Pauli Flow on Open Graphs with Unknown Measurement Labels

Piotr Mitosek
School of Computer Science, University of Birmingham
pbm148@student.bham.ac.uk

One-way quantum computation, or measurement-based quantum computation, is a universal model of quantum computation alternative to the circuit model. The computation progresses by measurements of a pre-prepared resource state together with corrections of undesired outcomes via applications of Pauli gates to yet unmeasured qubits. The fundamental question of this model is determining whether computation can be implemented deterministically. Pauli flow is one of the most general structures guaranteeing determinism. It is also essential for polynomial time ancilla-free circuit extraction. It is known how to efficiently determine the existence of Pauli flow given an open graph together with a measurement labelling (a choice of measurements to be performed).

In this work, we focus on the problem of deciding the existence of Pauli flow for a given open graph when the measurement labelling is unknown. We show that this problem is in RP by providing a random polynomial time algorithm. To do it, we extend previous algebraic interpretations of Pauli flow, by showing that, in the case of X and Z measurements only, flow existence corresponds to the right-invertibility of a matrix derived from the adjacency matrix. We also use this interpretation to show that the number of output qubits can always be reduced to match the number of input qubits while preserving the existence of flow.

1 Introduction

One-way quantum computation, or measurement-based quantum computation (MBQC), is a quantum computation model alternative to the circuit model, yet at least as powerful [30, 31, 32]. The computation is performed entirely by a series of measurements on a pre-prepared resource state. MBQC is expected to become critical for quantum communication applications. For instance, MBQC is used in blind quantum computation [4], a quantum protocol allowing a client to outsource part of the computation to a server without compromising the client's privacy. However, the quantum measurements are random by their nature, which means that in each step two different outcomes may be obtained, where only one is the desired outcome. Therefore, it is the fundamental problem of MBQC to determine whether the desired computation can be performed deterministically.

A structure essential for deterministic MBQC is flow defined on a labelled open graph – a graph state with designated input and output qubits together with a choice of measurements for non-outputs. The computation is performed by a series of measurements of non-outputs in one of six settings: planar XY, XZ, and YZ measurements and Pauli X, Y, and Z measurements. After each measurement, a correction might be applied which is intended to fix a potential undesired outcome, bringing the open graph into a state equivalent to one that would be reached if a desired outcome was observed. The corrections are performed by applying Pauli gates to yet unmeasured qubits. One of the notions of determinism for such computation corresponds to the existence of flow structure in the labelled open graph. There are many different types of flow, ranging from the causal flow [8], through generalized flow [5], all the way to the Pauli flow [5] and its extended variant [27]. Every causal flow is also a generalized flow. Every

A. Díaz-Caro and V. Zamdzhiev (Eds.):
Quantum Physics and Logic 2024 (QPL 2024)
EPTCS 406, 2024, pp. 117–136, doi:10.4204/EPTCS.406.6

generalized flow is also a Pauli flow. However, the reverses are not true, thus making Pauli flow the most universal.

The flow is also crucial when working with ZX calculus [7, 37], a graphical calculus for quantum computation, where the notions of flow correspond to the most general subclasses of diagrams for which a polynomial time ancilla-free circuit extraction algorithm exists [2, 34]. While diagrams from graphical calculi are easier to work with than circuits, they are not a notion of computation understood by quantum computers. For that, a corresponding circuit must be found, however, circuit extraction in general is #P-hard [11]. In [13], circuits are optimized by translating them to ZX, simplifying diagrams, and translating them back to circuits. Other versions of this approach have appeared since for example in [35]. There, diagram transformations necessarily must preserve flow, which was a subject of for instance [23, 24].

While the existence of different flow variants can be efficiently determined for a given labelled open graph [10, 9, 26, 2, 34], many questions remain open. For instance, what are other properties of labelled open graphs with flow? Previous works looking at the necessary properties the graph must satisfy to contain flow include [25, 22, 34]. We contribute to the general picture of what flow is and which graphs exhibit it.

Results overview In this work, we show that given an (unlabelled) open graph it is possible to efficiently determine whether there exists a choice of measurements resulting in flow. In particular, we show the following problem is in RP, meaning that there exists a random polynomial-time algorithm solving it.

> **FlowSearch**
> **Input:** An open graph (G, I, O).
> **Output:** $True$ if there exists a measurement labelling λ such that the labelled open graph (G, I, O, λ) exhibits Pauli flow, and $False$ otherwise.

To show that, we adapt the algebraic interpretation of flow from [25]. In that paper, the authors relate the existence of generalised flow for XY planar measurements only to the existence of the right inverse of a particular matrix satisfying additional requirements. We relate Pauli flow in the case of only X and Z measurement to just right invertibility. Then, we use the algebraic interpretation to reduce **FlowSearch** to **MaxRank**, a problem about maximizing the rank of a matrix whose entries can contain variables. Our approach is limited to finding labelling consisting of Pauli X and Z measurements. The corresponding measurement scheme is Clifford. Thus, the current algorithm may not be ideal for searching more complex labels which make the corresponding labelled open graph have Pauli flow. However, as we will prove, the graph having Pauli flow for some measurement labels also has flow for labels consisting of only Pauli X and Z measurements. In other words, if a graph does not have flow for some label consisting of just X and Z measurements then it cannot have flow for any label. Therefore, our approach is sufficient to solve **FlowSearch**.

Further, we use the algebraic interpretation of flow to explore the conditions on the sets of inputs and outputs necessary to have flow. In particular, we prove that if a labelled open graph has Pauli flow, then some outputs can be removed and the resulting open graph will still have Pauli flow for some measurement labelling.

Structure In section 2, we present the background. Section 3 is the main part of the paper. There, we formally describe our results and prove them. Finally, in section 4, we discuss the conclusions and possible further work. We include some technical proofs and additional examples in the appendices.

2 Background

In this section, we first look at the general concept of measurement-based quantum computation, where we define Pauli flow. Next, we explain computational complexity definitions critical in our proofs.

2.1 Measurement-Based Quantum Computation

As outlined earlier, we focus on the version of MBQC where measurements are performed on open graphs and the allowed measurements are single-qubit: Pauli measurements or planar measurements from XY, YZ, or XZ planes. The description of the computation is given in the form of measurement patterns. The labelled open graphs can be viewed as runnable measurement patterns without corrections [2].

Definition 2.1 (Open Graph)**.** An **open graph** is a triple (G, I, O) consisting of:

- an undirected graph $G = (V, E)$,
- a set of **inputs** $I \subseteq V$,
- a set of **outputs** $O \subseteq V$.

We define the sets of **non-inputs** $\overline{I} := V \setminus I$, **non-outputs** $\overline{O} := V \setminus O$ and **internal vertices** $B := V \setminus (I \cup O) = \overline{I} \cap \overline{O}$.

An **odd neighbourhood** of a set of vertices A is denoted $Odd(A)$ and consists of all vertices neighbouring an odd number of elements of A: $Odd(A) := \{v \in V \mid \#\{a \in A \mid va \in E\} \text{ is odd}\}$.

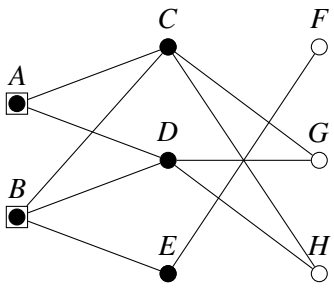
Definition 2.2. A **measurement labelling** for an open graph (G, I, O) is any function λ sending non-outputs $\overline{O}$ to labels $\{X, Y, Z, XZ, XY, YZ\}$, satisfying $\lambda(v) \in \{X, XY, Y\}$ for all $v \in I \setminus O$. A **labelled open graph** is a quadruple (G, I, O, λ) where (G, I, O) is an open graph and λ is a measurement labelling.

An open graph is prepared by entangling input qubits, corresponding to the vertices in I, with qubits prepared in $|+\rangle$ corresponding to the vertices in $\overline{I}$, by applying CZ gate between pair of qubits corresponding to each edge in the graph. Since the CZ gates commute, the order of CZ applications does not matter. When the set of inputs is empty, the notion of an open graph collapses to the graph state. Next, qubits corresponding to the elements of $\overline{O}$ are measured according to the measurement labelling. See figure 1 for an example.

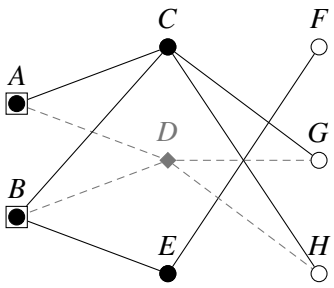
The computation given in the form of the measurement pattern depends on the outcomes of the measurements. After each measurement, an undesired outcome may occur and it must be corrected. The corrections are performed by applying Pauli X and Z gates to the yet unmeasured qubits. The result of a chosen measurement outcome, together with corrections, is called a branch. There are many different notions of determinism [5, 25]. The one we focus on is the strong, uniform, and stepwise determinism – all branches of the computation are equal up to a global phase, for any choice of measurement angles, and the intermediate patterns after performing a subset of measurements also have these properties [5, 25, 2]. The measurement pattern is strongly, uniformly, and stepwise deterministic when the corresponding labelled open graph has Pauli flow [5]. We define the notion of Pauli flow (based on [34]).

