical mechanics-an exact result. *The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics*, 6(68):1061–1063, 1961. [arXiv:https://doi.org/10.1080/14786436108243366](https://doi.org/10.1080/14786436108243366), doi:10.1080/14786436108243366.
- [23] Leslie G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189–201, 1979. URL: <https://www.sciencedirect.com/science/article/pii/0304397579900446>, doi:https://doi.org/10.1016/0304-3975(79)90044-6.
- [24] Leslie G. Valiant. Quantum circuits that can be simulated classically in polynomial time. *SIAM J. Comput.*, 31(4):1229–1254, 2002. doi:10.1137/S0097539700377025.
- [25] Leslie G Valiant. Holographic algorithms. *SIAM Journal on Computing*, 37(5):1565–1594, 2008.
- [26] John van de Wetering. Zx-calculus for the working quantum computer scientist. *arXiv preprint arXiv:2012.13966*, 2020.
- [27] Vijay V. Vazirani. NC algorithms for computing the number of perfect matchings in k3, 3-free graphs and related problems. In Rolf G. Karlsson and Andrzej Lingas, editors, *SWAT 88, 1st Scandinavian Workshop on Algorithm Theory, Halmstad, Sweden, July 5-8, 1988, Proceedings*, volume 318 of *Lecture Notes in Computer Science*, pages 233–242. Springer, 1988. doi:10.1007/3-540-19487-8_27.
- [28] Quanlong Wang and Richie Yeung. Differentiating and integrating zx diagrams. *arXiv preprint arXiv:2201.13250*, 2022.

Proof of Lemmas 3.3 and 3.4.



□

Proof of Proposition 3.6.

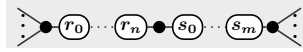


□

Proof of Proposition 3.7. We take each big step of the rewrite strategy, and show that each one terminates into the appropriate form.

1. We are going to define for every diagram a quantity, as a tuple, whose lexicographic order will be strictly reduced by any of the rewrite step. The very first quantity needed requires some focus.

We say that, whenever the following situation occurs (with $r_i \neq 0$ and $s_i \neq 0$), the two extremal black nodes are *fusion-equivalent* if they have degree ≥ 3 :



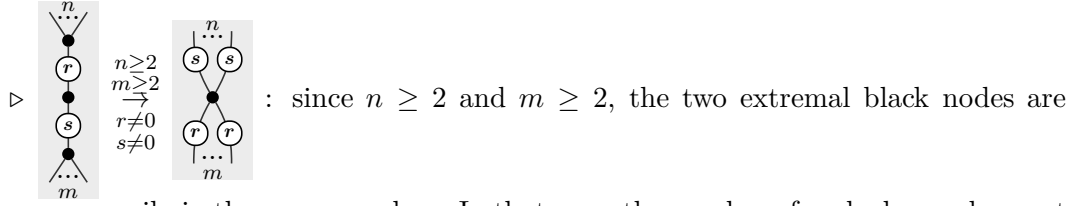
We define an equivalence relation \sim_f between black nodes by taking the reflexive, transitive closure of this relation. We say that an equivalence class is internal if *all* its constituents are internal nodes.

For a diagram D , we define $T(D) \in \mathbb{N}^6$ as the quantity:

$$T(D) := \left(\# \left(\begin{array}{c} \text{internal} \\ \sim_f \text{-classes} \end{array} \right), \# \left(\begin{array}{c} \bullet \\ \vdots \\ n \end{array} \right), \# \left(\begin{array}{c} \circ \\ \vdots \\ r \end{array} \right), \# \left(\begin{array}{c} \bullet \\ | \\ \bullet \end{array} \right), \# \left(\begin{array}{c} \circ \\ \diagup \ \diagdown \\ \bullet \end{array} \right), \# \left(\bullet \right) \right)$$

