ZX-Flow: A Flexible Criterion for Deterministic Computation with ZX-Diagrams

Aleks Kissinger
University of Oxford
aleks.kissinger@ox.ac.uk

John van de Wetering
University of Amsterdam, QuSoft
john@vdwetering.name

Flow criteria are used to efficiently extract computations, either in the form of measurement patterns or quantum circuits, from ZX-diagrams. Existing criteria such as causal flow, generalised flow, and Pauli flow, were all originally formulated for graph states, so they require ZX-diagrams to be in a very particular graph-state-like form. This form is easily broken by applying basic ZX rules and makes establishing some desirable properties very complicated. Here, we introduce a new "ZX-native" flow criterion called ZX-flow, formulated using a new type of decoration of a ZX-diagram we call Pauli semiwebs. These are a generalisation of Pauli webs, which have recently been used extensively in reasoning about fault-tolerant computations in the ZX-calculus. We show that ZX-flow is straightforwardly preserved by all Clifford rewrites and furthermore that a ZX-diagram has ZX-flow if and only if it is Clifford-equivalent to a graph-like ZX-diagram with Pauli flow. Finally, we show that any diagram with ZX-flow can be readily interpreted either as a deterministic measurement-based computation or as a Clifford isometry followed by a sequence of Pauli exponentials. The latter can then be efficiently extracted to a quantum circuit.

1 Introduction

The ZX-calculus [8] is a graphical language for reasoning about quantum processes that has many applications in quantum computation. It allows one to represent a computation as a graph-like structure called a ZX-diagram and rewrite it according to a complete set of graphical rules [1, 29] to transform and simplify quantum computations. It has been used for instance to optimise and verify quantum circuits [9, 28, 11, 15, 27, 13], analyse parametrised quantum circuits [21, 31, 26, 25, 30], develop new constructions in measurement-based quantum computing [2, 14, 10, 20, 3], and design fault-tolerant quantum computations [32, 22, 23, 12, 4].

These methods benefit from the ZX-calculus because of its increased flexibility with respect to the standard quantum circuit notation, which allows representations of non-unitary operations including projections, ancilla preparations, measurements and post-selections. This additional flexibility does however come with a cost: when you produce a ZX-diagram representing a computation, it is usually not immediately clear how to translate this diagram into a representation that allows direct execution on quantum hardware. We have to first *extract* a circuit from the diagram. This circuit extraction problem is very hard in the most general case [5], but there are several known conditions on diagrams that allow efficient extraction of a circuit. These so-called *flow* conditions were borrowed from the field of measurement-based quantum computation (MBQC), where they were originally used to associate single-qubit measurements to certain "correction sets", which identify future measurements that can be adapted to obtain a deterministic result. The most well-known such conditions are called generalised flow and Pauli flow [7].

It turns out that these conditions also suffice to turn a ZX-diagram into a quantum circuit expressed in a fixed basis of 1- and 2-qubit gates [9, 2, 24]. When optimising quantum circuits with the ZX-calculus, it is useful to preserve flow conditions to ensure that we can extract a circuit from the final, reduced

ZX-diagram. However, even some of the simplest ZX rules, like spider fusion, break flow conditions, making it awkward to transform ZX-diagrams in a flow-preserving way.

Previously, this has required one to express all ZX rewrites in terms of a handful of rules like local complementation and pivoting, which are then (arduously) proven to preserve generalised flow or Pauli flow [9, 2, 24, 18, 3]. There are two reasons this has been so complicated. The first is that flow conditions were originally designed for graph states, not ZX-diagrams, so it requires us to maintain ZX-diagrams in a very special, "graph-state-like" form for the conditions to even be formulated. The second reason is that these conditions require one to explicitly track the time ordering for the classically tractable, Clifford parts of a ZX-diagram, even though this ordering is to a large extent arbitrary. Pauli flow handles this to some extent, but it still imposes some conditions on Clifford nodes which block many ZX rewrites.

In this paper we introduce a new flow condition specifically designed to work on ZX-diagrams that we call *ZX-flow*. It is built around the concept of *Pauli semiwebs*, a new non-Clifford generalisation of Pauli webs. Pauli webs are a way of colouring the wires of a ZX-diagram to track the propagation of Pauli operators through it. They have been a useful tool in using ZX to reason about fault-tolerant computations, as they give an elegant way to formalise logical operators, stabilisers, and detecting sets of measurements associated with an error-correcting procedure [6, 23]. However, Pauli webs must satisfy strong local consistency conditions which break at non-Clifford locations, i.e. nodes where the phase parameter is not an integer multiple of $\frac{\pi}{2}$. Pauli semiwebs, on the other hand, allow certain limited violations of these conditions at certain locations, which we call *defects*, that allow them to handle non-Clifford structure. This new structure enables us to give a compact, easily-visualised flow criterion. We say a diagram has ZX-flow if its non-Clifford nodes can be given a time ordering, and each node can be associated to a Pauli semiweb which only has defects at the node itself and non-Clifford nodes in its future. We can then relate this to existing flow criteria with our first main result, which states that a diagram has ZX-flow if and only if it is Clifford-equivalent to a graph-like diagram with Pauli flow.

This structure gives us (at least) two ways to directly interpret a ZX-diagram as a quantum computation. The first is using MBQC, where we can interpret non-Clifford nodes as measurements of a stabiliser resource state and defects as locations where we feed-forward measurement outcomes. For our second main result, we show that a diagram with ZX-flow can be interpreted as a unitary quantum circuit, where the Pauli semiwebs enable us to read off a particular circuit decomposition just using the time-ordering and the support of semiwebs on the outputs. The latter is similar in spirit to the circuit extraction for Pauli flow due to Simmons [24], but it is much more direct and the proof of correctness follows straightforwardly from the properties of Pauli semiwebs.

In addition to the notion of ZX-flow and our two main theorems, a third contribution of this paper is the basic theory of Pauli semiwebs, which can be seen as generalising Pauli webs with a notion of "feed-forward". This unifies notions from fault-tolerance and MBQC, and may have broader applications, e.g. in the design of fault-tolerant computations involving non-Clifford components.

2 Preliminaries

2.1 ZX-diagrams

ZX-diagrams are string diagrams built from the following basic components:

$$\llbracket \; \vdots \; \alpha \; \vdots \; \rrbracket := |0\rangle^{\otimes m}\langle 0|^{\otimes n} + e^{i\alpha}|1\rangle^{\otimes m}\langle 1|^{\otimes n} \qquad \llbracket \; \vdots \; \alpha \; \vdots \; \rrbracket := |+\rangle^{\otimes m}\langle +|^{\otimes n} + e^{i\alpha}|-\rangle^{\otimes m}\langle -|^{\otimes n}$$

Z-spider $\qquad\qquad$ X-spider

$$[\![\ \underset{H\text{-gate}}{\text{—}\blacksquare\text{—}}\ ]\!] := |+\rangle\langle 0| + |-\rangle\langle 1| \qquad [\![\ \underset{\text{id}}{\text{——}}\ ]\!] := \sum_i |i\rangle\langle i| \qquad [\![\ \underset{\text{swap}}{\times}\ ]\!] := \sum_{ij} |ij\rangle\langle ji|$$

using parallel and sequential composition. Two ZX-diagrams are considered equivalent if one can be deformed into the other without changing the order of inputs/outputs.

It will be convenient to represent ZX-diagrams combinatorically. We treat a diagram D as a collection of nodes N_D and wires W_D, which are related by a binary adjacency relation $a_D \subseteq N_D \times W_D$. Each node v has a *type* $t_D(v) \in \{Z, X, H\}$ and a *phase* $\phi_D(v) \in \mathbb{R}$. Wires can be adjacent to zero, one, or two nodes. We define totally ordered, not necessarily disjoint subsets $I_D, O_D \subseteq W_D$ of *inputs* and *outputs*. We also allow diagrams to include a scalar factor $\lambda_D \in \mathbb{C}$ and impose a meta-rule that there is a unique zero diagram. That is, for all diagrams D, D' with the same number of inputs and outputs, $\lambda_D = \lambda_{D'} = 0 \implies D = D'$.

If $t_D(v) = H$, it represents an H-gate, must have phase 0, and be adjacent to precisely two wires. If $t_D(v) \in \{Z, X\}$, it is a *spider*. Notably, the combinatoric structure of D is enough to define the linear map $[\![D]\!]$ unambiguously. That is, the linear map represented by a ZX-diagram is fully specified by its nodes and connecting wires and isomorphic diagrams will define the same linear map. This principle is often called *only connectivity matters* (see e.g. [16]). Note we drop the subscript D from parts of the diagram and the semantic brackets $[\![.]\!]$ if they are clear from context.

It is sometimes useful to treat pairs of wires that share an H-gate as a single logical unit, especially when discussing the adjacency of spiders in a diagram.

Definition 2.1. A pair of wires $\{w_1, w_2\}$ that are both adjacent to the same H-gate are called an *H-edge*. Any single wire $\{w\}$ that is not part of an H-edge is called a *plain edge*. We consider two spiders to be adjacent if they are connected via a plain edge or an H-edge. Let $E_D \subseteq 2^W$ be the set of all plain edges and H-edges in D.

