




# Resource Optimisation of Coherently Controlled Quantum Computations with the PBS-Calculus

Alexandre Clément   

Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

Simon Perdrix   

Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

---

## Abstract

---

Coherent control of quantum computations can be used to improve some quantum protocols and algorithms. For instance, the complexity of implementing the permutation of some given unitary transformations can be strictly decreased by allowing coherent control, rather than using the standard quantum circuit model. In this paper, we address the problem of optimising the resources of coherently controlled quantum computations. We refine the PBS-calculus, a graphical language for coherent control which is inspired by quantum optics. In order to obtain a more resource-sensitive language, it manipulates abstract gates – that can be interpreted as queries to an oracle – and more importantly, it avoids the representation of useless wires by allowing unsaturated polarising beam splitters. Technically the language forms a coloured PROP. The language is equipped with an equational theory that we show to be sound, complete, and minimal.

Regarding resource optimisation, we introduce an efficient procedure to minimise the number of oracle queries of a given diagram. We also consider the problem of minimising both the number of oracle queries and the number of polarising beam splitters. We show that this optimisation problem is NP-hard in general, but introduce an efficient heuristic that produces optimal diagrams when at most one query to each oracle is required.

**2012 ACM Subject Classification** Theory of computation → Quantum computation theory; Theory of computation → Axiomatic semantics; Theory of computation → Categorical semantics

**Keywords and phrases** Quantum computing, Graphical language, Coherent control, Completeness, Resource optimisation, NP-hardness

**Digital Object Identifier** 10.4230/LIPIcs.MFCS.2022.36

**Related Version** *Preprint Version*: <https://arxiv.org/abs/2202.05260> [13]

**Funding** This work is funded by ANR-17-CE25-0009 SoftQPro, ANR-17-CE24-0035 VanQuTe, PIA-GDN/Quantex, and LUE/UOQ, the PEPR integrated project EPiQ ANR-22-PETQ-0007 part of Plan France 2030, and by “*Investissements d’avenir*” (ANR-15-IDEX-02) program of the French National Research Agency; the European Project NExt ApplicationS of Quantum Computing (NEASQC), funded by Horizon 2020 Program inside the call H2020-FETFLAG-2020-01 (Grant Agreement 951821), and the HPCQS European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101018180.

## 1 Introduction

Most models of quantum computation (like quantum circuits) and most quantum programming languages are based on the *quantum data/classical control* paradigm. In other words, based on a set of quantum primitives (e.g. unitary transformations, quantum measurements), the way these primitives are applied on a register of qubits is either fixed or classically controlled.

However, quantum mechanics offers more general control of operations: for instance in quantum optics it is easy to control the trajectory of a system, like a photon, based on its polarisation using a *polarising beam splitter*. One can then position distinct quantum primit-



© Alexandre Clément and Simon Perdrix;  
licensed under Creative Commons License CC-BY 4.0

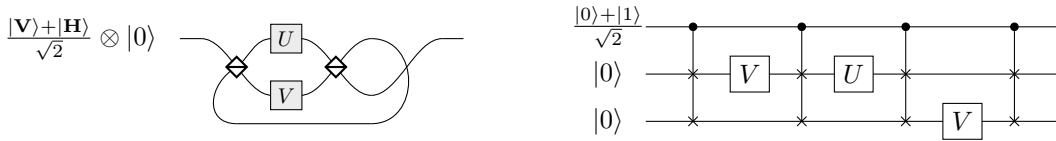
47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022).

Editors: Stefan Szeider, Robert Ganian, and Alexandra Silva; Article No. 36; pp. 36:1–36:15

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



■ **Figure 1** [Left] Coherently controlled quantum computation for solving the commuting problem. Only two queries are used: one query to  $U$  and one query to  $V$ . [Right] Optimal circuit for solving the commuting problem, where the 3-qubit gate is a control-swap. Notice that three queries are necessary in the quantum circuit model.

ives on the distinct trajectories. Since the polarisation of a photon can be in superposition, it achieves some form of quantum control, called coherent control: the quantum primitives are applied in superposition depending on the state of another quantum system. Coherent control is not only a subject of interest for foundations of quantum mechanics [21, 26, 33], it also leads to advantages in solving computational problems [18, 4, 14, 27] and in designing more efficient protocols [19, 9, 1, 17, 20].