Definition 2.3 (Pauli flow)**.** A **Pauli flow** for a labelled open graph (G, I, O, λ) is a pair $(c, \prec)$ where c is a function $\overline{O} \to \mathcal{P}(\overline{I})$ and $\prec$ is a strict partial order on $\overline{O}$, such that for all $u \in \overline{O}$:

(P1) $\forall v \in c(u).u \neq v \wedge \lambda(v) \notin \{X, Y\} \Rightarrow u \prec v$
(P2) $\forall v \in Odd(c(u)).u \neq v \wedge \lambda(v) \notin \{Y, Z\} \Rightarrow u \prec v$
(P3) $\forall v \in \overline{O}.\neg(u \prec v) \wedge u \neq v \wedge \lambda(v) = Y \Rightarrow (v \in c(u) \Leftrightarrow v \in Odd(c(u)))$
(P4) $\lambda(u) = XY \Rightarrow u \notin c(u) \wedge u \in Odd(c(u))$
(P5) $\lambda(u) = XZ \Rightarrow u \in c(u) \wedge u \in Odd(c(u))$



(a) A labelled open graph with two inputs and three outputs. All non-outputs are X labelled.



(b) The labelled open graph from 1a, but with vertex D labelled Z.

Figure 1: Examples of labelled open graphs. Outputs are denoted by empty circles, X labelled non-outputs by filled circles and inputs by a square. The Z labelled vertices are denoted with grey diamonds and edges including them are dashed and grey.

(P6) $\lambda(u) = YZ \Rightarrow u \in c(u) \wedge u \notin Odd(c(u))$
(P7) $\lambda(u) = X \Rightarrow u \in Odd(c(u))$
(P8) $\lambda(u) = Z \Rightarrow u \in c(u)$
(P9) $\lambda(u) = Y \Rightarrow (u \in c(u) \oplus u \in Odd(c(u)))$, where $\oplus$ stands for XOR.

We call c the **correction function** and sets $c(v)$ for $v \in \overline{O}$ are called the **correction sets**.

Example 2.4. Consider open graphs from figure 1. The graph from 1a does not have Pauli flow. On the other hand, the graph from 1b has Pauli flow. For instance, taking $D \prec A$ and $D \prec B$ with the following correction function result in the flow:

$$c(A) = \{C, E\}, \quad Odd(c(A)) = \{A, F, G, H\}, \quad c(B) = \{E\}, \quad Odd(c(B)) = \{B, F\},$$
$$c(C) = \{G\}, \quad Odd(c(C)) = \{C, D\}, \quad c(D) = \{D\}, \quad Odd(c(D)) = \{A, B, G, H\},$$
$$c(E) = \{F\}, \quad Odd(c(E)) = \{E\}.$$

In MBQC, the meaning of the Pauli flow is as follows. The prepared open graph state is measured according to the partial order. When a measurement of a vertex u results in an undesired outcome, then the measurement error can be fixed by applying the X gate to all vertices in $c(u)$. The flow guarantees that each correction is physically possible and that every error can be corrected independently of the outcomes of the previous measurements. In other words, the MBQC becomes deterministic. For a detailed explanation, see [34].

2.2 Computational Complexity

We assume familiarity with standard complexity terminology. We are mainly concerned about the **random polynomial time** class RP, where randomness is permitted with one-sided bounded error:

Definition 2.5. The class of RP consists of problems A solvable by a non-deterministic Turing Machine M that given input a proceeds as follows:

- if a is a "NO" instance, then M always rejects a,
- if a is a "YES" instance, then M accepts a on at least half of its computation paths.

For a problem in RP there exists a polytime algorithm with random number generator access that for "NO" instances always returns "NO" and for "YES" instances it returns "YES" with probability at least $\frac{1}{2}$ and otherwise it returns "NO". This last case is the error of the computation and happens with the probability smaller than $\frac{1}{2}$. By running the algorithm multiple times it is possible to reduce the error probability – 50 runs results in error probability below $\frac{1}{2^{50}}$ which is sufficient for all practical applications.

The following is the essential problem in our work. The definition is adapted from [6] (in contrast, we do not require the input matrix to be square).

MaxRank
Fixed: A commutative ring R, and subsets $E, S \subseteq R$ of entries and solutions.
Input: Natural numbers m, n, t, r and $m \times n$ matrix M with entries from $E \cup \{x_1, \ldots, x_t\}$.
Output: $True$ if $\operatorname{rank} M(a_1, \ldots, a_t) \geq r$ for some $a_1 \ldots a_t \in S^t$, and $False$ otherwise.

$M(a_1, \ldots, a_t)$ stands for the matrix with substituted variables $x_1 \mapsto a_1, \ldots, x_t \mapsto a_t$. We skip the specification of m, n, t, as these numbers are explicit from the input matrix. Thus, we will write instances of **MaxRank** as pairs (M, r) of the matrix and the desired minimal rank under some valuation.

In [6], it is shown that when R is a finite field and each variable occurs at most once, the problem is in RP. We extend their approach to show that the problem is in RP also when each variable appears in at most one row or one column. For that, we need the notion of multi-affine polynomials (adapted from [6]). In general, the **MaxRank** problem with R being a finite field is NP-complete [6].

Definition 2.6. A multivariable polynomial is **multi-affine** when each variable has a degree at most one.

By extending the proof [6, Theorem 28], we get the following version. Here, the entries of the matrix are allowed to be given by multi-affine expressions over $E \cup \{x_1, \ldots, x_t\}$, not just elements of the set.

Theorem 2.7. The following version of **MaxRank** is in RP:

- $R = E = S = \mathbb{F}_s$ is a finite field,
- each variable appears in at most one row or at most one column[1],
- the entries are given by (polynomially long) multi-affine expressions over $E \cup \{x_1, \ldots, x_t\}$.

We prove the above theorem in the appendix A, where we also give an example and talk about other known results for **MaxRank** and similar problems.

3 Finding labelling resulting in Pauli flow

We start by formally defining theorems capturing our results. We show that given an open graph (G, I, O), there exists a random polytime algorithm deciding the existence of measurement labelling λ such that the labelled open graph (G, I, O, λ) has Pauli flow, i.e we show that **FlowSearch** defined in the introduction is in RP.

Theorem 3.1. FlowSearch is in RP.

Example 3.2. Consider the open graph from figure 1, ignoring the labelling. When considered as an instance of **FlowSearch**, the answer is $True$, because the labelling from figure 1b results in Pauli flow.

To show the above theorem, we expand on the known [25] correspondence between flow and certain algebraic properties of various matrices. We also show that given an open graph with more outputs than inputs, it is always possible to reduce the number of outputs to match the number of inputs, while preserving the flow.

[1] That is if a variable appears in entries at positions $(a_1, b_1), (a_2, b_2), (a_3, b_3), \ldots$, then either $a_1 = a_2 = a_3 = \ldots$ or $b_1 = b_2 = b_3 = \ldots$.

Theorem 3.3. Suppose (G, I, O, λ) has Pauli flow and $|O| > |I|$. Then there exists a subset $O' \subseteq O$ such that $|O'| = |I|$ and a labelling λ' such that (G, I, O', λ') has Pauli flow.

Labelled open graphs with Pauli flow and equal numbers of inputs and outputs have multiple elegant properties. Firstly, they correspond to a circuit without ancillas and hence a unitary. Further, the graph and flow can be "reversed" ([25, Theorem 3.4] for generalized flow theorem). Next, there is a unique correction function in the focussed Pauli flow. In the case of X and Z labels only, the existence of such a correction function corresponds to the invertibility of what we call the flow matrix, rather than just right-invertibility. Finding the inverse can be implemented with faster algorithms than Gaussian elimination. Thus, it is useful to transform labeled open graphs with more outputs than inputs in order to equalise both sizes.

The remainder of this section are the proofs of the theorems 3.1 and 3.3. The first proof is divided into three subsections. In the first one, we do preliminary simplifications of the **FlowSearch** problem. The second subsection presents an algebraic interpretation of the Pauli flow, by giving a correspondence between Pauli flow and matrix invertibility. Finally, in the third subsection, we show how **FlowSearch** instances can be transformed into **MaxRank** instances, and that those instances can be solved by a random polytime algorithm. The proof of theorem 3.3 is presented in the final subsection.

3.1 Reducing measurement options

We start by reducing the number of possible options for the measurement basis to just Pauli measurements.