The linear map of a ZX-diagram D is also preserved by transformation according to a set of rewrite rules called the ZX-calculus. While there exist several variations, for our purposes we will focus on the *extended Clifford ZX-calculus*, as shown in Figure 2 of Section 4.1. We say a spider is *Clifford* if its phase is a multiple of $\frac{\pi}{2}$, and is *non-Clifford* otherwise. Diagrams consisting just of H-gates and Clifford spiders can be efficiently reduced to normal form [16] using the rules of the ZX-calculus. In particular, this implies efficient equality checking for such diagrams and a graphical version of the Gottesman-Knill theorem.

2.2 Pauli webs

A Pauli web is an annotation on a ZX-diagram that denotes how Pauli operators propagate through the diagram.

Definition 2.2. Let D be a ZX-diagram and let $v \in N_D$ be a node. The associated *local state* $|v\rangle$ is defined as: $|0\rangle^{\otimes k} + e^{i\phi(v)}|1\rangle^{\otimes k}$ for Z-spiders, $|+\rangle^{\otimes k} + e^{i\phi(v)}|-\rangle^{\otimes k}$ for X-spiders, and $|0+\rangle + |1-\rangle$ for H-gates.

Let $\mathscr{P}_n$ denote the Pauli group on n qubits. For a Pauli operator $\vec{P} \in \mathscr{P}_{|W|}$, let $\vec{P}_j$ and $\vec{P}_J$ be the Pauli operator restricted to a particular wire $j \in W$ and set of wires $J \subseteq W$, respectively. For a node v, we let $\vec{P}_v := \vec{P}_{a(v)}$ be $\vec{P}$ restricted to the neighborhood of v. As these restrictions are technically only defined up to a global phase, we fix the phase such that $\vec{P}_j$ and $\vec{P}_v$ are (tensor products of) the standard Pauli operators $\{I, X, Y, Z\}$.

Definition 2.3. A *Pauli web* for a diagram D is a Pauli operator $\vec{w} \in \mathscr{P}_{|W|}$ such that $|v\rangle$ is an eigenstate for $\vec{w}_v$ for all $v \in N$. We say a Pauli web has *X-support* at a wire j if $\vec{w}_j$ is either X or Y and has *Z-support* if it is either Y or Z.

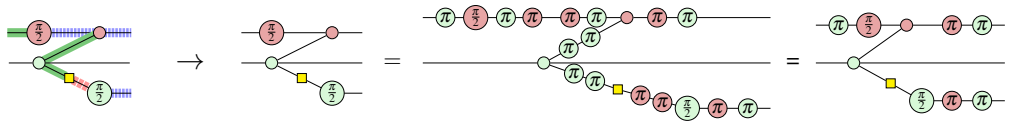
We distinguish the four different ways a wire can be labelled by a Pauli web as follows:

Pauli webs track the propagation of Pauli operators through ZX-diagrams. In particular, they characterise the Pauli operators that leave a Clifford map invariant.

Theorem 2.4. Let $\vec{w}$ be a Pauli web for a ZX-diagram D, then we have $D = (-1)^k \vec{w}_O D \vec{w}_I$, where $\vec{w}_I$ and $\vec{w}_O$ are the restrictions of $\vec{w}$ to the inputs and outputs of D, respectively.

Proof. We can show this by "firing" the Pauli web $\vec{w}$ (see Example 2.5 below). That is, we can treat the overall linear map of D as a tensor product of local states $|v\rangle$, where we use "caps" $\langle 00| + \langle 11|$ to form internal wires and inputs. We can then replace each local state $|v\rangle$ with $\lambda \vec{w}_v |v\rangle$. On interior wires, Pauli operators cancel out (possibly up to a sign, since $Y^T = -Y$), leaving just $\vec{w}_I$ on the inputs and $\vec{w}_O$ on the outputs. The sign σ is computed as the product of all of the local eigenvalues λ, multiplied by -1 for each Y on an interior or input wire of $\vec{w}$. □

Example 2.5 (Firing a Pauli web)**.** Consider the following Pauli web on a diagram D. To "fire" this Pauli web, we add for each spider the corresponding Paulis around it determined by the colouring on that wire:



On each internal wire, each Pauli now appears twice so it cancels out. The only Paulis are $\vec{P}$ on inputs and $\vec{Q}$ on outputs, hence we obtain $D \propto \vec{Q} D \vec{P}$.

Definition 2.3 is about eigenvalues, which is not always the easiest way to think of Pauli webs. It is sometimes useful to unpack the Pauli web criteria into more combinatoric conditions. We will say support of the *same type* to mean Z-support for Z-spiders and X-support for X-spiders, and we will say support for the *opposite type* to mean X-support for Z-spiders and Z-support for X-spiders. It is straightforward to check that following 4 conditions characterise the Pauli operators for which a node $|v\rangle$ is an eigenstate.

Theorem 2.6. A Pauli operator $\vec{w} \in \mathscr{P}_{|W|}$ is a Pauli web if and only if it satisfies the following criteria:

1. **H:** every H-gate has a neighbourhood coloured from the set $\{II, XZ, ZX, YY\}$;
2. **all-or-nothing:** the neighbourhood of every spider must either have empty or full support of the opposite type (meaning either all the wires are coloured in the opposing colour, or none of them are);
3. **parity:** the neighbourhood of all spiders must have even support of the same type, unless their phase is an odd multiple of $\frac{\pi}{2}$, in which case it has even support of the same type iff it has no support of the opposite type; and
4. **Clifford:** if the neighbourhood of a spider has support of the opposite type, it must be Clifford.

It is clear from Definition 2.3 that the product of Pauli webs is again a Pauli web. For Clifford ZX-diagrams, it is often therefore useful to find a minimal generating set of webs. The combinatoric criteria from Theorem 2.6 can be expressed as $\mathbb{F}_2$-linear equations over $2|W|$ boolean variables, indicating the Z-support and X-support at each wire in W. Hence, a generating set of Pauli webs can be computed efficiently by computing a generating set of solutions to a system of linear equations.

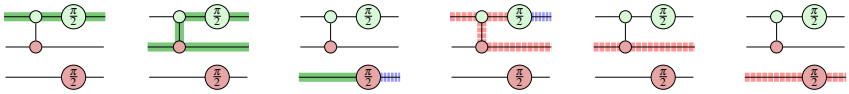
We can divide Pauli webs into four categories: (i) *detectors*, which have no support on inputs or outputs, (ii) *stabilisers*, which only have support on outputs, (iii) *costabilisers*, which only have support on inputs, and (iv) *logicals*, which have support on both inputs and outputs. All four types play a role in quantum error correction and fault-tolerant quantum computing (see e.g. [23, 6]). For our purposes, we will be most interested in logical Pauli webs, which for isometries, take a special form.

Theorem 2.7. Any Clifford diagram D is an isometry, up to a scalar, if and only if its logical Pauli webs are generated by Pauli webs $\{\ell_Z(i)\}_{i\in I}$ and $\{\ell_X(i)\}_{i\in I}$, where $\ell_Z(i)$ (resp. $\ell_X(i)$) has Z-support (resp. X-support) on input i and no other support on the inputs of D.

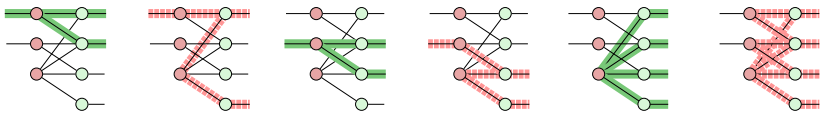
Proof. By Theorem 2.4, the existence of the Pauli web $\ell_Z(i)$ implies that $DZ_i = \vec{\mathscr{Z}}_i D$ for some Pauli $\vec{\mathscr{Z}}_i$, and similarly $\ell_X(i)$ implies $DX_i = \vec{\mathscr{X}}_i D$ for some $\vec{\mathscr{X}}_i$. The Paulis $\vec{\mathscr{Z}}_i$, $\vec{\mathscr{X}}_i$ then give a complete stabiliser tableau for D, which implies that it is an isometry (these Paulis $\vec{\mathscr{Z}}_i$, $\vec{\mathscr{X}}_i$ must be independent in $\mathscr{P}_n$ since otherwise $\ell_Z(i)$ and $\ell_X(i)$ wouldn't be independent generators of the logical Pauli webs). For the converse direction we find the logical operators of the isometry, and construct the corresponding Pauli webs by observing how we should push the Paulis from input to outputs. □

If D is an isometry with n inputs and n outputs, then it is a unitary. Hence, it is completely fixed by its $2n$ logical Pauli webs. If it has $k < n$ inputs, then it is fixed by $2k$ logical webs plus $(n+k)-2k = n-k$ additional stabilising Pauli webs. These correspond precisely to the stabilisers and logical operators of a quantum error correcting code.