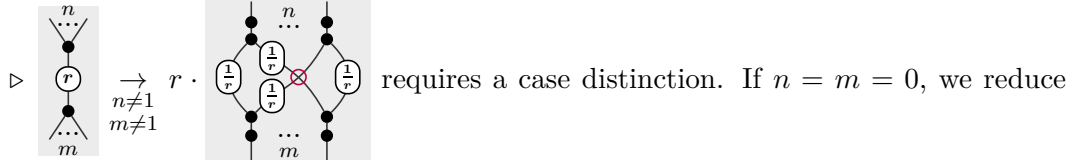
We can show that if $D \rightarrow D'$ then $T(D') < T(D)$ in the lexicographic order:

- ▷ may reduce the first two values (if one of the black nodes initially has degree 3), but it at least reduces the number of edges.
- ▷ similarly may reduce the first two values, but at least the third one (the number of edges).
- ▷ can only reduce the third value (and does not change the others).
- ▷ both reduce the fifth value without changing the other ones
- ▷ can only reduce the third value.
- ▷ only changes the number of binary black nodes (in particular, it cannot create new \sim_f -classes).
- ▷ may decrease the two first values, but necessarily decreases the third one.



necessarily in the same \sim_f -class. In that case, the number of such classes does not change, but the second value is decreased. Notice that, when applied repeatedly, this rule eventually reduces every class to a single element.

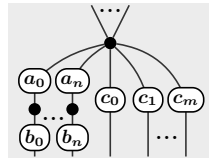
▷ reduces the last value.



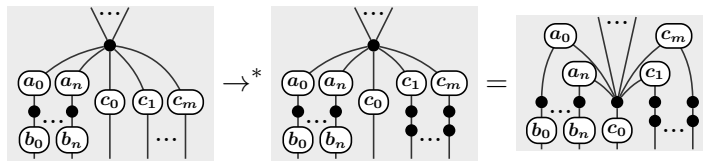
the number of edges. If $n = 0$ and $m \geq 2$ if the bottom black node is internal, we decrease the first value, if not, the second value (similarly, the case $n \geq 2$ and $m = 0$ decreases T). In the case where $n \geq 2$ and $m \geq 2$, let us focus on one of the two black nodes. Since the rule can only be applied when none other can, all its neighbours each constitute their own \sim_f -class. After the rewrite, all the top non-binary black nodes will join the \sim_f -class of the neighbour they connect to, so the number of *internal* \sim_f -classes is either unchanged if the node was not internal, or reduced by one if it was. Since at least one of the two nodes had to be internal for the rewrite to apply, the overall number of \sim_f -classes reduces.

At the end of these rewrites, the diagram does not have any internal node left (all were removed by Rule (*) or fused into boundary nodes). The only possible form of a diagram with no internal nodes is the pseudo-WGS-X form. Notice that in between applications of (*), which decreases the number of internal \sim_f -classes, there can only be a linear (in the size of the diagram) number of rewrites that can be applied. When (*) is applied, there is no more than $O(n^2)$ generators (n being the number of black nodes). This forces the step to stop after a polynomial amount of time.

2. Consider a boundary node that is not in the proper form for the diagram to be in WGS-X form. In all generality, the node's neighbourhood is in the form (up to rearranging of the output wires):

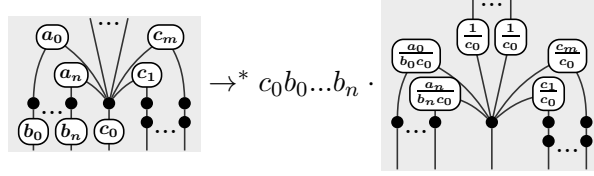


We then apply two black nodes on the m rightmost outputs:



Doing this for all “improper” nodes allows us to associate exactly one type-1 boundary node to each output. At this point, we can move all up into the graph part of the diagram. This changes the order of the vertices, and potentially adds -1 weights

on some edges, potentially on output wires. The weights on output wires are precisely handled by the last substep:

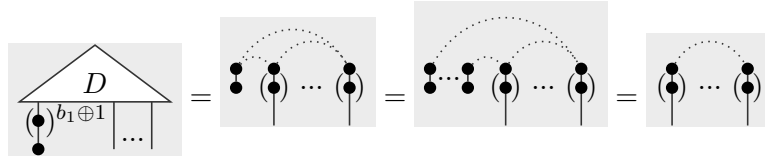


When all weights are moved up, the diagram ends up in WGS-X form. This step obviously stops after a polynomial amount of time.