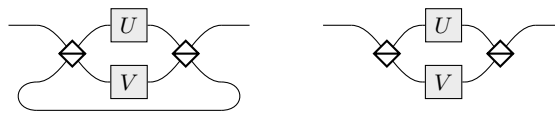
Indeed, some problems can be solved more efficiently by using coherent control rather than the usual quantum circuits. This separation has been proved in a multi-oracle model where the measure of complexity is the number of queries to (a single or several distinct) oracles, which are generally unitary maps. The simplest example is the following problem [9]: given two oracles  $U$  and  $V$  with the promise that they are either commuting or anti-commuting, decide whether  $U$  and  $V$  are commuting or not. This problem can be solved using the so-called quantum switch [10] which can be implemented using only two queries by means of coherent control, whereas solving this problem requires at least 3 queries (e.g. two queries to  $U$  and one query to  $V$ ) in the quantum circuit model (see Figure 1).

In this paper, we address the problem of optimising resources of coherently controlled quantum computations. To do so, we first refine the framework of the PBS-calculus – a graphical language for coherently controlled quantum computation – to make it more resource-sensitive. Then, we consider the problem of optimising the number of queries, and also the number of polarising beam splitters, of a given coherently controlled quantum computation, described as a PBS-diagram.


**PBS-calculus.** The PBS-calculus is a graphical language that has been introduced [12] to represent and reason about quantum computations involving coherent control of quantum operations. Inspired by quantum optics [11], the polarising beam splitter (PBS for short), denoted  $\bowtie$  is at the heart of the language: when a photon enters the PBS, say from the top left, it is reflected (and hence outputted on the top right) if its polarisation is vertical; or transmitted (and hence outputted on the bottom right) if its polarisation is horizontal. If the polarisation is a superposition of vertical and horizontal, the photon is outputted in a superposition of two positions. As a consequence, the trajectory of a particle, say a photon, will depend on its polarisation. The second main ingredient of the PBS-calculus are the gates, denoted  $\boxed{U}$  which applies some transformation  $U$  on a data register. Notice that the gates never act on the polarisation of the particle.

PBS-diagrams, which form a traced symmetric monoidal category (more precisely a traced prop [24]), are equipped with an equational theory that allows one to transform a diagram. The equational theory has been proved to be sound, complete, and minimal [12].

Notice that a PBS-diagram may have some useless wires, like in the example of the “half quantum switch”, see Figure 2 (left). We refine the PBS-calculus in order to allow one to remove these useless wires, leading to unsaturated PBS (or 3-leg PBS) like  $\overline{\bowtie}$  or  $\overline{\bowtie}$ .



■ **Figure 2** A coherent control of  $U$  and  $V$ , also called a half quantum switch: when the initial polarisation is vertical ( $\mathbf{V}$ ),  $U$  is applied on the data register, when the polarisation is horizontal ( $\mathbf{H}$ ),  $V$  is applied. Whatever the polarisation is, the particle always goes out of the top port of the second beam splitter. On the right-hand side the diagram is made of beam splitters with a missing leg, whereas on the left-hand side standard beam splitters are used, and a useless trace is added.

To avoid ill-formed diagrams like , a typing discipline is necessary. To this end, we use the framework of coloured props: each wire has 3 possible colours: black, red and blue which can be interpreted as follows: a photon going through a blue (resp. red) wire must have a horizontal (resp. vertical) polarisation.

The introduction of unsaturated polarising beam splitters requires to revisit the equational theory of the PBS-calculus. The heart of the refined equational theory is the axiomatisation of the 3-leg polarising beam splitters, together with some additional equations which govern how 4-leg polarising beam splitters can be decomposed into 3-leg ones. To show the completeness of the refined equational theory, we introduce normal forms and show that any diagram can be put in normal form. Finally, we also show the minimality of the equational theory by proving that none of the equations can be derived from the other ones.

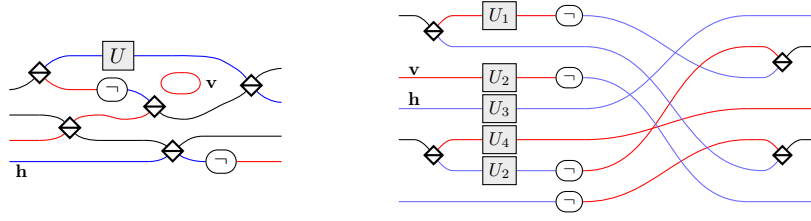
**Resource Optimisation.** The PBS-calculus, thanks to its refined equational theory, provides a way to detect and remove dead-code in a diagram. We exploit this property to address the crucial question of resource optimisation. We introduce a specific form of diagrams that minimises the number of gates, more precisely the number of queries to oracles, with an appropriate modelisation of oracles. We provide an efficient procedure to transform any diagram into this specific form. We then focus on the problem of optimising both the number of queries and the number of polarising beam splitters. We refine the previous procedure, leading to an efficient heuristic. We show that the produced diagrams are optimal when every oracle is queried at most once, but might not be optimal in general. We actually show that the general optimisation problem is NP-hard using a reduction from the *maximum Eulerian cycle decomposition problem* [8].