Example 2.8 (Pauli webs for a unitary)**.** The following 3 qubit unitary has six independent logical Pauli webs, which correspond to its stabiliser tableau:



Example 2.9 (Pauli webs for an isometry)**.** The following isometry is the embedding map for the $[[4,2,2]]$ error detecting code. It has 4 logical Pauli webs and 2 stabiliser Pauli webs, corresponding to the logical operators and stabilisers of that code:



2.3 Pauli Flow

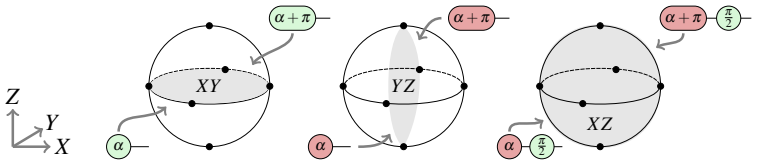
The standard flow conditions (causal flow, generalised flow, Pauli flow) were originally defined for graph states, but they also make sense for *graph-like* ZX-diagrams [9, 2].

Definition 2.10. A ZX-diagram is called *graph-like* if:

1. every spider is of type Z and internal wires only connect Z-spiders to H-gates;
2. no H-gate is connected to the same spider via both adjacent wires (no self-loops) and no two H-gates are connected to the same pair of spiders (no parallel edges); and
3. every input and output is connected to a spider via a plain edge, and every spider is connected to at most 1 input and at most 1 output.

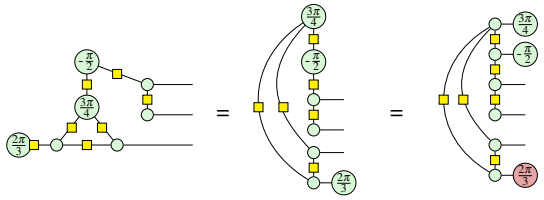
Graph-like diagrams are those ZX-diagrams D whose structure is uniquely fixed by an *open graph* G_D. An open graph is a simple, undirected graph with chosen (not necessarily disjoint) subsets of input and output vertices. G_D is the open graph whose vertices are Z-spiders and whose edges are connections between spiders via an H-gate. Input vertices are the set of spiders adjacent to an input wire and output vertices are the set of spiders adjacent to an output wire. We will represent the H-edges using dashed blue lines following [9].

An open graph, along with an ordering of its vertices and a choice of measurement angles, defines part of a measurement-based quantum computation (MBQC) in the one-way model [7]. In MBQC, one prepares a type of stabiliser state called a *graph state* and performs single-qubit measurements, which can be either Pauli measurements or measurements within one of the three planes XY, YZ, or XZ of the Bloch sphere.



(1)

Graph-like ZX-diagrams can be interpreted as graph states, where each of the non-output vertices is being measured in a basis determined by its phase, e.g.



(2)

While individual measurement outcomes are non-deterministic, one can recover deterministic computation via a mechanism called *feed forward*, where later measurement choices are adapted based on intermediate measurement outcomes. We will only very roughly sketch the one-way model and its relationship to the ZX-calculus here, referring the interested reader to e.g. [2] or [16, Chapter 9] for a more in-depth treatment.

A generic way to define a correction strategy is to define a *correction set* $c(v)$ for each non-output vertex v. Let $\bar{I} := V_G \backslash I$ and $\overline{O} := V_G \backslash O$ be the *non-inputs* and *non-outputs*, respectively. Let $c : \overline{O} \to 2^{\bar{I}}$ be defined as a function from non-output vertices to correction sets.

Similar to a Pauli web, a correction set can be "fired" on a graph state to introduce a stabiliser for that graph state. The stabilisers of graph states are all of the form $S_v := X_v \Pi_{w \in \mathcal{N}(v)} Z_w$. That is, they introduce a Pauli X on a given vertex and a Pauli Z on all neighbours of that vertex. We can "fire" a correction set $c(v)$ by introducing the stabiliser $\Pi_{u \in c(v)} S_u$. This will introduce an X on all the vertices in the set $c(v)$ and introduce a Z on all the vertices in the *odd neighbourhood* $Odd(c(v))$, i.e. the set of all vertices connected to $c(v)$ oddly many times.

Now, we want to arrange our $c(v)$ such that "firing" $c(v)$ will flip (i.e. correct) the measurement outcome at v while only disturbing other measurements in the future of v. How we choose $c(v)$ depends on what kind of measurement we are doing at v, which is dictated by a measurement label $\mu(v) \in \{X, Y, Z, XY, YZ, XZ\}$. For convenience, we will treat these six measurement labels as 1- and 2-element sets, writing e.g. $X \in \mu(v)$ to mean $\mu(v) = X, XY$, or XZ.

A choice of correction sets accounting for all six of these types of measurements is called *Pauli flow*. Normally, this is stated as nine distinct conditions. However, we can treat all of the measurement choices

symmetrically if we define the following three *anti-commutation sets*:

$$A_X(u) = Odd(c(u)) \qquad A_Y(u) = c(u)\Delta Odd(c(u)) \qquad A_Z(u) = c(u)$$

i.e. locations which get a Pauli that anti-commutes with the given Pauli when the stabiliser $\Pi_{u\in c(v)}S_u$ is introduced.

Definition 2.11. An open graph (G,I,O) with a choice of measurements μ has *Pauli flow* if there exists a pair $(\preceq, c)$ such that:

(i) $P \in \mu(u) \implies u \in A_P(u)$

(ii) $P \in \mu(v), v \in A_P(u) \implies u \preceq v$

This is not the standard way Pauli flow is presented in the literature, but it will make the proofs in the following sections much simpler. A proof of equivalence to the standard definition from [7] is given in Appendix A.

Although Pauli flow is defined for open graphs, we can also interpret it as a condition on graph-like ZX-diagrams. While in general there can be different choices of measurement labels compatible with a ZX-diagram, we will assume measurement labels are chosen as follows: (i) spiders whose phase is a multiple of π have $\mu(v) = X$, (ii) spiders whose phase is an odd multiple of $\pi/2$ have $\mu(v) = Y$, and (iii) all other (necessarily non-Clifford) spiders have $\mu(v) = XY$.

As explained in [17, Section 4.2], spiders with measurement labels in the YZ and XZ planes can always be replaced with pairs of spiders, where one is measured in X and Y and the other in XY (e.g. we interpret the $\frac{\pi}{2}$ phases in (1) for XZ as additional spiders in the diagram). For this reason, we will only consider ZX-diagrams with measurement labels in the set X, Y, and XY and we choose as many spiders to be Pauli-measured as possible, based on their angle.

3 Pauli Semiwebs

In this section, we introduce a new generalisation of Pauli webs, which allow the web conditions to be relaxed at certain locations, which we call *defects*. This generalisation will allow us to define many Pauli semiwebs on non-Clifford ZX-diagrams.

For a spider $v \in N$, we define its *twisted local state* $|v_\alpha\rangle$ as $|0\rangle^{\otimes k} + e^{i(\phi(v)+\alpha)}|1\rangle^{\otimes k}$ for Z-spiders, $|+\rangle^{\otimes k} + e^{i(\phi(v)+\alpha)}|-\rangle^{\otimes k}$ for X-spiders, and $|v\rangle$ for H-gates. Hence, $|v_0\rangle = |v\rangle$ and twisting has no effect on H-gates. This is just a notational convenience to avoid a case distinction in the following definition.

Definition 3.1. A *Pauli semiweb* for a diagram D is a Pauli operator $\vec{w} \in \mathscr{P}_{|W|}$ such that for all nodes v there exists an α such that:

$$\vec{w}_v|v\rangle = \lambda|v_\alpha\rangle \tag{3}$$

If $\alpha \neq 0$, then v is called an α-*defect*, or simply a *defect*, of the Pauli semiweb.

By definition, a Pauli semiweb with no defects is a Pauli web. If a spider has one or more legs, then α is uniquely fixed by (3). Like Pauli webs, we can characterise Pauli semiwebs in terms of the local Z- and X-support around nodes. In fact, the semiweb conditions are a strict subset of the Pauli web conditions from Theorem 2.6.

Theorem 3.2. A Pauli operator $\vec{w} \in \mathscr{P}_{|W|}$ is a Pauli semiweb iff it satisfies the **H** and **all-or-nothing** conditions from Theorem 2.6.

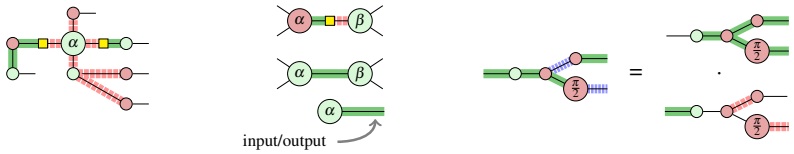


Figure 1: Left: Example of a basic semiweb. Note that all the spiders on the boundary of the web have a π-defect. Middle: different possibilities for an edge semiweb. Right: decomposing a Pauli semiweb into a basic semiweb (top) and edge semiwebs (bottom).