Theorem 3.4. Suppose that a labelled open graph (G, I, O, λ) has Pauli flow. Then, there exists $\lambda' : \overline{O} \to \{X, Y, Z\}$ such that (G, I, O, λ') has Pauli flow.

Proof. We start by fixing a Pauli flow $(c, \prec)$ on (G, I, O, λ). The conditions for a planar measurement XY combine the requirements for the two Pauli measurements X and Y. Hence, swapping XY measurements to X preserves $(c, \prec)$ as the Pauli flow. Similarly, we can swap XZ to X and YZ to Z. □

There can be many different Pauli flows for a single labelled open graph. However, there exists a special type of Pauli flow known as the **focussed Pauli flow**. The following definition is based on [34, Definition 4.3].

Definition 3.5 (Focussed Pauli flow)**.** The Pauli flow $(c, \prec)$ is **focussed** when for all $v \in \overline{O}$ the following hold:

(F1) $\forall w \in (\overline{O} \setminus \{v\}) \cap c(v). \lambda(w) \in \{XY, X, Y\}$
(F2) $\forall w \in (\overline{O} \setminus \{v\}) \cap Odd(c(v)). \lambda(w) \in \{XZ, YZ, Y, Z\}$
(F3) $\forall w \in (\overline{O} \setminus \{v\}). \lambda(w) = Y \Rightarrow (w \in c(v) \Leftrightarrow w \in Odd(c(v)))$

Example 3.6. Consider the Pauli flow from the example 2.4 – it is not focussed as $Odd(c(D))$ contains X labelled vertices A and B. Changing $c(D)$ to $\{C, D\}$ results in $Odd(c(D)) = \emptyset$ and the flow becomes focussed.

Importantly, the existence of flow is equivalent to the existence of focussed Pauli flow, as captured by the following theorem [34, Lemma 4.6]:

Theorem 3.7 ([34, Lemma 4.6])**.** For any open labelled graph, if a Pauli flow exists then there also exists a focussed Pauli flow.

In the case of only Pauli measurements, the conditions from the focussed flow mean that the corrector sets can only consist of X and Y measured vertices and the corrected vertex, while the odd neighbourhoods of the corrector sets can only consist of the Z and Y measured vertices and the corrected vertex.

It is easier to search for the focussed Pauli flow, as the notion of the focussed flow in the case of Pauli bases does not require partial order as captured below. The proof follows from [34, Lemma B.11].

Lemma 3.8. If $(c, \prec)$ is a focussed Pauli flow for a labelled open graph (G, I, O, λ) with only Pauli measurements, then so is $(c, \emptyset)$.

Further, we can improve theorem 3.4 to only look for X and Z labelling, getting rid of the Y measurements:

Theorem 3.9. Suppose that a labelled open graph (G, I, O, λ) has Pauli flow. Then, there exists $\lambda' \colon \overline{O} \to \{X, Z\}$ such that (G, I, O, λ') has Pauli flow.

Proof. Let (G, I, O, λ) have Pauli flow. By theorem 3.4, there is $\lambda_1 \colon \overline{O} \to \{X, Y, Z\}$ such that (G, I, O, λ_1) has Pauli flow. By theorem 3.7 and lemma 3.8, (G, I, O, λ_1) has some focussed Pauli flow $(c, \emptyset)$. If there is no u with $\lambda_1(u) = Y$, the thesis follows. Otherwise, consider such u. Define $\lambda_2 \colon \overline{O} \to \{X, Y, Z\}$ as follows. $\lambda_2(v) = \lambda_1(v)$ for $v \neq u$. By (P9), $u \in p(u) \oplus u \in Odd(p(u))$. If $u \in p(u)$, we set $\lambda_2(u) = Z$ and when $u \in Odd(p(u))$, we set $\lambda_2(u) = X$. Then (G, I, O, λ_2) has Pauli flow $(c, \prec)$, where c is the same correction function as for (G, I, O, λ_1) and $\prec$ is given by $v \prec u$ for all $v \in \overline{O} \setminus \{u\}$, i.e. u is the last vertex in this order. We verify $(c, \prec)$ indeed is the Pauli flow. Conditions (P4), (P5), (P6) hold automatically. Similarly, (P7), (P8), (P9) hold for vertices other than u. The choice for the new label of u is done by using (P9) for the old flow, thus ensuring that c works for correction of u with new label. Hence, only (P1), (P2) and (P3) remain. By construction, u is last in the order, hence if $u \in c(v)$ or $u \in Odd(c(v))$, the necessary order condition is guaranteed to hold. When $v \in c(u)$ or $v \in Odd(c(u))$, then the $\lambda_2(v) = \lambda_1(v)$ ensures that no new order requirement appears. Similarly, $\emptyset$ order worked for other pairs of vertices. Thus, (P1), (P2) and (P3) all hold. By focussing the flow and relabelling one vertex labelled with Y at a time, we can get rid of all Y measured vertices, ending the proof. □

Thanks to the above lemmata, instead of looking for λ resulting in Pauli flow $(c, \prec)$, we can just look for λ into Pauli measurements resulting in a focussed Pauli flow $(c, \emptyset)$.

Finally, we can omit cases with $I \cap O \neq \emptyset$, by reducing those to have $I \cap O = \emptyset$.

Theorem 3.10. Let (G, I, O, λ) be a labelled open graph. Then (G, I, O, λ) has Pauli flow if and only if $(G', I \setminus O, O \setminus I, \lambda)$ does, where G' is G with vertices in $I \cap O$ removed.

Proof. Suppose $v \in I \cap O$. A correction function c in Pauli flow has codomain $\mathcal{P}(\overline{I})$ and $v \in I$ so v cannot be used in any correction set. Further, as $v \in O$, v is not measured and so $c(v)$ is undefined and the partial order does not operate on v. Therefore v does not impact any of the nine flow conditions in any way. □

Combining the above simplifications, we can restrict **FlowSearch** problem to cases (G, I, O) with $I \cap O = \emptyset$, where we are looking for $\lambda \colon \overline{O} \to \{X, Z\}$ such that (G, I, O, λ) has focussed Pauli flow.

Since the removal of Z-measured vertices preserves Pauli flow (which we mention later), it is also possible to view the above problem as follows: given (G, I, O), is there an induced open subgraph with the same set of inputs and outputs, that has Pauli flow with only X labels?

3.2 Algebraic interpretation of flow

Unless specified otherwise, from now on, we only consider (G, I, O) with $I \cap O = \emptyset$. The key construction used to translate the **FlowSearch** problem into a linear algebra problem is the reduced adjacency matrix [25] (note, that in that paper the word *induced* is used in place of *reduced*).

Definition 3.11. Let (G, I, O) be an open graph. We define the **adjacency matrix** A_G over $\mathbb{F}_2$ as a $|V| \times |V|$ matrix with $(A_G)_{u,v} = 1$ when $uv \in E$ and 0 otherwise. The **reduced adjacency matrix** $A_G\,|^{\overline{O}}_{\overline{I}}$ is the $|\overline{O}| \times |\overline{I}|$ minor of the adjacency matrix of G. The minor is obtained by removing the outputs' rows and the inputs' columns.

The key property of the reduced adjacency matrix linking it to the Pauli flow is right-invertibility. In [25], a version lining XY measurements only to the right-invertibility was established. Subsequently, this theorem was extended by Miriam Backens to work for both X and XY measurements:

Theorem 3.12. Let $\mathcal{G} = (G, I, O, \lambda)$ be a labelled open graph with $\lambda(v) \in \{X, XY\}$ for all $v \in \overline{O}$. Then $\mathcal{G}$ has focussed Pauli flow if and only if there exists a directed graph $F = (V, E_F)$ satisfying the following two properties:

- Let $E'_F = \{(u, v) \in E_F \mid \lambda(u) = XY\}$, then the subgraph $F' = (V, E'_F)$ is acyclic,
- $A_G\,|^{\overline{O}}_{\overline{I}} \cdot A_F\,|^{\overline{I}}_{\overline{O}} = Id_{\overline{O}}$, where $A_F\,|^{\overline{I}}_{\overline{O}}$ is the $\overline{I} \times \overline{O}$ minor of A_F obtained by removing inputs' rows and outputs' columns.

Further, the columns of $A_F\,|^{\overline{I}}_{\overline{O}}$ encode the correction sets of the vertices in $\overline{O}$.

The proof has not yet been published [28]. Note, that in the case of X measurements only, the graph F' in the theorem 3.12 is empty and hence it is automatically acyclic. Hence, the condition simplifies to just the existence of the right inverse.

Corollary 3.13. Let $\mathcal{G} = (G, I, O, \lambda)$ be a labelled open graph with $\lambda(v) = X$ for all $v \in \overline{O}$. Then $\mathcal{G}$ has focussed Pauli flow if and only if $A_G\,|^{\overline{O}}_{\overline{I}}$ is right-invertible over $\mathbb{F}_2$. Further, the columns of a potential right inverse of $A_G\,|^{\overline{O}}_{\overline{I}}$ encode the correction sets of the vertices in $\overline{O}$.