3. The rule (*) together with that of Figure 3 cover all situations when a type-0 boundary node has a right neighbour (notice that boundary nodes of arity 1 cannot have neighbours). In the process of this step, we are rewriting the WGS-X state in an equivalent one, with fewer type-0 nodes with a right neighbour. The step hence eventually terminates, and in such a situation that none of the type-0 boundary nodes have a right neighbour. This is exactly the form of a reduced WGS-X state. For the same reason as in step 1, this step terminates in polynomial time.

Finally, you may notice that the rewrite strategy terminates in polynomial time, as each of the three steps is polynomial. \square

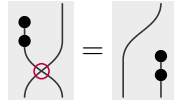
Proof of Lemma 3.8. W.l.o.g. we can assume $i = 1$. We then get:



where dotted lines represent potential edges (potentially with weights), that cross with fermionic swaps \bowtie ; and where we used the following steps:



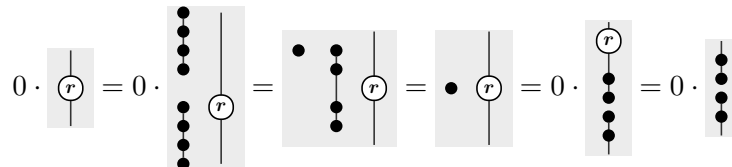
together with the fact that fermionic swaps do not interfere here as:



\square

Proof of Lemma 3.9. The right to left implication is obvious. Now suppose $s \neq 0$. Then, applying $\bullet^{\otimes n} \circ \bullet^{\otimes (\vec{b}_1 \oplus 1 \dots 1)}$ to the outputs reduces to the WGS-X state $(s, G_\emptyset, ())$ thanks to Lemma 3.8 (where $G_\emptyset = ([], \emptyset, -)$ is the empty graph), whose interpretation is scalar s . Since $s \neq 0$, $\llbracket D \rrbracket \neq 0$. \square

Proof of Lemma 3.10. We can start by removing all edges from the WGS-X state thanks to:



and then fusing black nodes together when possible (again, the fermionic swaps do not cause issues here). When done, we end up with a tensor of $\begin{array}{c} \bullet \\ | \end{array}$ s and $\begin{array}{c} \bullet \\ \bullet \\ | \end{array}$ s. It remains to show the following:

$$0 \cdot \begin{array}{c} \bullet \\ | \end{array} = \begin{array}{c} \bullet \\ \bullet \\ | \end{array} = \begin{array}{c} \bullet \\ \bullet \\ \bullet \\ | \end{array} = 0 \cdot \begin{array}{c} \bullet \\ \bullet \\ \bullet \\ \bullet \\ | \end{array} = 0 \cdot \begin{array}{c} \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \\ | \end{array} = 0 \times 0 \cdot \begin{array}{c} \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \\ | \end{array} = 0 \cdot \begin{array}{c} \bullet \\ \bullet \\ | \end{array}$$

□

Proof of Theorem 3.11. The right to left implication is soundness of the rules, which is obvious as the equational theory is a subpart of the equational theory for ZW-diagrams, which is known to be sound [13]. Let us now assume that $\llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket$. Using Proposition 3.7 and Proposition 3.6, we can turn both D_1 and D_2 into respectively d_1 and d_2 , in rWGS-X form, and in a way that preserves semantics, i.e. $\llbracket d_1 \rrbracket = \llbracket d_2 \rrbracket$. Let us denote $(s_i, G_i = ([1, n], E_i, w_i), \vec{b}_i)$ the rWGS-X form of d_i .

Notice that if $\llbracket d_1 \rrbracket = \llbracket d_2 \rrbracket = 0$, Lemma 3.10 ensures that d_1 and d_2 can both be put in the same form, which proves the result in that case. In the following, we can hence consider that the diagrams have non-0 interpretation.