Proof. The behaviour of H-gates is the same for Pauli semiwebs, so it suffices to show that the **all-or-nothing** condition is equivalent to (3) for all spiders v. Let v be a Z-spider. Suppose we write $\vec{w}_v|v\rangle = \lambda \vec{P}_X \vec{P}_Z |v\rangle$, where $\vec{P}_Z$ contains only Z operators and $\vec{P}_X$ contains only X operators. Then, the RHS equals $\lambda \vec{P}_X |v_\alpha\rangle$ for $\alpha = 0$ if the number of Z operators is even and $\alpha = \pi$ if it is odd. Now, (3) will be satisfied iff $\vec{P}_X|v_\alpha\rangle$ is proportional to a Z-spider. This happens if and only if $\vec{P}_X$ either preserves the basis states $|0...0\rangle$ and $|1...1\rangle$ or it swaps them, which is true iff $\vec{P}_X$ is either I or $X \otimes \ldots \otimes X$. Hence, (3) is satisfied for Z-spiders iff their neighbourhood satisfies the all-or-nothing condition. We can argue similarly for X-spiders. □

As a consequence of this theorem, defects are introduced precisely when the **Clifford** and **parity** conditions from Theorem 2.6 are violated. If the **Clifford** condition is violated, we get a -2α defect, if the **parity** condition is violated, we get a π defect, and if both are violated, we get a $\pi - 2\alpha$ defect. The following result is an immediate consequence of the fact that products of Pauli operators preserve the **H** and **all-or-nothing** criteria.

Corollary 3.3. The product of Pauli semiwebs is a Pauli semiweb.

Because of this, we can also characterise Pauli semiwebs by identifying a generating set of semiwebs.

Definition 3.4. For a spider v in D, the *basic semiweb* $\vec{b}_v$ of v is the smallest semiweb that has support of the opposite colour on a wire adjacent to v. That is, it has X-support if v is a Z-spider and it has Z-support if v is an X-spider.

For a graph-like ZX-diagram, the basic semiweb $\vec{b}_v$ at a spider v is the Pauli semiweb with an X operator on all of the wires adjacent to v and a Z operator on all wires that connect via an H-gate to wires adjacent to v. In other words, it colours all of the H-edges, inputs, and outputs adjacent to v. In general, basic semiwebs will colour fusable components of the diagram. We say a pair of spiders is *fusable* if they are either the same colour and connected via a plain edge or opposite colours and connected by an H-edge. A fusable component is a connected sub-diagram of fusable spiders. For a spider v, the basic semiweb $\vec{b}_v$ colours the edges adjacent to every spider in the fusable component of v. See Figure 1 for an example.

Basic semiwebs nearly generate all of the Pauli semiwebs. However, there is one case that they don't cover: if a plain edge or H-edge connects a fusable pair of spiders or it contains an input/output wire, we can always form a semiweb consisting just of that edge, with defects at each adjacent spider.

Definition 3.5. An *edge semiweb* is a Pauli semiweb that colours a single plain edge X or Z or a single H-edge XZ.

Theorem 3.6. Any Pauli semiweb can be written as $\vec{w} = \vec{e} \prod_{v \in B} \vec{b}_v$ for $\vec{e}$ a product of edge semiwebs. In this case, we call B a *generating set of spiders* for the Pauli semiweb $\vec{w}$.

Proof. Let $\vec{w}$ be a Pauli semiweb on D. It suffices to show that $\vec{w}$ can be reduced to the identity by multiplying by basic semiwebs and edge semiwebs. Suppose a spider v is adjacent to a wire coloured with the opposite type. Then, by minimality of $\vec{b}_v$, the Z-support and X-support of $\vec{w}$ must contain the support of b_v. Hence, multiplying $\vec{w}$ by b_v strictly reduces the support of $\vec{w}$. We can repeat this until no spider is adjacent to a wire of the opposite colour. The only remaining support will be on plain edges or H-edges whose ends are at inputs/outputs or at spiders of the same type. This can be represented as a product of edge webs. □

Note that when D is graph-like, the set of generating spiders of a semiweb is unique, so we call it *the* generating set of spiders for $\vec{w}$. See Figure 1 for an example of decomposing a Pauli web into basic and edge semiwebs.

4 ZX-flow

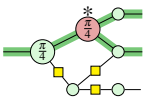
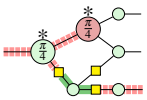
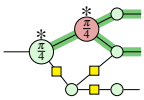
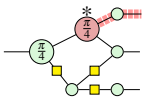
Now that we have established some basic concepts for Pauli semiwebs, we are ready to state our main definition.

Definition 4.1. We say a ZX-diagram D has *ZX-flow* if there exists a partial ordering $\preceq \subseteq T \times T$ on the non-Clifford spiders $T \subseteq N_D$ and we have the following Pauli semiwebs:

- for every input wire $i \in I$, a pair of Pauli semiwebs $\ell_Z(i)$ and $\ell_X(i)$ that only have defects at non-Clifford spiders and whose restrictions to input wires are Z_i and X_i, respectively; and
- for every non-Clifford spider $v \in T$, a Pauli semiweb $f(v)$ that has a π-defect at v and all other defects at non-Clifford spiders v' where $v \preceq v'$.

We call $\ell_X(i)$ and $\ell_Z(i)$ the *logical semiwebs* of an input i, and we call $f(v)$ the *flow semiweb* of a non-Clifford spider v.

Example 4.2. The following ZX-diagram has the following logical Z- and X-semiwebs for input i_1 and flow semiwebs for the two non-Clifford spiders v_1, v_2:

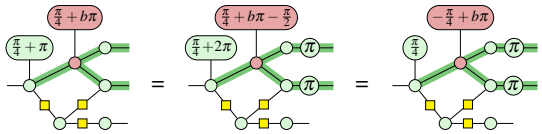
$\ell_Z(i_1) :=$  $\ell_X(i_1) :=$  $f(v_1) :=$  $f(v_2) :=$ 

Here we have marked the spiders with a defect with $*$. This defines a ZX-flow with $v_1 \preceq v_2$.

One way we can interpret a ZX-flow for a diagram D is that it gives us a recipe for implementing D using a Clifford isometry D' (thought of as a fixed resource, generalising e.g. graph states in the one-way model) and adaptive, single-qubit, non-Clifford measurements. A spider v with non-Clifford phase α represents a single-qubit measurement in the basis $\{-(\alpha+k\pi)\}_{k=0,1}$. If $k=0$, we got the desired outcome, so we proceed to the next measurement. If $k=1$, we have an undesired π phase, which we can cancel with another π-defect by "firing" the flow semiweb $f(v)$, as we did for Pauli webs in Example 2.5. However, unlike firing Pauli webs, firing semiwebs can change the angles of some spiders. Notably, it will send $\alpha+\pi$ to α, as desired, but it may also change some future measurement angles, located at the defects of $f(v)$.

Example 4.3. Taking the diagram with ZX-flow from Example 4.2, the flow semiweb $f(v_1)$ has a π defect at v_1 and a $-\frac{\pi}{2}$ defect at v_2. This means if we get an unwanted outcome at v_1, we can push it into

the future by firing $f(v_1)$:



Note that the defect at the $\frac{\pi}{4}$ X-spider results in flipping the $\frac{\pi}{4}$ to $-\frac{\pi}{4}$, meaning that its measurement angle depends on the outcome of measuring the previous $\frac{\pi}{4}$ spider.

Whereas ZX-flow assigns a flow semiweb to each non-Clifford spider, we can define a stricter notion that requires a flow semiweb for every spider.

Definition 4.4. A *strong ZX-flow* for a diagram D consists of a partial ordering $\preceq \subseteq S \times S$ on all spiders $S \subseteq N_D$ and the following Pauli semiwebs:

- for every input wire $i \in I$, a pair of *logical semiwebs* $\ell_Z(i)$ and $\ell_X(i)$ whose restrictions to input wires are Z_i and X_i, respectively; and
- for every spider $v \in S$, a *flow semiweb* $f(v)$ that has a π-defect at v and all other defects at spiders v' where $v \preceq v'$.

As we will see in Section 4.2, this stronger notion of ZX-flow is equivalent to Pauli flow in a strict sense. Namely, graph-like ZX-diagrams have strong ZX-flow if and only if they have Pauli flow. However, this stronger notion is not preserved by arbitrary Clifford rewrite rules, hence it is more easily broken by the kinds of simplifications we would like to do on ZX-diagrams.

It is not immediately clear that the existence of a strong ZX-flow implies the existence of a ZX-flow, because the former allows Pauli semiwebs to have defects at arbitrary spiders, not just the non-Clifford spiders. To solve this issue, we will use the technique of *focusing*. In generalised flow and Pauli flow, focusing the flow corresponds to delaying measurement corrections in MBQC as long as possible. For ZX-flow, we see this concept reflected in trying to make semiwebs satisfy the **parity** condition from Theorem 2.6 for as many spiders v as possible. For a non-Clifford Z-spider (resp. X-spider), this means v should have even Z-support (resp. X-support) in its neighborhood, but it might still be a defect. For Clifford spiders, this means all the Pauli web criteria are satisfied, so there will not be a defect at v.