See figure 2a for an example of using corollary 3.13 and figure B.1 in appendix B for an example of using theorem 3.12.

The Z labelled vertices can be removed and introduced without affecting flow existence, as captured by the following lemmata. In these, $G[A]$ stands for the subgraph of G induced by vertices in A.

Lemma 3.14 (Removal of Z measured vertex [34, Lemma D.6])**.** Let (G, I, O, λ) be a labelled open graph with Pauli flow and with $\lambda(v) = Z$ for some $v \in \overline{O}$. Then v can be removed without affecting flow existence. In other words, $(G[V \setminus \{v\}], I, O, \lambda|_{\overline{O} \setminus \{v\}})$ has Pauli flow.

Lemma 3.15 (Introduction of Z measured vertex [23, Proposition 4.1])**.** Let (G, I, O, λ) have Pauli flow. Then a new Z measured vertex $v \notin V$ can be added to G, with any edges from v, without affecting flow existence. In other words, any labelled open graph (G', I, O, λ') has Pauli flow, where:

$$V(G') = V \cup \{v\} \qquad G'[V] = G \qquad \lambda'|_{V \setminus O} = \lambda \qquad \lambda'(v) = Z$$

The proof for the removal appears in [34, Lemma D.6], and is a reformulation of [2, Lemma 4.7].

Now, we extend the corollary 3.13 to also capture Z measurements, by considering a slightly different matrix that we call the flow matrix.

Definition 3.16 (Flow matrix)**.** Let $\mathcal{G} = (G, I, O, \lambda)$ be a labelled open graph with $\lambda(v) \in \{X, Z\}$ for all $v \in \overline{O}$. Let G_{disc} be G with Z labelled vertices disconnected from the rest of the graph. We define the **flow matrix** $M_{\mathcal{G}}$ as the sum of the reduced adjacency matrix $A_{G_{disc}}|^{\overline{O}}_{I}$ and matrix that has 1 at the intersections of v row and v column for all v with $\lambda(v) = Z$, and 0 otherwise.

Theorem 3.17. Let $\mathcal{G} = (G, I, O, \lambda)$ be a labelled open graph with $\lambda(v) \in \{X, Z\}$ for all $v \in \overline{O}$. Then $\mathcal{G}$ has focussed Pauli flow if and only if $M_{\mathcal{G}}$ is right-invertible over $\mathbb{F}_2$.

Proof. Let G_{disc} be as in the definition 3.16. Let $G_X = (V_X, E_X)$ be G (equivalently G_{disc}) with Z-labelled vertices removed from the rest of the graph. Then, the following facts are equivalent:

1. $\mathcal{G}$ has focussed Pauli flow,
2. $\mathcal{G}_X := (G_X, I, O, \lambda_X)$ has focussed Pauli flow, where $\lambda_X = \lambda|_{\left\{v \in \overline{O} | \lambda(v) = X\right\}}$,
3. $A_{G_X}|^{V_X \setminus O}_{V_X \setminus I}$ is right-invertible,
4. $M_{\mathcal{G}_X}$ is right-invertible,
5. $M_{\mathcal{G}_{disc}}$ is right-invertible, where $\mathcal{G}_{disc} = (G_{disc}, I, O, \lambda)$,
6. $M_{\mathcal{G}}$ is right-invertible.

$(1 \Leftrightarrow 2)$: follows from lemmata 3.14 and 3.15 – the Z labelled vertices can be removed and introduced without affecting flow existence; the existences of Pauli flow and focussed Pauli flow are equivalent by theorem 3.7.

$(2 \Leftrightarrow 3)$: follows from corollary 3.13, as in G_X there are no Z labelled vertices.

$(3 \Leftrightarrow 4)$: the notions of reduced adjacency matrix and flow matrix agree in the case of only X measurements, thus $A_{G_X}|^{V_X \setminus O}_{V_X \setminus I} = M_{\mathcal{G}_X}$.

$(4 \Leftrightarrow 5)$: if there are no Z labelled vertices in G_{disc}, then $G_{disc} = G_X$ and matrices from 4 and 5 are equal. Otherwise, let $v \in \overline{O}$ be any vertex with $\lambda(v) = Z$. In the flow matrix $M_{\mathcal{G}_{disc}}$, the v row and the v column are 0 everywhere except for the intersection, as v is disconnected from other vertices by assumption. The intersection of the v row and the v column contains 1 by the construction of the flow matrix. Thus, $M_{\mathcal{G}_{disc}}$ is right-invertible if and only if its minor obtained by removing the v row and the v column is right-invertible. Performance of such removals for all Z labelled vertices results in $M_{\mathcal{G}_X}$, and thus $M_{\mathcal{G}_{disc}}$ is right-invertible if and only if $M_{\mathcal{G}_X}$ is.

$(5 \Leftrightarrow 6)$: by construction of flow matrix, $M_{\mathcal{G}_{disc}} = M_{\mathcal{G}}$ and the equivalence follows.

Thus, $1 \Leftrightarrow 6$, as required. □

The above theorem will be essential in the next subsection. See figure 2b for an example.

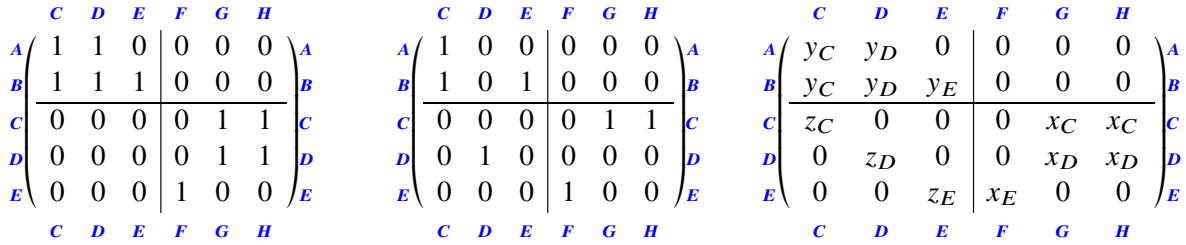
3.3 Reduction to MaxRank

Finally, we can transform **FlowSearch** into a special case of **MaxRank** problem.

Definition 3.18. Let (G, I, O) be any open graph, i.e. an input for **FlowSearch**. We define the **variable flow matrix** $M'_{G,I,O}$ as a matrix obtained as follows. (As a reminder $B := V \setminus \{I \cup O\}$.)

1. Start with the reduced adjacency matrix $A_G|^{\overline{O}}_{I}$.
2. For each $v \in B$, multiply the v row by a variable x_v and the v column by a variable y_v.
3. For each $v \in B$, set the intersection of the v row and the v column to $(1 + x_v)(1 + y_v)$.

For an example, consider figure 2c.



(a) The reduced adjacency matrix of the graph in 1a. This matrix is not right-invertible, so there is no Pauli flow.

(b) The flow matrix of the graph in 1b. This matrix is right-invertible, so there is Pauli flow. It differs from the matrix in 2a in vertex D.

(c) The variable flow matrix of the open graph in 1 (ignoring the labellings). z_v is a shorthand for $(1+x_v)(1+y_v)$ where $v \in \{C, D, E\}$.

Figure 2: Examples of the matrix interpretations of flows for labelled open graphs from 1

Theorem 3.19. The answer to the (G, I, O) instance of **FlowSearch** is $True$ if and only if there exists a valuation to $\{0,1\}$ of all variables in $M'_{G,I,O}$ resulting in a right-invertible matrix, i.e. when the answer to the instance of **MaxRank** given by the pair $(M'_{G,I,O}, |\overline{O}|)$ is "YES".

Proof. By theorem 3.9, we can consider only λ with codomain $\{X, Z\}$ and by theorem 3.10, we can can assume $I \cap O = \emptyset$.

($\Rightarrow$): let $\lambda : \overline{O} \to \{X, Z\}$ be a measurement labelling for which (G, I, O, λ) has focussed Pauli flow. For $v \in B$, send x_v and y_v to 1 when $\lambda(v) = X$ and to 0 when $\lambda(v) = Z$. Then, under this valuation, $M'_{G,I,O}$ evaluates to $M_{(G,I,O,\lambda)}$ which is right-invertible by theorem 3.17.