**Related work.** Several languages have been designed to represent coherently controlled quantum computation: some of them are extensions of quantum circuits, and other diagrammatic languages [30, 5, 31, 29]; others are based on abstract programming languages [2, 32, 16, 15, 6]. While there are numerous works on resource-optimisation of quantum computation, in particular for quantum circuits [23, 3, 25], there was, up to our knowledge, no procedure for resource optimisation of coherently controlled quantum computation.

All omitted proofs can be found in the preprint version of the paper [13].

## 2 Coloured PBS-diagrams

We use the formalism of traced coloured props (i.e. small traced symmetric strict monoidal categories whose objects are freely spanned by the elements of a set of colours) to represent coherently controlled quantum computations. We are going to use the “colours”  $\mathbf{v}$ ,  $\mathbf{h}$ ,  $\top$ , to denote respectively vertical, horizontal or possibly both polarisations.



■ **Figure 3** (Left) An example of diagram of type  $\top \oplus \top \oplus \mathbf{v} \oplus \mathbf{h} \rightarrow \top \oplus \mathbf{h} \oplus \top \oplus \mathbf{v}$ . (Right) An example of a diagram of type  $\top \oplus \mathbf{v} \oplus \mathbf{h} \oplus \top \oplus \mathbf{h} \rightarrow \mathbf{h} \oplus \top \oplus \mathbf{v} \oplus \top \oplus \mathbf{h}$ , in a particular form that we will call *normal form* (see Definition 15).

► **Definition 1.** Given a monoid  $M$ , let  $\mathbf{Diag}^M$  be the traced coloured prop with colours  $\{\mathbf{v}, \mathbf{h}, \top\}$  freely generated by the following generators, for any  $U \in M$ :

	$\top \oplus \top \rightarrow \top \oplus \top$		$\top \oplus \mathbf{v} \rightarrow \mathbf{v} \oplus \top$		$\top \rightarrow \mathbf{h} \oplus \mathbf{v}$
	$\mathbf{h} \oplus \top \rightarrow \mathbf{h} \oplus \top$		$\mathbf{v} \oplus \top \rightarrow \top \oplus \mathbf{v}$		$\mathbf{v} \oplus \mathbf{h} \rightarrow \top$
	$\top \oplus \mathbf{h} \rightarrow \top \oplus \mathbf{h}$		$\top \rightarrow \mathbf{v} \oplus \mathbf{h}$		$\mathbf{h} \oplus \mathbf{v} \rightarrow \top$
	$\top \rightarrow \top$		$\mathbf{v} \rightarrow \mathbf{h}$		$\mathbf{h} \rightarrow \mathbf{v}$
	$\top \rightarrow \top$		$\mathbf{v} \rightarrow \mathbf{v}$		$\mathbf{h} \rightarrow \mathbf{h}$

The morphisms of  $\mathbf{Diag}^M$  are called  $M$ -diagrams or simply diagrams when  $M$  is irrelevant or clear from the context. Intuitively, the diagrams are inductively obtained by composition of the generators from Definition 1 using the sequential composition  $D_2 \circ D_1$ , the parallel composition  $D_3 \oplus D_4$ , and the trace  $Tr_a(D)$  which are respectively depicted as follows:

follows:  $\begin{array}{c} \vdots \\ \boxed{D_1} \\ \vdots \end{array} \begin{array}{c} \vdots \\ \boxed{D_2} \\ \vdots \end{array} \begin{array}{c} \vdots \\ \boxed{D_3} \\ \vdots \\ \vdots \\ \boxed{D_4} \\ \vdots \end{array} \begin{array}{c} \vdots \\ \boxed{D} \\ \vdots \\ \text{---} \\ \vdots \end{array}$ . Notice that these compositions should type-check,

i.e.  $D_1 : a \rightarrow b$ ,  $D_2 : b \rightarrow c$  and  $D : a \oplus d \rightarrow b \oplus d$  with  $d \in \{\top, \mathbf{v}, \mathbf{h}\}$ . The axioms of the traced coloured prop guarantee that the diagrams are defined up to deformation: two diagrams whose graphical representations are isomorphic are equal.