First, let us show that $\vec{b}_1 = \vec{b}_2$. Let us consider the first index i for which $b_{1i} \neq b_{2i}$. As both d_1 and d_2 are in reduced form, for all $j < i$, $b_{1j} = b_{2j} = 1$. Suppose w.l.o.g. that $b_{1i} = 0$ and $b_{2i} = 1$. Again, since they are in reduced form, the i th vertex in G_1 can only have neighbours on its left. Let us apply $\begin{array}{c} \bullet \\ | \end{array}$ to the first i qubits in d_1 and d_2 . On the one hand:

$$\begin{array}{c} \triangle \\ d_1 \\ \bullet \dots \bullet \dots \bullet \\ \vdots \\ i \end{array} = s_1 \cdot \begin{array}{c} \dots \\ \bullet \dots \bullet \dots \bullet \dots \bullet \\ \vdots \\ i \end{array} = s_1 \cdot \begin{array}{c} \bullet \dots \bullet \dots \bullet \\ \vdots \\ i \end{array} = 0$$

where we used Lemma 3.8 for the first $i - 1$ qubits and the black node fusion for the i th. The map hence obtained from d_1 is the zero map. On the other hand, using Lemma 3.8 again:

$$\begin{array}{c} \triangle \\ d_2 \\ \bullet \dots \bullet \dots \bullet \\ \vdots \\ i \end{array} = s_2 \cdot \begin{array}{c} \dots \\ \bullet \dots \bullet \dots \bullet \dots \bullet \\ \vdots \\ i \end{array} = s_2 \cdot \begin{array}{c} \bullet \dots \bullet \dots \bullet \\ \vdots \\ i \end{array}$$

which thanks to Lemma 3.9 is necessarily not zero. There is a contradiction, hence, if d_1 and d_2 are in rWGS-X form with $\llbracket d_1 \rrbracket = \llbracket d_2 \rrbracket$, then $\vec{b}_1 = \vec{b}_2$.

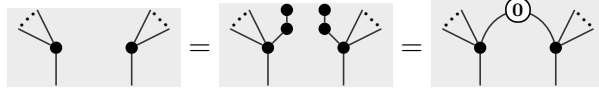
We can then show that we can recover $s_1 = s_2$. Indeed, if we apply $\begin{array}{c} \bullet \\ | \end{array}^{\otimes n} \circ \begin{array}{c} \bullet \\ | \end{array}^{\vec{b}_1 \oplus 1 \dots 1}$ on both diagrams, we get (Lemma 3.8):

$$\begin{array}{c} \triangle \\ d_1 \\ \begin{array}{c} \bullet \\ \vdots \\ b_{11} \end{array} \dots \begin{array}{c} \bullet \\ \vdots \\ b_{1n} \end{array} \\ \vdots \end{array} = s_1 \cdot \begin{array}{c} \dots \\ \bullet \dots \bullet \dots \bullet \dots \bullet \\ \vdots \end{array} = s_1 \cdot \begin{array}{c} \dots \\ \bullet \dots \bullet \dots \bullet \dots \bullet \\ \vdots \end{array}$$

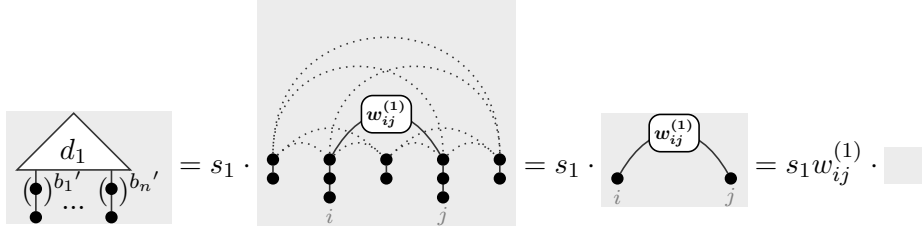
Similarly on d_2 , we get $s_2 \cdot \begin{array}{c} \dots \\ \bullet \dots \bullet \dots \bullet \dots \bullet \\ \vdots \end{array}$, which proves $s_1 = s_2$.