($\Leftarrow$): let σ be a valuation sending variables in $M'_{G,I,O}$ to $\{0,1\}$ for which $M'_{G,I,O}$ is right-invertible. Suppose $\sigma(x_v) = 0 \neq 1 = \sigma(y_v)$ for some $v \in B$. Under σ, the v row in $M'_{G,I,O}$ equals 0 everywhere, and the matrix cannot be right-invertible as it does not have maximal row rank, contradiction. Now, suppose $\sigma(x_v) = 1 \neq 0 = \sigma(y_v)$ for some $v \in B$. Under σ, the v column in $M'_{G,I,O}$ equals 0 everywhere. Changing $\sigma(y_v)$ to 1 could change the entries of the v column, otherwise leaving the matrix unchanged. Thus, such a change cannot lower the row rank and $M'_{G,I,O}$ would stay right-invertible. Hence, there exists a valuation σ' resulting in right-invertible matrix with $\sigma'(x_v) = \sigma'(y_v)$ for all $v \in B$. Now, we define $\lambda : \overline{O} \to \{X, Z\}$ as follows. For $i \in I$: $\lambda(i) = X$. For $v \in B$, set $\lambda(v) = X$ when $\sigma'(x_v) = \sigma'(y_v) = 1$ and $\lambda(v) = Z$ when $\sigma'(x_v) = \sigma'(y_v) = 0$. Then, $M_{(G,I,O,\lambda)}$ equals $M'_{G,I,O}$ under σ'. Thus, $M_{(G,I,O,\lambda)}$ is right-invertible and by theorem 3.17, (G, I, O, λ) has focussed Pauli flow, ending the proof. □

Two variables x_v and y_v for each $v \in B$ might seem unnecessary. Multiplying both the row and the column by x_v and setting the intersection to $1 + x_v$ would also work. The problem is that then each variable no longer appears in one column or row only, and theorem 2.7 would not work.

Theorem 3.1 (Repeated)**. FlowSearch** is in RP.

Proof. By theorem 3.19, answering (G, I, O) instance of **FlowSearch** is equivalent to answering instance of **MaxRank** given by $(M'_{G,I,O}, |\overline{O}|)$ over the field $\mathbb{F}_2$. The entries of $M'_{G,I,O}$ are all multi-affine – they are either 0, 1, a variable, a product of two different variables or $(x_v + 1)(y_v + 1)$ for some $v \in B$. Hence, such matrix instances satisfy all conditions of theorem 2.7, and the problem is in RP. □

An analogous procedure can also determine whether a partial labelling $\lambda : \overline{O} \hookrightarrow \{X, Z\}$ can be extended to a full labelling with flow. To do so, rather than using **MaxRank** on $M'_{G,I,O}$, we can consider

it on $M'_{G,I,O}$ with some variables evaluated to 1 or 0 depending on λ. It means, that the problem of finding λ such that (G,I,O,λ) has Pauli flow is also solvable in random polynomial time, as captured by the following corollary with initial $\lambda = \emptyset$.

Corollary 3.20. Given an open graph (G,I,O) and a partial measurement labelling λ with codomain $\{X,Z\}$, it is in RP to check if λ can be extended to a full labelling λ' such that (G,I,O,λ') has Pauli flow.

The pseudocodes of the algorithms arising from the above theorems and theorem 2.7 can be found in the appendix C. In the appendix D, we discuss the complexity and the possible implementation.

3.4 Inputs and outputs

Here, we show that given a labelled open graph with Pauli flow, it is always possible to reduce the number of outputs to match the number of inputs while preserving the existence of Pauli flow for some measurement labelling.

Theorem 3.3 (Repeated)**.** Suppose (G,I,O,λ) has Pauli flow and $|O| > |I|$. Then there exists a subset $O' \subseteq O$ such that $|O'| = |I|$ and a labelling λ' such that (G,I,O',λ') has Pauli flow.

Proof. Let $M = M_{(G,I,O,\lambda)}$ be the flow matrix of (G,I,O,λ) with Pauli flow and with $|O| > |I|$. By theorem 3.17, M is right-invertible. Hence, M has $|\overline{O}| \times |\overline{O}|$ invertible minor. Let C be the set of vertices corresponding to the columns that are not in such minor. Suppose that $o \in C \cap O$. The output o can be removed from the graph without breaking the flow existence – M without o column still contains an invertible square minor of maximal size. By lemma 3.15, we can equivalently change o to be Z labelled without breaking flow existence. Now, assume that $C \cap O = \emptyset$. In the flow matrix, the column and row of v with $\lambda(v) = Z$ are 0 except for the intersection. Therefore, the v column must be included in the maximal invertible minor and thus $v \notin C$. Hence, C is a subset of the set of X labelled vertices. Let $v \in C$. We can remove v from the graph, keeping the flow existence: removal of v column does not impact right-invertibility, and removal of v row cannot break right-invertibility either – other rows would remain linearly independent. By lemma 3.15, v can be reintroduced with the same neighbours as previously, but with $\lambda(v) = Z$. The same change can be applied to all vertices in C. Thus, if $|O| > |I|$ it is always possible to decrease the number of outputs or X measured vertices. As the second does not affect $|O| > |I|$, by repeating this process, eventually the number of outputs must decrease and match $|I|$. □

For examples, see figure E.1 in the appendix E. The procedure used in the proof above can be efficiently implemented by utilizing the basis-finding algorithm i.e. Gaussian elimination. The basis-finding procedure has another usage. In the case of X labelling only, suppose that we are given O but not I. We can then find I with $|I| = |O|$ resulting in flow, or determine that no flow exists for any set of inputs.

Lemma 3.21. Let G be a graph and $O \subseteq V$. Let $\lambda\colon \overline{O} \to \{X\}$. Then either (G,I,O,λ) does not have Pauli flow for any I, or (G,I,O,λ) has Pauli flow for some I with $|I| = |O|$.

Proof. Consider $(G,\emptyset,O,\lambda)$. Suppose that it does not have Pauli flow. Then, the reduced adjacency matrix of $(G,\emptyset,O)$ is not right-invertible. Changing inputs to a different set than $\emptyset$ corresponds to the removal of columns but not rows from the reduced adjacency matrix – it cannot make the matrix right-invertible. Hence there is no I for which (G,I,O,λ) has Pauli flow. Conversely, if $(G,\emptyset,O,\lambda)$ has Pauli flow, then we can choose $|\overline{O}| \times |\overline{O}|$ minor from the reduced adjacency matrix. The columns that are not chosen can be removed without breaking the flow, i.e. the corresponding vertices can be changed to be inputs. Note, that some output could be changed to also be an input. This process always turns $|V| - |\overline{O}| = |O|$ vertices into inputs, which ends the proof. □

Finally, again in the case of X labelling only, suppose that we are given inputs. Can we find a minimal (smallest) set of outputs resulting in Pauli flow? The answer is yes.

Lemma 3.22. Let G be a graph and $I \subseteq V$. Let $\lambda(v) = X$ for $v \in V$. Then a minimal O resulting in $(G, I, O, \lambda|_{\overline{O}})$ having Pauli flow can be efficiently found.

Proof. Let $M_\emptyset$ be the reduced adjacency matrix of $(G, I, \emptyset, \lambda)$. Let D be a set of rows forming the basis of the space given by all rows. Let C be the set of vertices whose rows are not in D. Then $O = C$ is the required minimal set – clearly $(G, I, O, \lambda|_{\overline{O}})$ has Pauli flow – in its reduced adjacency matrix M_O the rows are linearly independent, so the matrix is right-invertible. Also, O is minimal – any smaller set O' of outputs cannot result in a right-invertible reduced adjacency matrix, as such matrix would have more rows than M_O, and all of its rows would be from $M_\emptyset$. But M_O has the maximal number of linearly independent rows, as those rows form a basis of the space spanned by the rows of $M_\emptyset$. Note, that some inputs could be changed to also be outputs. □

4 Conclusions and further work

We have shown that given an open graph it is in RP to determine whether there exists a measurement labelling for which the open graph has Pauli flow. In other words, there is a random polynomial time algorithm deciding whether an open graph state can be used for any type of deterministic computation. To obtain this result, we developed an algebraic interpretation of flow for the case with two Pauli measurements X and Z, and then performed a reduction to a known problem from computational complexity. We have also shown that the algebraic interpretation can be useful when looking for the necessary properties that a set of inputs or outputs must satisfy. In particular, we showed that it is always possible to reduce the number of outputs to match the number of inputs when it is allowed to change some measurement labels to Z. Our results contribute to the general picture of what the Pauli flow structure is and which open graphs can exhibit it.

Sometimes a graph state can be prepared, but it might be difficult to check if such a state can be useful in MBQC. A possible approach can be checking other states in orbit [1]. Our result can be interpreted as answering whether an open graph can be used in any form of quantum computation, or to give some conditions on such computation that are necessary to achieve deterministic labelled open graphs.

The remainder is a discussion of possible future work.

Polynomial time When showing that some problem, **FlowSearch** in our case, is in RP, it is natural to ask whether a problem is also in P. It would be interesting to modify reduction to **MaxRank** to obtain instances for which polynomial time algorithms exist, like the instances in [18].