The following definition makes sense both for ZX-flow and strong ZX-flow, so we will state it as a single definition and corresponding focusing result.

Definition 4.5. A *focused (strong) ZX-flow* is a (strong) ZX-flow whose logical semiwebs satisfy the **parity** condition at every spider and whose flow semiwebs $f(v)$ satisfy the **parity** condition at every spider $v' \neq v$.

Theorem 4.6. ZX-diagrams admit (strong) ZX-flow if and only if they admit *focused* (strong) ZX-flow.

Proof of Theorem 4.6. One direction is trivial. For the other direction, we prove this constructively by efficiently transforming a (strong) ZX-flow $(\ell_Z, \ell_X, f, \preceq)$ into a focused (strong) ZX-flow. The technique is the same in both cases, where the only difference is whether we operate on just the non-Clifford spiders in case of normal ZX-flow and all spiders for strong ZX-flow.

We pick the $\preceq$-maximal spider $v \in T$ (or S in the case of strong ZX-flow) such that $f(v)$ violates the **parity** condition for at least one spider $v' \neq v$. Let $v_1, \ldots, v_k$ be the spiders where the **parity** condition is violated. Since these must all be greater than v with respect to $\preceq$, it must be the case that $f(v_j)$ can only violate the **parity** condition at v_j (because otherwise v would not be maximal in having this property). Note that $f(v_j)$ has a π-defect at v_j and that this implies the **parity** condition is violated at

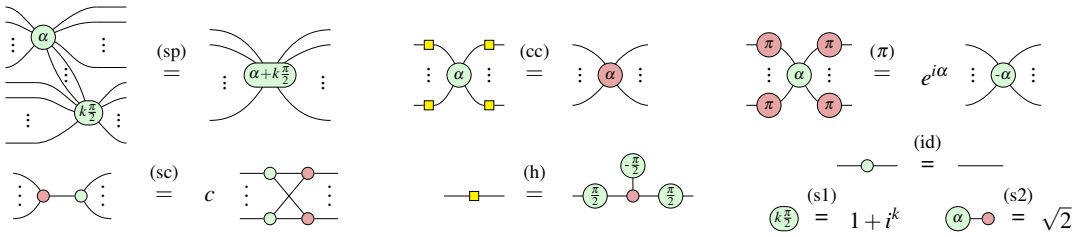


Figure 2: The extended Clifford ZX-calculus, where $\alpha \in \mathbb{R}$, $k \in \mathbb{Z}$, and $c = \sqrt{2}^{(n-1)(m-1)}$ for n inputs and m outputs. The rules are pronounced: (sp) spider, (cc) colour-change, (π) π stabiliser, (sc) strong complementarity, (h) Hadamard, (id) identity, (s1) and (s2) scalar rules.

v_j. The product of two semiwebs that violate the **parity** condition at a spider v' will no longer violate it. Hence, the product $\vec{w} = f(v)f(v_1)\dots f(v_k)$ will violate the **parity** condition at v and nowhere else. We therefore set $f(v) := \vec{w}$ and then repeat until none of the flow semiwebs violate the focusing condition. Once this is done, we can remove any **parity** violations from the logical semiwebs $\ell_Z(i), \ell_X(i)$ similarly, by multiplying with the flow semiwebs of the spiders where there are **parity** violations. □

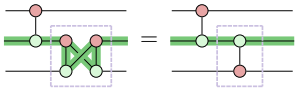
Being focused implies that none of the logical semiwebs have defects at Clifford nodes and the flow semiwebs $f(v)$ only have a defect at a Clifford node if v is itself Clifford. Hence, a strong focused ZX-flow on all spiders will restrict to a valid ZX-flow on the non-Clifford spiders. The following is then an immediate consequence of Theorem 4.6.

Corollary 4.7. If a ZX-diagram admits a strong ZX-flow, it admits a ZX-flow.

4.1 Clifford preservation of ZX-flow

In this section, we will show that all of the rules of the Clifford ZX-calculus preserve ZX-flow. In fact, we can show this fact for a bit more than the usual Clifford ZX-calculus, as we can allow rules that have arbitrary phase parameters in certain locations; see Figure 2. In particular, spider fusion with arbitrary angles preserves ZX-flow in either direction with only one exception: it cannot be applied to unfuse a spider with a Clifford angle into two spiders with non-Clifford angles.

The key observation is that when we replace a diagram consisting of only Clifford spiders with another one, we can always locally update Pauli semiwebs thanks to Theorem 2.4. For example:



We formalise the notion of local preservation of Pauli webs by purely Clifford rules in the following Lemma.

Lemma 4.8. For two Clifford ZX-diagrams D, D', if $[\![D]\!] = [\![D']\!]$, then for any Pauli web $\vec{w}$ on D we can find a corresponding Pauli web $\vec{w}'$ on D' such that $\vec{w}$ and $\vec{w}'$ have the same support on the boundary.

It then only remains to show that any non-Clifford spiders that we modify can be assigned valid flow semiwebs after a rule application. This gives rise to the following result.

Theorem 4.9. The rules of the extended Clifford ZX-calculus (Figure 2) preserve ZX-flow.

Proof. All the semiwebs used in the ZX-flow only have defects at non-Clifford spiders, so that in the neighbourhood of the Clifford spiders, the semiwebs satisfy all the local Pauli web rules. Preservation of

ZX-flow for ZX-rules involving only Clifford spiders then follows from Lemma 4.8. It then remains to consider (sp), (cc), (π), and (s2) for $\alpha \neq j\frac{\pi}{2}$.

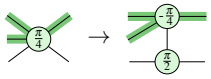
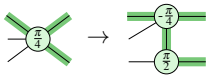
For (sp), let v be the non-Clifford spider on the LHS and v' be the non-Clifford spider on the RHS. If we apply (sp) left-to-right, we define $\preceq$ on the resulting diagram by replacing v with v'. We then need to show v' has a suitable flow semiweb, that the other semiwebs in the diagram remain valid, and that the defects respect the new ordering $\preceq$.

The wires on the RHS are a subset of the wires on the LHS, so we can restrict all logical and flow webs $\vec{w}$ to the wires of the new diagram to obtain a new semiweb $\vec{w}^-$. All $\vec{w}^-$ will satisfy the **H** condition, and if any of the input/output wires on the LHS of (sp) have X-support for any semiweb, then they all must have X-support, so all $\vec{w}^-$ also satisfy the **all-or-nothing** condition, so that all the previously existing semiwebs remain valid.

Now, let $f(v') := f(v)^-$. We know that $f(v)$ had a π-defect at v, so it must be the case that oddly many wires adjacent to v have Z-support and no adjacent wires have X-support. But then the Clifford spider on the LHS must have Z-support on an even number of adjacent wires. We then distinguish two cases. Either an odd number of internal wires have Z-support, or an even number do. If this is odd, then the non-Clifford spider has an even number of boundary wires with Z-support, and the Clifford spider has an odd amount. After fusion this hence combines as even+odd = odd boundary wires with Z-support. If instead an even number of internal wires has Z-support, then the non-Clifford spider has an odd number of boundary wires with Z-support and the Clifford spider has an odd amount, so that again we have even+odd = odd boundary wires with Z-support after fusion. Hence in both cases $f(v')$ must have Z-support on oddly many adjacent wires and X-support on no adjacent wires, so that $f(v')$ satisfies the flow condition.

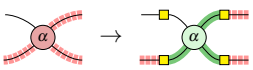
Finally, for any other non-Clifford spider v'', if $f(v'')^-$ has a defect at v', then it previously had a defect at v, so $v'' \preceq v'$, so that the defects still respect $\preceq$.

Applying (sp) from right-to-left is similar, except we extend each logical and flow semiweb $\vec{w}$ to a new semiweb $\vec{w}^+$ which may have additional support on the newly-created wire(s) between the unfused spiders. We do this such that the Pauli web condition is satisfied for the Clifford spider on the LHS. Namely, the new wires have X-support if and only if the boundary wires do, and we introduce Z-support on 0 or 1 wires in order to make the Z-support of the Clifford spider even. Intuitively, we can think of whether we colour the connecting edge as capturing whether the Pauli semiweb "exits" through the Clifford spider. For example:

Case 0:  Case 1: 

This will imply that the Z-support of $f(v')^+$ is odd, so we can set $f(v) := f(v')^+$.

The arguments for (cc) and (π) are very similar, as the LHS and RHS both contain exactly 1 non-Clifford spider. In fact, these cases are a bit simpler, as the extended Pauli semiwebs when applying the rules from right-to-left are uniquely fixed. This is because the H-gates in the (cc) and the π-spiders in the (π) uniquely fix the colour of the internal wires based on the colour of the inputs/outputs. For example:



Applying (s2) left-to-right trivially preserves ZX-flow. From right-to-left, we can give the non-Clifford spider a flow semiweb just by introducing Z-support on the one wire in the LHS. □

ZX-flow is also preserved by spider (un)fusions involving non-Clifford angles, as long as we don't use it to create non-Clifford spiders at a location where there was previously only a Clifford spider.