Regarding notations, we use actual colours for wires: blue for  $\mathbf{h}$ -wires, red for  $\mathbf{v}$ -wires, and black for  $\top$ -wires. We also add labels on the wires, which are omitted when clear from the context, so that there is no loss of information in the case of a colour-blind reader or black and white printing. Two examples of diagrams are given in Figure 3.

Unless specified, the unit of  $M$  is denoted  $I$  and its composition is  $\cdot$  which will be generally omitted ( $VU$  rather than  $V \cdot U$ ). The main two examples of monoids we consider in the rest of the paper are:

- The monoid  $\mathcal{U}(\mathcal{H})$  of isometries of a Hilbert space  $\mathcal{H}$  with the usual composition. When  $\mathcal{H}$  is of finite dimension, the elements of  $\mathcal{U}(\mathcal{H})$  are unitary maps. With a slight abuse of notations, the corresponding traced coloured prop of diagrams is denoted  $\mathbf{Diag}^{\mathcal{H}}$ .
- The free monoid  $\mathcal{G}^*$  on some set  $\mathcal{G}$ . The gates, when the monoid is freely generated, can be interpreted as queries to oracles (each element of  $\mathcal{G}$  corresponds to an oracle): the gates implement a priori arbitrary operations with no particular structures. We use the term *abstract diagram* when the underlying monoid is freely generated. Notice that the free monoid case can also be seen as an extension of the *bare diagrams* [7] whose gates are labelled with *names*.

### 3 Semantics

The input of a diagram is a single particle, which has a polarisation, a position and a data register. A basis state for the polarisation is either vertical or horizontal, and a basis state for the position is an integer which corresponds to the wire on which the particle is located. The type of a diagram restricts the possible input/output configurations: if  $D : \mathbf{v} \oplus \top \rightarrow \mathbf{h} \oplus \mathbf{h} \oplus \mathbf{v}$  then the possible input (resp. output) configurations are the following polarisation-position pairs:  $\{(\mathbf{V}, 0), (\mathbf{V}, 1), (\mathbf{H}, 1)\}$  (resp.  $\{(\mathbf{H}, 0), (\mathbf{H}, 1), (\mathbf{V}, 2)\}$ ). More generally for any object  $a$ , let  $[a]$  be the set of possible configurations, and  $|a|$  be its size, inductively defined as follows:  $|I| = 0$ ,  $|a \oplus \top| = |a \oplus \mathbf{v}| = |a \oplus \mathbf{h}| = |a| + 1$ , and  $[I] = \emptyset$ ,  $[a \oplus \mathbf{v}] = [a] \cup \{(\mathbf{V}, |a|)\}$ ,  $[a \oplus \mathbf{h}] = [a] \cup \{(\mathbf{H}, |a|)\}$  and  $[a \oplus \top] = [a] \cup \{(\mathbf{V}, |a|), (\mathbf{H}, |a|)\}$ .

The semantics of a  $\mathbf{M}$ -diagram  $D : a \rightarrow b$  is a map  $[a] \rightarrow [b] \times \mathbf{M}$  which associates with an input configuration  $(c, p)$ , an output configuration  $(c', p')$  and a side effect  $U_k \dots U_1 \in \mathbf{M}$  which represents the action applied on a data register of the particle. Thus the semantics of a diagram can be formulated as follows:

► **Definition 2.** Given an  $\mathbf{M}$ -diagram  $D : a \rightarrow b$ , let  $\llbracket D \rrbracket : [a] \rightarrow [b] \times \mathbf{M}$  be inductively defined as:  $\forall D_1 : a \rightarrow b, D_2 : b \rightarrow d, D_3 : d \rightarrow e, D_4 : a \oplus f \rightarrow b \oplus f$ , where  $f \in \{\top, \mathbf{v}, \mathbf{h}\}$ :