Other measurements The main trick we have used to show **FlowSearch** $\in$ RP, was the restriction of possible measurements to Pauli X and Z measurements for which we developed algebraic interpretation. A similar interpretation of XY planar measurements is known [25] and may be extended to allow Pauli Z measurements, but the structure is harder to work with due to the partial order requirements. Extending our results to also work for XY and Z measurements is an interesting direction for further research – X and Z measurements result in the Clifford fragment which can be classically simulated [15], while XY planar measurements are sufficient for universality even on cluster states [21]. It would also be interesting to check if our approach can be used to minimise the number of vertices that need to be Z measured to have flow. The main problem with adapting the presented methods to finding labels with

planar measurements is the order. Without verifying the order (ignoring conditions (P1), (P2), (P3)), a reduction to the **MaxRank** problem should still be possible. However, order changes everything. There might be a flow where a particular vertex has a label X or Z. Yet, measuring such vertex in any plane can break the flow.

Altering the set of edges Another problem that can be interpreted as **MaxRank** instance is finding how to change the set of edges to get an open graph with flow. For instance, when all non-outputs are X labelled, we can put a new variable in the place of 0 entries corresponding to the lack of an edge between vertices. An interesting question would be to determine how many edges must be added to get flow or how many must be removed. If such a problem is tractable, it could have practical usage, where given an open graph state one could say how far the state is from one allowing deterministic computation. The number of edges that need to be flipped corresponds to the number of CZ gates that must be applied.

Circuit extraction Our results also connect to the ZX calculus, where Pauli flow is a necessary condition for efficient circuit extraction [34]. It would be interesting to expand on this connection, for instance, by attempting the classification of small open graphs, that, when contained in ZX diagram, are guaranteed to break flow, and thus should be avoided in any form of optimization utilizing ZX. Another connection could be circuit extraction from phase-free ZH – X and Z measurements are sufficient for universality in MBQC on hypergraph states [36]. There, the errors can be multi-qubit byproducts and their fixing is done by switching measurement bases. Our method of checking the existence of measurement labelling resulting in Pauli flow could be helpful when looking for measurements that can be swapped without removing the property of determinism (i.e. Pauli flow) in the subparts without any hyperedges.

Acknowledgement

I thank Miriam Backens, Tommy McElvanney, and Korbinian Staudacher for helpful discussions. Special thanks go to my supervisor Miriam Backens, for sharing theorem 3.12 with me. I also thank anonymous reviewers for their useful comments.

References

[1] Jeremy C. Adcock, Sam Morley-Short, Axel Dahlberg & Joshua W. Silverstone (2020): *Mapping Graph State Orbits under Local Complementation*. *Quantum* 4, p. 305, doi:10.22331/q-2020-08-07-305. arXiv:1910.03969.

[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. arXiv:2003.01664.

[3] Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray Perlner, Daniel Smith-Tone, Jean-Pierre Tillich & Javier Verbel (2020): *Improvements of Algebraic Attacks for Solving the Rank Decoding and MinRank Problems*. In Shiho Moriai & Huaxiong Wang, editors: *Advances in Cryptology – ASIACRYPT* 2020, Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 507–536, doi:10.1007/978-3-030-64837-4_17.

[4] Anne Broadbent, Joseph Fitzsimons & Elham Kashefi (2009): *Universal Blind Quantum Computation*. In: 2009 50*th Annual IEEE Symposium on Foundations of Computer Science*, pp. 517–526, doi:10.1109/FOCS.2009.36.

[5] 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, doi:10.1088/1367-2630/9/8/250.

[6] Jonathan F. Buss, Gudmund S. Frandsen & Jeffrey O. Shallit (1999): *The Computational Complexity of Some Problems of Linear Algebra*. *Journal of Computer and System Sciences* 58(3), pp. 572–596, doi:10.1006/jcss.1998.1608.

[7] Bob Coecke & Ross Duncan (2011): *Interacting Quantum Observables: Categorical Algebra and Diagrammatics*. *New Journal of Physics* 13(4), p. 043016, doi:10.1088/1367-2630/13/4/043016. arXiv:0906.4725.

[8] Vincent Danos & Elham Kashefi (2006): *Determinism in the One-Way Model*. *Physical Review A* 74(5), p. 052310, doi:10.1103/PhysRevA.74.052310.

[9] Niel de Beaudrap (2007): *A Complete Algorithm to Find Flows in the One-Way Measurement Model*, doi:10.48550/arXiv.quant-ph/0603072. arXiv:quant-ph/0603072.

[10] Niel de Beaudrap (2008): *Finding Flows in the One-Way Measurement Model*. *Physical Review A* 77(2), p. 022328, doi:10.1103/PhysRevA.77.022328.

[11] Niel de Beaudrap, Aleks Kissinger & John van de Wetering (2022): *Circuit Extraction for ZX-Diagrams Can Be* #P*-Hard*. In: 49*th International Colloquium on Automata, Languages, and Programming* (*ICALP* 2022), *LIPIcs* 229, pp. 119:1–119:19, doi:10.4230/LIPIcs.ICALP.2022.119.

[12] Richard A. Demillo & Richard J. Lipton (1978): *A Probabilistic Remark on Algebraic Program Testing*. *Information Processing Letters* 7(4), pp. 193–195, doi:10.1016/0020-0190(78)90067-4.

[13] 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. arXiv:1902.03178.

[14] Sergey B. Gashkov & Igor S. Sergeev (2013): *Complexity of Computation in Finite Fields*. *Journal of Mathematical Sciences* 191(5), pp. 661–685, doi:10.1007/s10958-013-1350-5.

[15] Daniel Gottesman (1998): *The Heisenberg Representation of Quantum Computers*, doi:10.48550/arXiv.quant-ph/9807006. arXiv:quant-ph/9807006.

[16] Nicholas J. A. Harvey, David R. Karger & Sergey Yekhanin (2006): *The Complexity of Matrix Completion*. In: *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm - SODA '06*, ACM Press, Miami, Florida, pp. 1103–1111, doi:10.1145/1109557.1109679.

[17] Matt Hostetter (2020): *Galois: A performant NumPy extension for Galois fields*. https://github.com/mhostetter/galois. (accessed February 2024).

[18] Gábor Ivanyos, Marek Karpinski & Nitin Saxena (2010): *Deterministic Polynomial Time Algorithms for Matrix Completion Problems*. *SIAM Journal on Computing*, doi:10.1137/090781231.

[19] Frank Luebeck (2021): *Conway Polynomials for Finite Fields*. https://www.math.rwth-aachen.de/~Frank.Luebeck/data/ConwayPol/index.html. (accessed February 2024).

[20] Meena Mahajan & Jayalal M. N. Sarma (2010): *On the Complexity of Matrix Rank and Rigidity*. *Theory of Computing Systems* 46(1), pp. 9–26, doi:10.1007/s00224-008-9136-8.

[21] Atul Mantri, Tommaso F. Demarie & Joseph F. Fitzsimons (2017): *Universality of Quantum Computation with Cluster States and (X, Y)-Plane Measurements*. *Scientific Reports* 7(1), p. 42861, doi:10.1038/srep42861.

[22] Damian Markham & Elham Kashefi (2014): *Entanglement, Flow and Classical Simulatability in Measurement Based Quantum Computation*. In Franck van Breugel, Elham Kashefi, Catuscia Palamidessi & Jan Rutten, editors: *Horizons of the Mind. A Tribute to Prakash Panangaden*: *Essays Dedicated to Prakash Panangaden on the Occasion of His* 60*th Birthday*, Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 427–453, doi:10.1007/978-3-319-06880-0_22.

[23] Tommy McElvanney & Miriam Backens (2023): *Complete Flow-Preserving Rewrite Rules for MBQC Patterns with Pauli Measurements*. *Electronic Proceedings in Theoretical Computer Science* 394, pp. 66–82, doi:10.4204/EPTCS.394.5.

[24] Tommy McElvanney & Miriam Backens (2023): *Flow-Preserving ZX-calculus Rewrite Rules for Optimisation and Obfuscation*. *Electronic Proceedings in Theoretical Computer Science* 384, pp. 203–219, doi:10.4204/EPTCS.384.12. arXiv:2304.08166.

[25] Mehdi Mhalla, Mio Murao, Simon Perdrix, Masato Someya & Peter S. Turner (2014): *Which Graph States Are Useful for Quantum Information Processing?* In Dave Bacon, Miguel Martin-Delgado & Martin Roetteler, editors: *Theory of Quantum Computation, Communication, and Cryptography*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 174–187, doi:10.1007/978-3-642-54429-3_12.

[26] Mehdi Mhalla & Simon Perdrix (2008): *Finding Optimal Flows Efficiently*. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir & Igor Walukiewicz, editors: *Automata, Languages and Programming*, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, pp. 857–868, doi:10.1007/978-3-540-70575-8_70.