Theorem 4.10. Spider fusion between a pair of non-Clifford spiders preserves ZX-flow from left-to-right, and it preserves ZX-flow from right-to-left as long as we don't unfuse a Clifford spider into a pair of non-Clifford spiders.

Proof. When we apply the rule from left-to-right, the only case not covered by Theorem 4.9 is when α and β are both non-Clifford. As in the proof of that theorem, we define new Pauli semiwebs by restricting the old ones. If $\alpha+\beta$ is Clifford, we are done. If $\alpha+\beta$ is non-Clifford, the spider on the RHS v' inherits the flow semiweb and its position w.r.t. $\preceq$ from either spider v on the LHS.

Now let v'' be any other spider in the diagram we apply the rewrite to and suppose it has a defect at one of the spiders in the LHS. We can assume without loss of generality that the ZX-flow is focused, hence this defect must not be a π-defect. Hence, $f(v'')$ must have X-support on all the edges in the LHS of (sp'), so it must have a defect at both spiders, so v'' is before both spiders. This implies that if $f(v'')^-$ has a defect at the spider v' on the RHS of (sp'), then $v'' \preceq v'$.

Applying the rule from right-to-left, again we only need to consider the case where both spiders on the LHS are non-Clifford. We place them both at the same location as the spider on the RHS w.r.t. $\preceq$. The ordering between the two spiders on the LHS is irrelevant. We extend the Pauli semiwebs as described in the proof of Theorem 4.9 and note that any spider v'' with a defect at one of the two spiders on the LHS must be in the past, as it would have also had a defect on the spider in the RHS. □

4.2 Equivalence of ZX-Flow and Pauli flow

Once we have a strong ZX-flow, we can define a Pauli flow by setting correction sets $c(v) := B$, where B is a generating set of spiders for $f(v)$ (cf. Theorem 3.6). Conversely, correction sets give flow semiwebs by taking products of basic semiwebs b_v for $v \in c(v)$. This gives the following equivalence, which we prove in detail in Appendix B.

Lemma 4.11. A graph-like ZX-diagram has strong ZX-flow if and only if its induced labelled open graph has Pauli flow.

A consequence of this lemma and Theorem 4.9 is that any ZX-diagram that is Clifford-equivalent to one with Pauli flow has ZX-flow. For the converse, if we start with a ZX-diagram that has ZX-flow, we can use Clifford rewrites to make it graph-like and remove all unnecessary Clifford spiders. Once we do this, we can show that the ZX-flow is actually now a strong ZX-flow, hence we can apply the lemma above to get a Pauli flow. Using this technique, we prove one of our main results.

Theorem 4.12. A ZX-diagram D has ZX-flow if and only if it is Clifford-equivalent to a graph-like ZX-diagram D' with Pauli flow.

Proof. Suppose D is Clifford-equivalent to some graph-like D' with Pauli flow. Then, D' has strong ZX-flow by Lemma 4.11. By Corollary 4.7 D' then also has regular ZX-flow. Then, by Theorem 4.9, D must have ZX-flow.

Conversely, assume D has ZX-flow. Using the Clifford simplification routine from e.g. [15], we can apply Clifford rewrites to reduce D to its "skeleton" D', i.e. a graph-like diagram whose only interior Clifford spiders are part of *phase gadgets*, and which might have Hadamards on the boundary wires. That is, every interior Clifford spider v must be connected via an H-edge to a 1-legged non-Clifford spider v'. D' must have a ZX-flow by Theorem 4.9, but we can furthermore give it a strong ZX-flow by placing every interior Clifford spider v directly before its associated 1-legged non-Clifford spider v' in $\preceq$. We then let $f(v) := \vec{b}_{v'}$, which only covers the H-edge between v' and v. This basic semiweb has a π defect at v and another defect at v', so it is a valid flow semiweb for v.

Place input Clifford spiders as minimal elements in $\preceq$. For a Clifford spider v adjacent to an input wire i, we construct $f(v)$ by multiplying one of the two logical semiwebs $\ell_Z(i)$ or $\ell_X(i)$ with an edge semiweb that contains i to remove its support on the input wire. Output Clifford spiders can be placed arbitrarily w.r.t. $\preceq$, and we let $f(v)$ be the edge semiweb that contains its adjacent output.

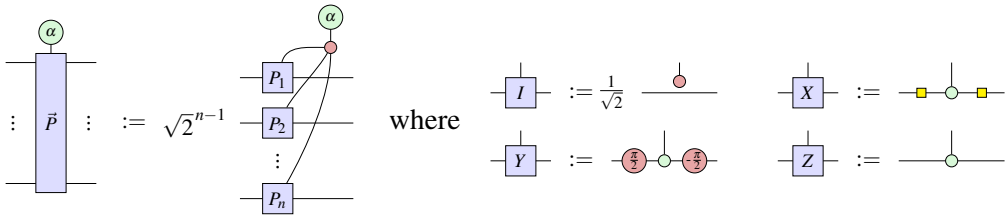
We have then shown that D' has strong ZX-flow. In order to invoke Lemma 4.11, we must first make it fully graph-like, which means we need to introduce identity spiders on those boundary wires that contain Hadamards. Introducing these identities trivially preserves the desired properties of all the existing semiwebs, so we just have to find semiwebs for these identity spiders. Since these are all boundary spiders, we do this in the same way as we did with the previous boundary spiders.

We have thus shown that D' has a strong ZX-flow and is graph-like, so by Lemma 4.11, it has Pauli flow. □

5 Circuit extraction

We will now show how we can perform circuit extraction for a ZX-diagram with ZX-flow. The main idea is that we can interpret a ZX-diagram with a focused ZX-flow directly as a Clifford circuit, followed by a sequence of *Pauli exponentials* (a.k.a. Pauli gadgets or Pauli-product rotations).

Definition 5.1. For a Pauli operator $\vec{P} = P_1 \otimes \ldots \otimes P_n$ and angle $\alpha \in \mathbb{R}$, a *Pauli exponential* is an n-qubit unitary of the form $\vec{P}[\alpha] := e^{-\frac{i\alpha}{2}\vec{P}}$, or diagrammatically:

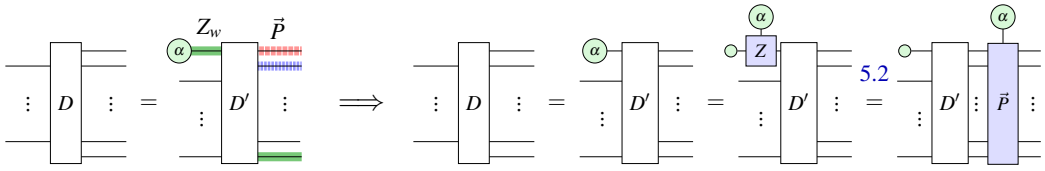


A useful feature of Pauli gadgets is that they commute through other maps in exactly the same way as Pauli operators. The following can be shown by direct calculation or using the Clifford ZX-calculus rules (see e.g. [16]).

Proposition 5.2. For a linear map $L : (\mathbb{C}^2)^{\otimes n} \to (\mathbb{C}^2)^{\otimes m}$ and Pauli operators $\vec{P} \in \mathscr{P}_n, \vec{Q} \in \mathscr{P}_m$, if $L\vec{P} = \vec{Q}L$, then $L\vec{P}[\alpha] = \vec{Q}[\alpha]L$.

Combining this with Theorem 2.4, we can see that Pauli webs enable us to push Pauli exponentials through a ZX-diagram. Suppose we take a ZX-diagram D with focused ZX-flow and pick a non-Clifford spider that is maximal with respect to $\preceq$. If it is a Z-spider, we unfuse a 1-legged Z-spider v with phase angle α connected via a plain edge. If it is an X-spider, we unfuse a 1-legged Z-spider connected via an H-edge. In either case, we obtain a flow semiweb that has Z-support on the wire w adjacent to v and it has support on the outputs given by some Pauli $\vec{P}$. We will call the restriction $\vec{P}_O$ of a semiweb to the outputs of a diagram its *output colouring*.

Since v is maximal and the ZX-flow is focused, $f(v)$ only has a defect at v itself, so we can treat $f(v)$ as a Pauli web on a sub-diagram D' which contains everything in D except for v. We then use this Pauli web to push the non-Clifford angle α out of the diagram as follows:

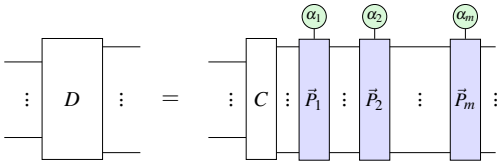


This strictly reduces the number of non-Clifford spiders in D, so at some point this will terminate when D' is a Clifford ZX-diagram. Because the ZX-flow is focused, the logical semiwebs $\ell_Z(i), \ell_X(i)$ have no defects, so that they are logical Pauli webs. This will give us a stabiliser tableau for the initial Clifford part.

Using known procedures to synthesise Clifford isometries and Pauli exponentials, this gives us a complete procedure for extracting a ZX-diagram into a basic set of gates such as CNOT, H, $Z[\alpha]$.