$$\begin{aligned}
\llbracket \text{---} \oplus \text{---} \rrbracket &= \begin{cases} (\mathbf{V}, 0) \mapsto ((\mathbf{V}, 0), I) \\ (\mathbf{H}, 0) \mapsto ((\mathbf{H}, 1), I) \end{cases} & \llbracket \text{---} \oplus \text{---} \rrbracket &= \begin{cases} (\mathbf{V}, 0) \mapsto ((\mathbf{V}, 1), I) \\ (\mathbf{H}, 0) \mapsto ((\mathbf{H}, 0), I) \end{cases} \\
\llbracket \text{---} \oplus \text{---} \rrbracket &= \begin{cases} (\mathbf{V}, 0) \mapsto ((\mathbf{V}, 0), I) \\ (\mathbf{H}, 1) \mapsto ((\mathbf{H}, 0), I) \end{cases} & \llbracket \text{---} \oplus \text{---} \rrbracket &= \begin{cases} (\mathbf{V}, 1) \mapsto ((\mathbf{V}, 0), I) \\ (\mathbf{H}, 0) \mapsto ((\mathbf{H}, 0), I) \end{cases} \\
\llbracket \begin{array}{c} a \\ \oplus \\ b \end{array} \rrbracket &= (c, p) \mapsto \begin{cases} ((c, p), I) & \text{if } c = \mathbf{V} \\ ((c, 1 - p), I) & \text{if } c = \mathbf{H} \end{cases} & \llbracket \begin{array}{c} a \\ \oplus \\ b \end{array} \rrbracket &= (c, p) \mapsto ((c, 1 - p), I) \\
\llbracket \text{---} \oplus \text{---} \rrbracket &= (c, 0) \mapsto ((c, 0), I) & \llbracket \begin{array}{c} a \\ \oplus \\ \ominus \end{array} \rrbracket &= \begin{cases} (\mathbf{V}, 0) \mapsto ((\mathbf{H}, 0), I) \\ (\mathbf{H}, 0) \mapsto ((\mathbf{V}, 0), I) \end{cases} \\
\llbracket \begin{array}{c} a \\ \oplus \\ U \end{array} \rrbracket &= (c, 0) \mapsto ((c, 0), U) \\
\llbracket D_1 \oplus D_3 \rrbracket &= (c, p) \mapsto \begin{cases} \llbracket D_1 \rrbracket (c, p) & \text{if } p < |a| \\ S_a(\llbracket D_3 \rrbracket (c, p - |a|)) & \text{otherwise} \end{cases} & \llbracket D_2 \circ D_1 \rrbracket &= \llbracket D_2 \rrbracket \circ \llbracket D_1 \rrbracket \\
\llbracket Tr_f(D_4) \rrbracket &= (c, p) \mapsto \begin{cases} \llbracket D_4 \rrbracket (c, p) & \text{if } \pi_{pos}(\llbracket D_4 \rrbracket (c, p)) < |b| \\ \llbracket D_4 \rrbracket \circ S_{a-b}(\llbracket D_4 \rrbracket (c, p)) & \\ \text{if } \pi_{pos}(\llbracket D_4 \rrbracket \circ S_{a-b}(\llbracket D_4 \rrbracket (c, p))) < |b| \leq \pi_{pos}(\llbracket D_4 \rrbracket (c, p)) & \\ \llbracket D_4 \rrbracket \circ S_{a-b}(\llbracket D_4 \rrbracket \circ S_{a-b}(\llbracket D_4 \rrbracket (c, p))) & \text{otherwise} \end{cases}
\end{aligned}$$

where the composition is:  $g \circ f(c, p) = ((c'', p''), U'U)$  with  $f(c, p) = ((c', p'), U)$  and  $g(c', p') = ((c'', p''), U')$ ;  $\pi_{pos} : [a] \times \mathbf{M} \rightarrow \mathbb{N} = ((c, p), U) \mapsto p$  is the projector on the position, and  $S_a : [b] \times \mathbf{M} \rightarrow [a \oplus b] \times \mathbf{M} = ((c, p), U) \mapsto ((c, p + |a|), U)$  and  $S_{a-b} : [b] \times \mathbf{M} \rightarrow [a] \times \mathbf{M} = ((c, p), U) \mapsto ((c, p + |a| - |b|), U)$  shift the position.

Given  $D : a \rightarrow b$  and  $(c, p) \in [a]$ , we denote respectively by  $c_{c,p}^D$ ,  $p_{c,p}^D$  and  $U_{c,p}^D$  the polarisation, the position and the element of  $\mathbf{M}$ , such that  $\llbracket D \rrbracket (c, p) = ((c_{c,p}^D, p_{c,p}^D), U_{c,p}^D)$ . In the case where  $\mathbf{M}$  is the free monoid  $\mathcal{G}^*$ , its elements can be seen as words, so we will use the notation  $w_{c,p}^D$  instead of  $U_{c,p}^D$ .

Notice that the semantics of the trace is not defined as a fixed point but as a finite number of unfoldings. Indeed, like for PBS-diagrams, one can show that any wire of a diagram is used at most twice, each time with a distinct polarisation.