[27] Mehdi Mhalla, Simon Perdrix & Luc Sanselme (2022): *Characterising Determinism in MBQCs Involving Pauli Measurements*, doi:10.48550/arXiv.2207.09368. arXiv:2207.09368.

[28] Piotr Mitosek & Miriam Backens: *Unpublished Upcoming Paper*.

[29] Øystein Ore (1921): *Über Höhere Kongruenzen*. Norsk Matematisk Forenings Skrifter, Grøndahl.

[30] Robert Raussendorf & Hans J. Briegel (2001): *A One-Way Quantum Computer*. *Physical Review Letters* 86(22), pp. 5188–5191, doi:10.1103/PhysRevLett.86.5188.

[31] Robert Raussendorf, Daniel E. Browne & Hans J. Briegel (2002): *The One-Way Quantum Computer - a Non-Network Model of Quantum Computation*. *Journal of Modern Optics* 49(8), pp. 1299–1306, doi:10.1080/09500340110107487. arXiv:quant-ph/0108118.

[32] Robert Raussendorf, Daniel E. Browne & Hans J. Briegel (2003): *Measurement-Based Quantum Computation on Cluster States*. *Physical Review A* 68(2), p. 022312, doi:10.1103/PhysRevA.68.022312.

[33] Jacob T. Schwartz (1980): *Fast Probabilistic Algorithms for Verification of Polynomial Identities*. *Journal of the ACM* 27(4), pp. 701–717, doi:10.1145/322217.322225.

[34] Will Simmons (2021): *Relating Measurement Patterns to Circuits via Pauli Flow*. *Electronic Proceedings in Theoretical Computer Science* 343, pp. 50–101, doi:10.4204/EPTCS.343.4. arXiv:2109.05654.

[35] Korbinian Staudacher, Tobias Guggemos, Sophia Grundner-Culemann & Wolfgang Gehrke (2023): *Reducing 2-QuBit Gate Count for ZX-Calculus Based Quantum Circuit Optimization*. *EPTCS* 394, pp. 29–45, doi:10.4204/EPTCS.394.3.

[36] Yuki Takeuchi, Tomoyuki Morimae & Masahito Hayashi (2019): *Quantum Computational Universality of Hypergraph States with Pauli-X and Z Basis Measurements*. *Scientific Reports* 9(1), p. 13585, doi:10.1038/s41598-019-49968-3.

[37] John van de Wetering (2020): *ZX-calculus for the Working Quantum Computer Scientist*, doi:10.48550/arXiv.2012.13966. arXiv:2012.13966.

[38] Richard Zippel (1979): *Probabilistic Algorithms for Sparse Polynomials*. In Edward W. Ng, editor: *Symbolic and Algebraic Computation*, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, pp. 216–226, doi:10.1007/3-540-09519-5_73.

A More about MaxRank

In order to prove theorem 2.7, we need the following lemma.

Lemma A.1. A multi-affine polynomial is 0 over a finite field $\mathbb{F}_s$ if and only if it is 0 over $\mathbb{F}_{s^k}$ for any $k \in \mathbb{Z}_+$.

Proof. This follows from [6, Lemma 25 and Corollary 26]. □

Because of the above, instead of testing whether a multi-affine polynomial is 0 over $\mathbb{F}_s$, we can test whether it is 0 over some large field extension. In particular, the extension can be taken sufficiently large to ensure that the Schwartz-Zippel lemma [29, 33, 12, 38] applies (adapted to $\mathbb{F}_{s^k}$ only):

Theorem A.2. Let $P \in \mathbb{F}_{s^k}[x_1, \ldots, x_n]$ be a non-zero polynomial of total degree d over $\mathbb{F}_{s^k}$. Let $a_1, \ldots, a_n \in \mathbb{F}_{s^k}$ be chosen at random uniformly. Then:

$$\Pr[P(a_1, \ldots, a_n) = 0] \leq \frac{d}{s^k}.$$

We can now prove theorem 2.7.

Proof of theorem 2.7. Let M, r be a matrix and an integer forming an input to **MaxRank** satisfying the conditions from the theorem statement. If $r > \min(m,n)$ or $r < 0$, the answer is "NO" and can be returned immediately, so assume $0 \leq r \leq \min(m,n)$. Consider any $r \times r$ minor M' of M. We show $\det M'$ is multi-affine: consider any variable x in M'. Then, x appears in at most one row or one column of M'. By performing Laplace expansion on M' in such row (column), we find that $\det M'$ is a sum of determinants of $(r-1) \times (r-1)$ minors of M' that do not contain x multiplied by elements of the row (column) used for the expansion that itself may contain x only in degree 1 due to multi-affinity assumption about matrix entries. Therefore, x appears in degree at most 1 in $\det M'$ and the same for all other variables of $\det M'$, so the determinant is multi-affine. Therefore, the method from [6, Theorem 28] applies. We present a modified version for clarity.

Consider the following procedure. Let $p \leq \frac{1}{2}$ be the desired error probability. It is sufficient to consider $p = \frac{1}{2}$, but we present also how to achieve arbitrarily small error probability. Given M, r, p, let k be such that $\mathbb{F}_{s^k}$ has at least $\frac{t}{p}$ elements, i.e. $k = \left\lceil \log_s \frac{t}{p} \right\rceil$. Let $a_1, \ldots, a_t$ be a randomly chosen valuation of $x_1, \ldots, x_t$ from $\mathbb{F}_{s^k}$. Let $r_a = \operatorname{rank} M(a_1, \ldots, a_t)$, which can be found by Gaussian elimination i.e. in polynomial time. The computations over finite field are possible in time polynomial in $\log_2 s$ and k [14]. Return "YES" if and only if $r_a \geq r$. We show, that the following procedure shows RP containment.

Suppose, that the actual answer to the instance is "YES". Then, under some valuation, M has rank at least r. Under such valuation, M must have $r \times r$ reversible minor. Let M' be such minor. By the previous part, $\det M'$ is multi-affine. Let d be the total degree of $\det M'$. Then, $d \leq t$ again by multi-affinity. By lemma A.1, M' is 0 over $\mathbb{F}_s$ if and only if it is 0 over $\mathbb{F}_{s^k}$. Combining everything with the Schwartz-Zippel theorem A.2, we get that:

$$\Pr[\det M'(a_1, \ldots, a_t) =_{\mathbb{F}_{s^k}} 0] \leq \frac{d}{s^k} \leq \frac{t}{s^k} \leq \frac{t}{t/p} = p$$

The same holds for all $r \times r$ minors of M that are invertible under some valuation. Hence, with error probability at most p, a random valuation from $\mathbb{F}_{s^k}$ results in a non-root of some $r \times r$ minor's determinant of M in which case rank of the minor under such valuation is r and so rank M is at least r, i.e. the procedure above would find $r_a \geq r$ and return "YES". Hence, the procedure described above returns the correct answer with probability at least $1 - p \geq \frac{1}{2}$.

Now suppose, that the answer to the instance is "NO". Then, all $r \times r$ minors of M must have determinants equal 0 over $\mathbb{F}_s$. By multi-affinity, they are also equal 0 over $\mathbb{F}_{s^k}$. Hence, a random valuation $a_1, \ldots, a_t$ always results in $\operatorname{rank} M(a_1, \ldots, a_t) \leq k$. Hence, the procedure described above returns "NO" with probability 1, ending the proof of containment in RP. □

Example A.3. Consider the following matrix over $\mathbb{F}_2$:

$$M = \begin{pmatrix} x_1 & 0 & 1 & 0 \\ 0 & x_2x_3 & 0 & 0 \\ 1 & 0 & x_1 & 0 \\ x_1 & x_2 & x_3 & 0 \end{pmatrix}$$

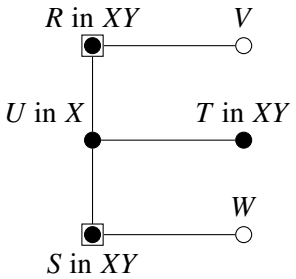
Setting $x_1 = 0$ and $x_2, x_3 = 1$ results in M having rank 3. However, there is no valuation resulting in the matrix having rank 4, as the fourth column contains 0s only. Hence, the answer to instances $(M,1), (M,2), (M,3)$ of **MaxRank** is "YES", but the answer to $(M,4)$ is "NO".

Some closely related problems also defined in [6] include **MinRank**, **Sing**, and **NonSing**. **MinRank** takes the same inputs as **MaxRank** and asks whether a rank $\leq r$ can be achieved. **Sing** and **NonSing** take a square matrix and ask whether the matrix can be made singular and non-singular respectively.

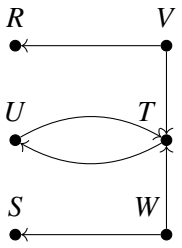
In general, these problems are hard or sometimes unsolvable. For instance, **MinRank** is undecidable when $R = \mathbb{Z}, E = S = \{0,1\}$ [6]. The problems are very natural and often appear when working on any linear algebra problems. **Sing** is useful in cryptography due to its hardness (for example, see [3]). It is also interesting from a complexity perspective (for example, see [20]).