Finally, we can show that the weight between every pair (i, j) of vertices in G_1 and G_2 is the same, with the convention that having no edge between two vertices is equivalent to

having an edge with weight 0:

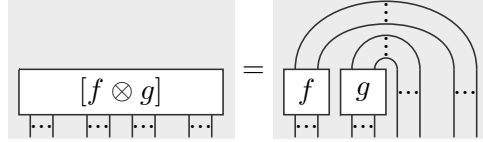


To that end, consider $\vec{b}' := \vec{b}_1 \oplus 1..101\dots101\dots1$. We may now apply $\bullet^{\otimes n} \circ \bullet^{\vec{b}'}$ on both diagrams, to get on the one hand:



and similarly on the other hand, we get $s_1 w_{ij}^{(2)}$ from d_2 . This implies $w_{ij}^{(1)} = w_{ij}^{(2)}$. Doing so for all pairs of vertices in G_i gives us $G_1 = G_2$. \square

Proof of Lemma 4.2. Given $\alpha \in \{0, 1\}^{a+c+d+b}$ we write $\alpha = \alpha^1 \cdot \alpha^2 \cdot \alpha^3$ with $\alpha^1 \in \{0, 1\}^a$, $\alpha^2 \in \{0, 1\}^{c+d}$ and $\alpha^3 \in \{0, 1\}^b$. We have $[f \otimes g]_\alpha = [f]_{\alpha^1 \cdot \alpha^3} [g]_{\alpha^2}$. Pictorially:



We compute the matchgate identity corresponding to $\alpha, \beta \in \{0, 1\}^{a+c+d+b}$.

$$\sum_{k=1}^{|\alpha \oplus \beta|} (-1)^k [f \otimes g]_{\alpha \oplus e_{p_k}} [f \otimes g]_{\beta \oplus e_{p_k}}$$

We start by splitting the sum into three terms, one for a , one for $c+d$ and one for b . By doing so, we also re-index the p_k for each sum:

$$\begin{aligned} & [g]_{\alpha^2} [g]_{\beta^2} \left(\sum_{k=1}^{|\alpha^1 \oplus \beta^1|} (-1)^k [f]_{(\alpha^1 \oplus e_{p_k}) \cdot \alpha^3} [f]_{(\beta^1 \oplus e_{p_k}) \cdot \beta^3} \right) \\ & + (-1)^{|\alpha^1 \oplus \beta^1|} [f]_{\alpha^1 \cdot \alpha^3} [f]_{\beta^1 \cdot \beta^3} \left(\sum_{k=1}^{|\alpha^2 \oplus \beta^2|} (-1)^k [g]_{\alpha^2 \oplus e_{p_k}} [g]_{\beta^2 \oplus e_{p_k}} \right) \\ & + (-1)^{|\alpha^1 \oplus \beta^1| + |\alpha^2 \oplus \beta^2|} [g]_{\alpha^2} [g]_{\beta^2} \left(\sum_{k=1}^{|\alpha^3 \oplus \beta^3|} (-1)^k [f]_{\alpha^1 \cdot (\alpha^3 \oplus e_{p_k})} [f]_{\beta^1 \cdot (\beta^3 \oplus e_{p_k})} \right) \end{aligned}$$

The second term is zero as we recognise the matchgate identity satisfied by $[g]$ for α^2 and β^2 . Furthermore if $|\alpha^2 \oplus \beta^2|_2 \neq 0$ the parity condition on $[g]$ implies that $[g]_{\alpha^2} [g]_{\beta^2} = 0$ canceling the first and third terms. In the case where $|\alpha^2 \oplus \beta^2|_2 = 0$, we can gather the first and third terms to get:

$$[g]_{\alpha^2} [g]_{\beta^2} \left(\sum_{k=1}^{|(\alpha^1 \cdot \alpha^3) \oplus (\beta^1 \cdot \beta^3)|} (-1)^k [f]_{(\alpha^1 \cdot \alpha^3) \oplus e_{p_k}} [f]_{(\beta^1 \cdot \beta^3) \oplus e_{p_k}} \right)$$