► **Proposition 3.**  $\llbracket \cdot \rrbracket$  is well defined, i.e. the axioms of the traced coloured prop are sound and the semantics of the trace is well defined.

### 3.1 Quantum semantics

Any diagram whose underlying monoid consists of linear maps, admits a *quantum semantics* defined as follows:

► **Definition 4** (Quantum semantics). Given a monoid  $\mathbf{M}$  of linear maps (with the standard composition) on a complex vector space  $\mathcal{V}$ , for any  $\mathbf{M}$ -diagram  $D : a \rightarrow b$  the quantum semantics of  $D$  is the linear map  $V_D : \mathbb{C}^{[a]} \otimes \mathcal{V} \rightarrow \mathbb{C}^{[b]} \otimes \mathcal{V} = |c, p\rangle \otimes |\phi\rangle \mapsto |c_{c,p}^D, p_{c,p}^D\rangle \otimes U_{c,p}^D |\phi\rangle$ .

The diagrams in  $\mathbf{Diag}^{\mathcal{H}}$  are valid by construction, in the sense that their semantics are valid quantum evolutions:

► **Proposition 5.** For any  $D \in \mathbf{Diag}^{\mathcal{H}}$ ,  $V_D : \mathbb{C}^{[a]} \otimes \mathcal{H} \rightarrow \mathbb{C}^{[b]} \otimes \mathcal{H}$  is an isometry.

Note that  $\llbracket D \rrbracket = \llbracket D' \rrbracket$  implies  $V_D = V_{D'}$ ; the converse is true if and only if  $0 \notin \mathbf{M}$ :

► **Proposition 6.** Given a monoid  $\mathbf{M}$  of complex linear maps, we have  $\forall D, D', \llbracket D \rrbracket = \llbracket D' \rrbracket \Leftrightarrow V_D = V_{D'}$ , if and only if  $0 \notin \mathbf{M}$ .

In particular, two diagrams in  $\mathbf{Diag}^{\mathcal{H}}$  have the same action semantics if and only if they have the same quantum semantics.

### 3.2 Interpretation

Given a monoid homomorphism  $\gamma : \mathbf{M} \rightarrow \mathbf{M}'$ , one can transform any  $\mathbf{M}$ -diagram into a  $\mathbf{M}'$ -diagram straightforwardly, by applying  $\gamma$  on each gate of the diagram:

► **Definition 7.** Given a  $\mathbf{M}$ -diagram  $D : a \rightarrow b$  and a monoid homomorphism  $\gamma : \mathbf{M} \rightarrow \mathbf{M}'$ , we define its  $\gamma$ -interpretation  $\gamma(D) : a \rightarrow b$  as the  $\mathbf{M}'$ -diagram obtained by applying  $\gamma$  to each gate of  $D$ . It is defined inductively as:  $\gamma(\overset{a}{\square} U \text{---} : a \rightarrow a) = \overset{a}{\square} \gamma(U) \text{---} : a \rightarrow a$ , for any other generator  $g$ ,  $\gamma(g) = g$ ,  $\gamma(D_2 \circ D_1) = \gamma(D_2) \circ \gamma(D_1)$ ,  $\gamma(D_1 \oplus D_2) = \gamma(D_1) \oplus \gamma(D_2)$ , and  $\gamma(\text{Tr}_e(D)) = \text{Tr}_e(\gamma(D))$ .

► **Proposition 8.** Any  $\mathbf{M}$ -diagram is the interpretation of an abstract diagram.

It is easy to see that the action of monoid homomorphisms on diagrams is well-behaved with respect to the semantics:

► **Proposition 9.** Given any  $\mathbf{M}$ -diagram  $D : a \rightarrow b$  and any monoid homomorphism  $\gamma : \mathbf{M} \rightarrow \mathbf{M}'$ , for any configuration  $(c, p) \in [a]$ , if  $\llbracket D \rrbracket (c, p) = ((c', p'), U)$  then  $\llbracket \gamma(D) \rrbracket (c, p) = ((c', p'), \gamma(U))$ .