We only work with **MaxRank** over finite fields (in fact, later we only consider $\mathbb{F}_{2^k}$). A variant where each variable can appear at most once is in P. A slightly more general version where a variable can appear in at most one row or one column but an unlimited number of times is also in P [18]. Note, that this result is not stronger than the presented theorem 2.7, as the entries of the matrix there cannot include products of variables. When each variable can appear at most twice in the matrix, but not necessarily in one row or one column, the problem already becomes NP-complete [16].

B Reduced adjacency matrix invertibility for X and XY measurements

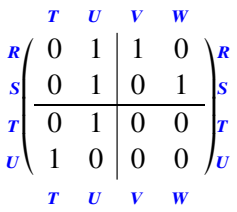


(a) A labelled open graph with two inputs and two outputs containing Pauli flow. Vertex U is X labelled and all other non-outputs are XY labelled.

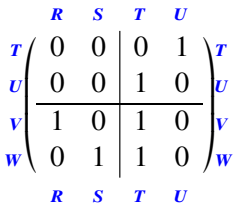


(b) A directed graph F for the labelled open graph from B.1a, obtained from theorem 3.12. It contains a cycle. The corresponding F' contains only one edge TU and is acyclic.

Figure B.1: An example of a labelled open graph with both X and XY measurements and explanation of theorem 3.12 acting on it.



(c) Reduced adjacency matrix of the labelled open graph from B.1a.



(d) The inverse of matrix in B.1c, i.e. the matrix $A_F\,|\frac{I}{O}$ where F is as in B.1b.

Figure B.1 (continued): An example of a labelled open graph with both X and XY measurements and explanation of theorem 3.12 acting on it.

C Pseudocode for algorithms

In the pseudocode below, we do not explicitly construct an instance of matrix used for **MaxRank** problem, as, in practice, it might be difficult to explicitly construct a matrix with variables that can be substituted with values from large field extensions of $\mathbb{F}_2$. Instead, we only construct the matrix under some valuation.

Checks if a partial X,Z labelling can be extended so that (G,I,O,λ) has Pauli flow. The error probability must be bounded above by p.

procedure FlowSearchAux(G,I,O,λ,p)
  $M \leftarrow A_G\,|\frac{O}{I}$ ▷ Construction of reduced adjacency matrix
  $Vars \leftarrow \emptyset$ ▷ Initialize set of variables
  for $v \in B$ ▷ Detecting unlabelled vertices
    if $\lambda(v)$ is defined
      if $\lambda(v) == Z$ ▷ Updating the row and the column of Z labelled vertex
        multiply v row of M by 0
        multiply v column of M by 0
        set the intersection of v row and v column of M to 1
    else
      $Vars \leftarrow Vars \cup \{x_v, y_v\}$
  $k \leftarrow \left\lceil \log_2 \frac{|Vars|}{p} \right\rceil$ ▷ Minimal k such that $\mathbb{F}_{2^k}$ has at least $\frac{|Vars|}{p}$ elements and the error probability is below p.
  Randomly sample $\sigma : Vars \to \mathbb{F}_{2^k}$
  for $v \in B$ ▷ Construction of $M'_{G,I,O}$ under valuation σ, computations are done in $\mathbb{F}_{2^k}$
    multiply v row of M by $\sigma(x_v)$
    multiply v column of M by $\sigma(y_v)$
    set the intersection of v row and v column to $(\sigma(x_v)+1)(\sigma(y_v)+1)$
  Gaussian eliminate M
  return (rowrank $M == |\overline{O}|$)

Main algorithm, returns $True$ if (G,I,O,λ) has Pauli flow for some λ. The error probability must

be bounded above by p.

procedure FlowSearch(G, I, O, p)
    return FlowSearchAux($G, I, O, \emptyset, p$)

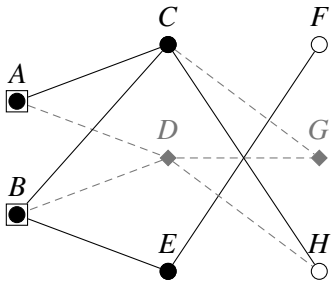
Finds λ such that (G, I, O, λ) has Pauli flow. Initially, checks whether any such λ exists, up to error probability p.

procedure FindLabelling(G, I, O, p)
    if not FlowSearch(G, I, O, p)
        return "NO λ EXISTS"
    $\lambda \leftarrow \emptyset$ ▷ Initialization of λ
    for $v \in I$
        $\lambda(v) \leftarrow X$ ▷ Inputs must be X labelled
    for $v \in B$
        $confirm_v \leftarrow False$
        $current_v \leftarrow X$ ▷ Initially attempt X label
        while not $confirm_v$ ▷ Alternate X and Z labels until one works
            $\lambda(v) \leftarrow current_v$
            if FlowSearchAux(G, I, O, λ, p) ▷ Note, that the error probability could be increased at the cost of possibly more tries being required.
                $confirm_v \leftarrow True$
            else
                if $current_v == X$
                    $current_v \leftarrow Z$
                else
                    $current_v \leftarrow X$
    return λ

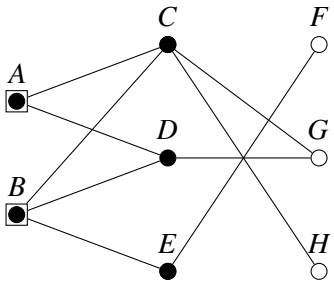
D Complexity of algorithms

The most memory-expensive part of the algorithms is the creation of $M'_{G,I,O}$ under some valuation from $\mathbb{F}_{2^k}$ where k depends on the desired error probability p and the size of the input graph. Since $k = \left\lceil \log_2 \frac{|Vars|}{p} \right\rceil$ and $|Vars| \leq 2 \cdot |B|$, we get that $k \in O\left(\log_2 \frac{|B|}{p}\right)$. The elements of such fields can be represented using $O(k)$ long vectors over $\mathbb{F}_2$ with the time complexity of basic arithmetic operations on such field bounded above by $O(k^2)$ [14]. Therefore, the memory requirement can be bounded above by $O\left(|\overline{O}| \times |\overline{I}| \times \log_2 \frac{|B|}{p}\right) \in O\left(n^2 \log_2 \frac{n}{p}\right)$ where $n = |V|$. The most time-expensive part of the algorithms are the Gaussian eliminations. Each Gaussian elimination requires $O\left(n^3\right)$ basic operations in $\mathbb{F}_{2^k}$. Thus, a (not very efficient) upper bound for the time complexity is $O\left(n^3 \log_2^2 \frac{n}{p}\right)$ for the decision variant and expected $O\left(n^4 \log_2^2 \frac{n}{p}\right)$ for the actual finding of the labelling resulting in a Pauli flow. Many programming languages offer packages for efficient computation in finite fields. For instance, in Python one can use Galois package [17] which works very well for $\mathbb{F}_{2^k}$ with k such that the precomputed Conway polynomial [19] is known, for example, all $1 \leq k \leq 91$. Such values of k are sufficient for all reasonable computations, as the error can be dropped below $\frac{1}{2^{50}}$. At that point, it is more likely for a random cosmic beam to corrupt the computation than to get an error due to the probabilistic nature of the algorithms.

E Figure for reducing the number of outputs



(a) The open graph from 1b with flow matrix from 2b. Columns given by vertices C, D, E, F and H form the basis. Thus, the output G can be removed (equivalently: changed to be Z labelled).



(b) An example of a labelled open graph with Pauli flow in which no output can be immediately changed to be Z measured. However, changing D to be Z measured makes it possible to also change output G to be Z measured.

$$\begin{array}{c} \begin{array}{cccccc} C & D & E & F & G & H \end{array} \\ \left(\begin{array}{cccc|cc} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right) \begin{array}{c} A \\ B \\ C \\ D \\ E \end{array} \\ \begin{array}{cccccc} C & D & E & F & G & H \end{array} \end{array}$$

(c) Flow matrix of the open graph in E.1b. Columns given by vertices C, E, F, G, H form the basis. Thus, the vertex D can be changed to be Z labelled.

$$\begin{array}{c} \begin{array}{cccccc} C & D & E & F & G & H \end{array} \\ \left(\begin{array}{cccc|cc} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right) \begin{array}{c} A \\ B \\ C \\ D \\ E \end{array} \\ \begin{array}{cccccc} C & D & E & F & G & H \end{array} \end{array}$$

(d) The flow matrix after switching vertex D from E.1b to be Z labelled. Columns given by vertices C, D, E, F, H form a basis, so output G can be removed (changed to be Z labelled).

Figure E.1: Examples of labelled open graphs with more outputs than inputs, and how the number of outputs can be reduced to match the number of inputs.