We now recognise the matchgate identity satisfied by $[f]$ for $\alpha^1 \cdot \alpha^3$ and $\beta^1 \cdot \beta^3$, which evaluates to zero. So $[f \otimes g]$ satisfies the matchgate identities. \square

Proof of Lemma 4.3. Let $i, i+1 \in \{1, \dots, n+2\}$ be two consecutive indices. We write $\alpha[xy] \in \{0, 1\}^{n+2}$ for the binary word $\alpha \in \{0, 1\}^n$ in which $x, y \in \{0, 1\}$ have been respectively inserted in position i and $i+1$. Denoting Γ' the tensor obtained by contracting the indices i and $i+1$ in Γ we have: $\Gamma'_\alpha = \Gamma_{\alpha[00]} + \Gamma_{\alpha[11]}$.

We now compute a matchgate identity:

$$\begin{aligned} & \sum_{k=1}^{|\alpha \oplus \beta|} (-1)^k \Gamma'_{\alpha \oplus e_{p_k}} \Gamma'_{\beta \oplus e_{p_k}} = \\ & \sum_{k=1}^{|\alpha \oplus \beta|} (-1)^k \left(\Gamma_{(\alpha \oplus e_{p_k})[00]} + \Gamma_{(\alpha \oplus e_{p_k})[11]} \right) \left(\Gamma_{(\beta \oplus e_{p_k})[00]} + \Gamma_{(\beta \oplus e_{p_k})[11]} \right) \end{aligned}$$

Distributing and using $(\alpha \oplus \beta)[xy] = \alpha[xy] \oplus \beta[00]$ gives us four terms:

$$\begin{aligned} & \sum_{k=1}^{|\alpha \oplus \beta|} (-1)^k \Gamma_{\alpha[00] \oplus e_{p_k}[00]} \Gamma_{\beta[00] \oplus e_{p_k}[00]} \\ & + \sum_{k=1}^{|\alpha \oplus \beta|} (-1)^k \Gamma_{\alpha[00] \oplus e_{p_k}[00]} \Gamma_{\beta[11] \oplus e_{p_k}[00]} \\ & + \sum_{k=1}^{|\alpha \oplus \beta|} (-1)^k \Gamma_{\alpha[11] \oplus e_{p_k}[00]} \Gamma_{\beta[00] \oplus e_{p_k}[00]} \\ & + \sum_{k=1}^{|\alpha \oplus \beta|} (-1)^k \Gamma_{\alpha[11] \oplus e_{p_k}[00]} \Gamma_{\beta[11] \oplus e_{p_k}[00]} \end{aligned}$$

The first and last terms correspond to matchgate identities respectively for $\alpha[00]$ and $\beta[00]$, and for $\alpha[11]$ and $\beta[11]$, so they are zero as Γ is required to satisfy the matchgate identities. We would like to say the same about the second and third terms. Sadly some terms are missing to get complete matchgate identities since we have added new positions where the words differ. The missing terms are $(-1)^\ell \Gamma_{\alpha[10]} \Gamma_{\beta[01]}$ and $(-1)^{\ell+1} \Gamma_{\alpha[01]} \Gamma_{\beta[10]}$ for the second line, and $(-1)^\ell \Gamma_{\alpha[01]} \Gamma_{\beta[10]}$ and $(-1)^{\ell+1} \Gamma_{\alpha[10]} \Gamma_{\beta[01]}$ for the third line, where ℓ is the index of the position i in the set of differing positions. We see that the four missing terms altogether cancel each other. So we can safely add the missing terms in the sum and get two complete matchgate identities, respectively, for $\alpha[00]$ and $\beta[11]$, and for $\alpha[11]$ and $\beta[00]$. Finally, the global sum is zero, and Γ' satisfies the matchgate identities. \square