As a consequence, given two abstract diagrams  $D_1, D_2 \in \mathbf{Diag}^{\mathcal{G}^*}$ , if  $\llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket$  then for any homomorphism  $\gamma : \mathcal{G}^* \rightarrow \mathbf{M}$ ,  $\llbracket \gamma(D_1) \rrbracket = \llbracket \gamma(D_2) \rrbracket$ . The converse is not true in general. Notice that in the framework of graphical languages an equation holds in graphical languages for traced symmetric (resp. dagger compact closed) monoidal categories if and only if it holds in finite-dimensional vector (resp. Hilbert) spaces [22, 28]. We prove a similar result by showing that interpreting abstract diagrams using 2-dimensional Hilbert spaces is enough to completely characterise their semantics:

► **Proposition 10.** *Given a Hilbert space  $\mathcal{H}$  of dimension at least 2 and a set  $\mathcal{G}$ ,  $\forall D_1, D_2 \in \mathbf{Diag}^{\mathcal{G}^*}$ , there exists a monoid homomorphism  $\gamma: \mathcal{G}^* \rightarrow \mathcal{U}(\mathcal{H})$  s.t.  $\llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket \Leftrightarrow \llbracket \gamma(D_1) \rrbracket = \llbracket \gamma(D_2) \rrbracket$ .*

A stronger result, where the homomorphism  $\gamma$  is independent of the diagrams, is also true, assuming the axiom of choice:

► **Proposition 11.** *Given a Hilbert space  $\mathcal{H}$  of dimension at least 2, and a set  $\mathcal{G}$  of cardinality at most the cardinality of  $\mathcal{U}(\mathcal{H})$ , there exists a monoid homomorphism  $\gamma: \mathcal{G}^* \rightarrow \mathcal{U}(\mathcal{H})$  s.t.  $\forall D_1, D_2 \in \mathbf{Diag}^{\mathcal{G}^*}$ ,  $\llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket \Leftrightarrow \llbracket \gamma(D_1) \rrbracket = \llbracket \gamma(D_2) \rrbracket$ .*

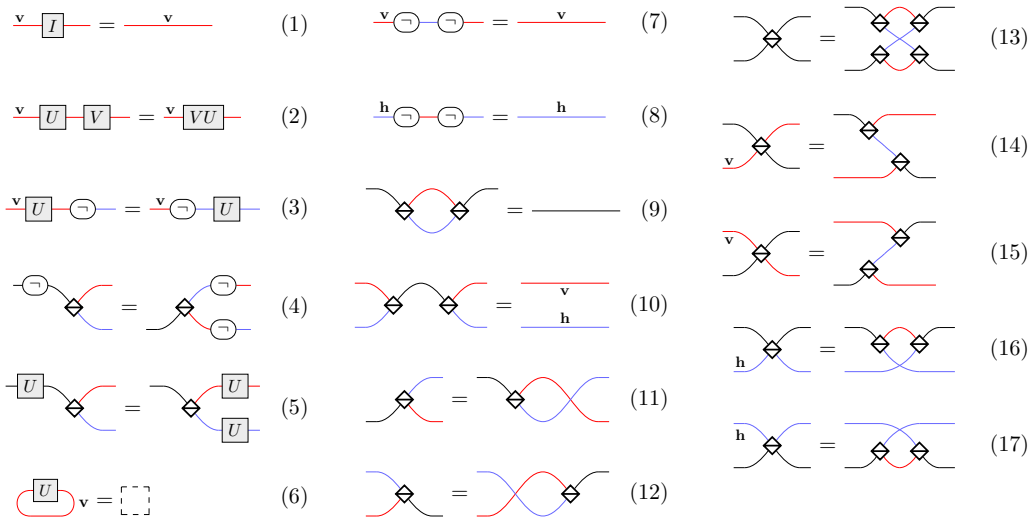
► **Remark 12.** Notice that the cardinality of  $\mathcal{U}(\mathcal{H})$  is  $\max(2^{\aleph_0}, 2^{\dim(\mathcal{H})})$  (where  $2^{\aleph_0}$  is the cardinality of  $\mathbb{R}$ ).

## 4 Equational theory

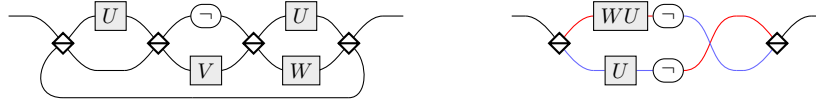
In this section, we introduce an equational theory which allows one to transform any M-diagram into an equivalent one. Indeed, all the equations we present in this section preserve the semantics of the diagrams (see Proposition 14).