Even though we showed this extraction procedure by rewriting the diagram, we can already read the form of the final extracted circuit from the focused ZX-flow data. This gives us the following theorem.

Theorem 5.3. Let D be a ZX-diagram with focused ZX-flow. Fix an ordering of non-Clifford spiders $v_1, \ldots, v_m$ consistent with $\preceq$ such that the angle of v_k is α_k and the output colouring of $f(v_k)$ is $\vec{P}_k$, then:



where C is a Clifford isometry with logical operators $\vec{\mathscr{Z}}_i, \vec{\mathscr{X}}_i$ given by the output colourings of $\ell_Z(i)$, respectively $\ell_X(i)$, on D.

6 Conclusion

We defined a new flow condition that is naturally suited to generic ZX-diagrams, which allows us to work with a broader class of diagrams while still retaining the ability to extract quantum computations, either via deterministic measurement patterns or quantum circuits. Whereas previous conditions were very closely related to graph states and the one-way model of measurement-based quantum computing, ZX-flow can be thought of as a "multi-paradigm" concept of runnability. Theorem 5.3, where we give a direct interpretation of a focused ZX-flow as a unitary circuit in a particular form, gives evidence of this. It would be interesting to explore other interpretations of ZX-flow in other quantum computational paradigms such as Pauli-based computation and lattice surgery. It may be useful when thinking about these other paradigms to think about a "Heisenberg-style" formulation of ZX-flow, where logical and flow semiwebs have support backward in time rather than forward, reflecting changes to the relevant Pauli observables that might happen as a result of past measurements.

Another direction for future exploration is efficiently finding ZX-flow when it exists. Identifying the necessary Pauli webs amounts to solving systems of $\mathbb{F}_2$-linear equations, so one could imagine a naïve procedure where we pick non-Clifford spiders one at a time and try to assign them flow semiwebs using spiders we have already seen as defects. This procedure is likely to run in $O(n^4)$, but perhaps one could do better by incorporating ideas from the efficient Pauli flow finding algorithm of [19].

Acknowledgements: AK is supported by the Engineering and Physical Sciences Research Council grant number EP/Z002230/1, *(De)constructing quantum software (DeQS)*. JvdW acknowledges support by a Veni grant from the Dutch Research Council (NWO). An LLM was used to proofread this paper and check for minor notational errors.

References

[1] Miriam Backens (2014): *The ZX-calculus is complete for stabilizer quantum mechanics*. *New Journal of Physics* 16(9), p. 093021, doi:10.1088/1367-2630/16/9/093021.

[2] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski & John van de Wetering (2021): *There and back again: A circuit extraction tale*. *Quantum* 5, p. 421, doi:10.22331/q-2021-03-25-421.

[3] Miriam Backens & Thomas Perez (2025): *Inserting Planar-Measured Qubits into MBQC Patterns while Preserving Flow*. In Alejandro Díaz-Caro, Ognyan Oreshkov & Ana Belén Sainz, editors: *Proceedings of the 22nd International Conference on Quantum Physics and Logic, Varna, Bulgaria, 14 July 2025 - 18 July 2025*, *Electronic Proceedings in Theoretical Computer Science* 426, Open Publishing Association, pp. 100–126, doi:10.4204/EPTCS.426.4.

[4] Andreas Bauer & Julio C. Magdalena de la Fuente (2025): *Planar fault-tolerant circuits for non-Clifford gates on the 2D color code*. *arXiv preprint arXiv:2505.05175*.

[5] Niel de Beaudrap, Aleks Kissinger & John van de Wetering (2022): *Circuit Extraction for ZX-Diagrams Can Be #P-Hard*. In Mikołaj Bojańczyk, Emanuela Merelli & David P. Woodruff, editors: *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, *Leibniz International Proceedings in Informatics (LIPIcs)* 229, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 119:1–119:19, doi:10.4230/LIPIcs.ICALP.2022.119.

[6] Hector Bombin, Daniel Litinski, Naomi Nickerson, Fernando Pastawski & Sam Roberts (2024): *Unifying flavors of fault tolerance with the ZX calculus*. *Quantum* 8, p. 1379, doi:10.22331/q-2024-06-18-1379.

[7] Daniel E Browne, Elham Kashefi, Mehdi Mhalla & Simon Perdrix (2007): *Generalized flow and determinism in measurement-based quantum computation*. *New Journal of Physics* 9(8), p. 250.

[8] Bob Coecke & Ross Duncan (2008): *Interacting quantum observables*. In: *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP)*, Lecture Notes in Computer Science, doi:10.1007/978-3-540-70583-3_25.

[9] Ross Duncan, Aleks Kissinger, Simon Perdrix & John van de Wetering (2020): *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*. *Quantum* 4, p. 279, doi:10.22331/q-2020-06-04-279.

[10] Giovanni de Felice, Boldizsár Poór, Cole Comfort, Lia Yeh, Mateusz Kupper, William Cashman & Bob Coecke (2026): *A dataflow programming framework for linear optical distributed quantum computing*. *Quantum* 10, p. 1972, doi:10.22331/q-2026-01-19-1972.

[11] Tobias Fischbach, Pierre Talbot & Pascal Bourvry (2025): *A Review on Quantum Circuit Optimization using ZX-Calculus*. *arXiv preprint arXiv:2509.20663*.

[12] Peter-Jan H. S. Derks, Alex Townsend-Teague, Jens Eisert, Markus S. Kesselring, Oscar Higgott & Benjamin J. Brown (2025): *Dynamical codes for hardware with noisy readouts*. *arXiv preprint arXiv:2505.07658*.

[13] Liam Hurwitz, Kamalika Datta, Abhoy Kole & Rolf Drechsler (2024): *Is Simulation the only Alternative for Effective Verification of Dynamic Quantum Circuits?* In: *International Conference on Reversible Computation*, Springer, pp. 201–217, doi:10.1007/978-3-031-62076-8_13.

[14] Aleks Kissinger & John van de Wetering (2019): *Universal MBQC with generalised parity-phase interactions and Pauli measurements*. *Quantum* 3, doi:10.22331/q-2019-04-26-134.

[15] Aleks Kissinger & John van de Wetering (2020): *Reducing T-count with the ZX-calculus*. *Physical Review A* 102, p. 022406, doi:10.1103/PhysRevA.102.022406.

[16] Aleks Kissinger & John van de Wetering (2024): *Picturing Quantum Software: An Introduction to the ZX-Calculus and Quantum Compilation*. Preprint.

[17] Tommy McElvanney (2025): *Preservation of determinism in MBQC under ZX-calculus rewrites*. Ph.D. thesis, University of Birmingham.

[18] Tommy McElvanney & Miriam Backens (2023): *Flow-preserving ZX-calculus Rewrite Rules for Optimisation and Obfuscation*. In Shane Mansfield, Benoit Valîron & Vladimir Zamdzhiev, editors: *Proceedings of the Twentieth International Conference on Quantum Physics and Logic, Paris, France, 17-21st July 2023*, *Electronic Proceedings in Theoretical Computer Science* 384, Open Publishing Association, pp. 203–219, doi:10.4204/EPTCS.384.12.

[19] Piotr Mitosek & Miriam Backens (2024): *An algebraic interpretation of Pauli flow, leading to faster flow-finding algorithms*. arXiv preprint arXiv:2410.23439.

[20] Maike Ostmann, Joshua Nunn & Alex E. Jones (2025): *Nonlinear photonic architecture for fault-tolerant quantum computing*. arXiv preprint arXiv:2510.06890.

[21] Tom Peham, Lukas Burgholzer & Robert Wille (2022): *Equivalence Checking of Parameterized Quantum Circuits: Verifying the Compilation of Variational Quantum Algorithms*. arXiv preprint arXiv:2210.12166, doi:10.1145/3566097.3567932.

[22] Boldizsár Poór, Benjamin Rodatz & Aleks Kissinger (2025): *Ultra Low Overhead Syndrome Extraction for the Steane code*. arXiv preprint arXiv:2511.13700.

[23] Benjamin Rodatz, Boldizsár Poór & Aleks Kissinger (2025): *Fault Tolerance by Construction*. arXiv preprint arXiv:2506.17181.

[24] Will Simmons (2021): *Relating Measurement Patterns to Circuits via Pauli Flow*. In Chris Heunen & Miriam Backens, editors: *Proceedings 18th International Conference on Quantum Physics and Logic, Gdansk, Poland, and online, 7-11 June 2021*, *Electronic Proceedings in Theoretical Computer Science* 343, Open Publishing Association, pp. 50–101, doi:10.4204/EPTCS.343.4.

[25] Tobias Stollenwerk & Stuart Hadfield (2023): *Diagrammatic Analysis for Parameterized Quantum Circuits*. In Stefano Gogioso & Matty Hoban, editors: *Proceedings 19th International Conference on Quantum Physics and Logic, Wolfson College, Oxford, UK, 27 June - 1 July 2022*, *Electronic Proceedings in Theoretical Computer Science* 394, Open Publishing Association, pp. 262–301, doi:10.4204/EPTCS.394.15.

[26] Tobias Stollenwerk & Stuart Hadfield (2024): *Measurement-Based Quantum Approximate Optimization*. In: *2024 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 1115–1127, doi:10.1109/IPDPSW63119.2024.00183.

[27] Dimitrios Thanos, Alejandro Villoria, Sebastiaan Brand, Arend-Jan Quist, Jingyi Mei, Tim Coopmans & Alfons Laarman (2024): *Automated reasoning in quantum circuit compilation*. In: *Model Checking Software (SPIN)*.

[28] Vivien Vandaele (2025): *Optimization of fault-tolerant quantum computing by the ZX-calculus*. Ph.D. thesis, Université de Lorraine.

[29] Renaud Vilmart (2019): *A Near-Minimal Axiomatisation of ZX-Calculus for Pure Qubit Quantum Mechanics*. In: *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pp. 1–10, doi:10.1109/LICS.2019.8785765.

[30] Quanlong Wang, Richie Yeung & Mark Koch (2024): *Differentiating and Integrating ZX Diagrams with Applications to Quantum Machine Learning*. *Quantum* 8, p. 1491, doi:10.22331/q-2024-10-04-1491.

[31] John van de Wetering, Richie Yeung, Tuomas Laakkonen & Aleks Kissinger (2025): *Optimal compilation of parametrised quantum circuits*. *Quantum* 9, p. 1828, doi:10.22331/q-2025-08-27-1828. Available at https://doi.org/10.22331/q-2025-08-27-1828.

[32] Junyu Zhou, Yuhao Liu, Ethan Decker, Justin Kalloor, Mathias Weiden, Kean Chen, Costin Iancu & Gushu Li (2026): *TopoLS: Lattice Surgery Compilation via Topological Program Transformations*. arXiv preprint arXiv:2601.23109.

A Correctness of the Definition of Pauli flow

We will prove that Definition 2.11 is equivalent to the standard one from the literature. For this, we will compare to the version that closely resembles the one given by Mitosek and Backens [19], which is an equivalent, more symmetric version of [7].

Theorem A.1. For an open graph (G,I,O) with a choice of measurements μ, a pair $(\preceq, c)$ is a Pauli flow in the sense of Definition 2.11 if and only if:

(P1) $\mu(v) \notin \{X,Y\}, u \not\preceq v \implies v \notin c(u)$

(P2) $\mu(v) \notin \{Y,Z\}, u \not\preceq v \implies v \notin Odd(c(u))$

(P3) $\mu(v) \notin \{X,Z\}, u \not\preceq v \implies v \notin c(u) \Delta Odd(c(u))$

(P4) $\mu(u) = XY \implies u \notin c(u) \wedge u \in Odd(c(u))$

(P5) $\mu(u) = XZ \implies u \in c(u) \wedge u \in Odd(c(u))$

(P6) $\mu(u) = YZ \implies u \in c(u) \wedge u \notin Odd(c(u))$

(P7) $\mu(u) = X \implies u \in Odd(c(u))$

(P8) $\mu(u) = Z \implies u \in c(u)$

(P9) $\mu(u) = Y \implies u \in c(u) \Delta Odd(c(u))$

Proof. First, assume $(\preceq, \mu, c)$ satisfies conditions (i) and (ii) from Definition 2.11. Condition (i) implies (P4)-(P9). Condition (ii) implies (P1)-(P3).

Conversely, assume the criteria (P1)-(P9) from the Theorem. We begin by showing condition (i) of Definition 2.11 by case distinction on the Pauli $P \in \mu(u)$. If X is in $\mu(u)$, then the predicate from either (P4), (P5), or (P7) is satisfied. In any of these cases, $u \in Odd(c(u)) = A_X(u)$. Similarly, if $Y \in \mu(u)$, the predicate from (P4), (P6), or (P9) is satisfied. Each of these cases imply the u is in the symmetric difference $c(u) \Delta Odd(c(u)) = A_Y(u)$. If $Z \in \mu(u)$, then the predicate from (P5), (P6), or (P8) is satisfied, hence $u \in c(u) = A_Z(u)$.

Condition (ii) follows similarly by case distinction. It can be equivalently stated as $P \in \mu(v), u \not\preceq v \implies v \notin A_P(u)$, so take some node v and some $u \not\preceq v$. If $Z \in \mu(v)$, then (P1) implies $v \notin c(u) = A_Z(u)$. If $Y \in \mu(v)$, then (P3) implies $v \notin c(u) \Delta Odd(c(u)) = A_Y(u)$. Finally, if $X \in \mu(v)$, then (P2) implies $v \notin Odd(c(u)) = A_X(u)$. □

B Pauli Flow Equivalence Proofs

Proof of Lemma 4.11. Let $(\preceq, \ell_Z, \ell_X, f)$ be a strong ZX-flow for D. For every non-output spider v, let the correction set $c(v)$ be the generating set of spiders for the Pauli semiweb $f(v)$, as defined in Theorem 3.6.

It must be the case that $f(v)$ has a π-defect at v. We will show that this implies the Pauli flow conditions at v for each of the measurement labels. First note that, since v is not adjacent to an output and inputs must be un-coloured by $f(v)$, any Z-coloured wires adjacent to v must come from basic semiwebs in $c(v)$.

- If $X \in \mu(v)$, then the phase of v is not an odd multiple of $\frac{\pi}{2}$. Hence, the only way $f(v)$ can have a π-defect at v is if there is an odd number of adjacent wires with Z-support, which implies v is in the odd neighbourhood of $c(v)$.

- If $Y \in \mu(v)$, then the phase of v is not a multiple of π. Hence, it must either be the case that (1) no adjacent wires have X-support and oddly many adjacent wires have Z-support, or (2) all adjacent edges have X-support and evenly-many adjacent wires have Z-support. This implies that v is in the symmetric difference of $c(v)$ and the odd neighborhood of $c(v)$.

Now, consider another spider v':

- If $X \in \mu(v')$ then the phase of v' is not an odd multiple of $\frac{\pi}{2}$. Hence, if v' is in the odd neighbourhood of $c(v)$, v must have a π-defect at v' so that $v \preceq v'$.
- If $Y \in \mu(v')$, then the phase α of v' is not a multiple of π. Hence, if v' is in the symmetric difference of $c(v)$ and $Odd(c(v))$, then it is either in $Odd(c(v))$ so that $f(v)$ has a π-defect due to the odd number of Z-support edges, or it is in $c(v)$ so that v' has all X-support and $f(v)$ has a 2α-defect there. In both cases $v \preceq v'$.

Hence, $(c, \mu, \preceq)$ satisfies the conditions of Pauli flow of Definition 2.11.

Now for the converse, let $(\preceq', \mu, c)$ be a Pauli flow for D. Let $\preceq$ be the extension of $\preceq'$ to all spiders, including output spiders, where output spiders are maximal with respect to $\preceq$.

We first define the flow semiwebs for D. Since we want a strong ZX-flow, we need to do it for all spiders, not just the non-Clifford ones.

For each output spider, we let $f(v)$ be the Pauli semiweb supported on the unique output wire adjacent to v. For each non-output spider, we let $f(v) := \prod_{v' \in c(v)} \vec{b}_{v'}$ where $\vec{b}_v$ is the basic semiweb at v (Definition 3.4).

If $\mu(v) = X$, its phase must be a multiple of π and $v \in Odd(c(v))$, so $f(v)$ has oddly many adjacent wires with Z-support, and hence has a π-defect. If $\mu(v) = Y$, its phase is $\pm\frac{\pi}{2}$ and v is either in $c(v)$ or $Odd(c(v))$, but not both. In either case, $f(v)$ has a π-defect. Finally, if $\mu(v) = XY$, then oddly many adjacent wires have Z-support and no adjacent wires have X-support, so it has a π-defect at v.

Now, consider another spider v' such that $f(v)$ has a defect at v'. If v' is adjacent to an output, then $v \preceq v'$. If it is a non-output, then if its phase is a multiple of π, then oddly many neighbouring wires must have Z-support in $f(v)$ for it to have a defect, so $v' \in Odd(c(v))$ and hence $v \preceq v'$. If the phase of v' is an odd multiple of $\frac{\pi}{2}$, then it can only have a defect if $v' \in c(v)$ or $v' \in Odd(c(v))$, but not both, so that $v' \in A_Y(v)$. Hence, also $v \preceq v'$. If instead the phase of v' is non-Clifford, $v' \in c(v)$ or $v' \in Odd(c(v))$. So, v' must either be in $A_X(v)$ or $A_Y(v)$, so that also $v \preceq v'$. All the flow semiwebs hence have the correct properties with respect to $\preceq$.

Finally, we construct the logical semiwebs as follows. $\ell_Z(i)$ is the edge semiweb supported on input i. This has Z-support just on input wire i and no X-support. We construct the logical semiweb $\ell_X(i)$ as the basic semiweb b_v of the spider v that is adjacent to input i. Since the diagram is graph-like, this basic semiweb restricted to inputs is X_i, as required. □