These equations are summarised in Figure 4. They form what we call the CPBS-calculus:

► **Definition 13** (CPBS-calculus). *Two M-diagrams  $D_1, D_2$  are equivalent according to the rules of the CPBS-calculus, denoted  $\text{CPBS} \vdash D_1 = D_2$ , if one can transform  $D_1$  into  $D_2$  using the equations given in Figure 4. More precisely,  $\text{CPBS} \vdash \cdot = \cdot$  is defined as the smallest congruence which satisfies equations of Figure 4 in addition to the axioms of coloured traced prop.*



■ **Figure 4** Axioms of the CPBS-calculus.  $U, V \in M$ . Equations (1) and (2) reflect the monoid structure of M; Equations (3) to (5) show how the three generators commute; Equation (6) means that a disconnected diagram (with no inputs/outputs) can be removed; Equations (7) to (10) witness the fact that the negation and the 3-leg PBS are invertible; Equations (11) and (12) are essentially topological rules; Equations (13) to (17) show how 4-leg PBS can be decomposed into 3-leg PBS. Notice in particular that the other rules do not use 4-leg PBS, as a consequence one could define the language using 3-leg PBS only and see the 4-leg PBS as syntactic sugar.



■ **Figure 5** An example of a CPBS-diagram (*left*) and its equivalent diagram in normal form (*right*).

Notice that the CPBS-calculus subsumes the PBS-calculus: the fragment of monochromatic (black)  $\mathcal{H}$ -diagrams of the CPBS-calculus coincides with the set of PBS-diagrams, moreover, for any two PBS-diagrams  $D_1, D_2$ ,  $\text{PBS} \vdash D_1 = D_2$  if and only if  $\text{CPBS} \vdash D_1 = D_2$ .

► **Proposition 14** (Soundness). *For any two M-diagrams  $D_1$  and  $D_2$ , if  $\text{CPBS} \vdash D_1 = D_2$  then  $\llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket$ .*

We introduce normal forms, that will be useful to prove that the equational theory is complete, and will also play a role in optimising the number of gates in a diagram in Section 5.

► **Definition 15.** *A diagram is said to be in normal form if it is of the form  $M \circ P \circ F \circ G \circ S$ , where:*

- $S$  is of the form  $b_1 \oplus \dots \oplus b_n$ , where each  $b_i$  is either  $\underline{v}$ ,  $\underline{h}$  or  $\overline{\diamond}$
- $G$  is of the form  $g_1 \oplus \dots \oplus g_k$ , where each  $g_i$  is either  $\underline{v}$ ,  $\underline{h}$ , or  $\underline{v} \boxed{U_i}$  or  $\underline{h} \boxed{U_i}$  with  $U_i \neq I$
- $F$  is of the form  $n_1 \oplus \dots \oplus n_k$ , where each  $n_i$  is either  $\underline{v}$ ,  $\underline{h}$ ,  $\underline{v} \ominus$  or  $\underline{h} \ominus$
- $P$  is a permutation of the wires, that is, a trace-free diagram in which all generators are identity wires or swaps
- $M$  is of the form  $w_1 \oplus \dots \oplus w_m$ , where each  $w_i$  is either  $\underline{v}$ ,  $\underline{h}$  or  $\overline{\diamond}$ .

For example, the diagram shown in Figure 3 (*right*) is in normal form.

► **Theorem 16.** *For any M-diagram  $D$ , there exists a M-diagram in normal form  $N$  such that  $\text{CPBS} \vdash D = N$ .*

Note that the structure of the normal form as well as the proof of Theorem 16 use in an essential way the removal of useless wires made possible by the use of colours, and in particular Equation (10), which has no equivalent in the monochromatic PBS-calculus of [12]. An example of CPBS-diagram and its normal form are given in Figure 5.

Now we use the normal form to prove the completeness of the CPBS-calculus:

► **Lemma 17** (Uniqueness of the normal form). *For any two diagrams in normal form  $N$  and  $N'$ , if  $\llbracket N \rrbracket = \llbracket N' \rrbracket$  then  $N = N'$ .*

► **Theorem 18** (Completeness). *Given any two M-diagrams  $D_1$  and  $D_2$ , if  $\llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket$  then  $\text{CPBS} \vdash D_1 = D_2$ .*

Finally, each equation of Figure 4 is necessary for the completeness:

► **Theorem 19** (Minimality). *None of the equations of Figure 4 is a consequence of the others.*

## 5 Resource optimisation

We show in this section that the equational theory of the CPBS-calculus can be used for resource optimisation.



### 5.1 Minimising the number of oracle queries

We consider the problem of minimising the number of oracle queries: given a set  $\mathcal{G}$  of (distinct) oracles and a  $\mathcal{G}^*$ -diagram  $D$ , the objective is to find a diagram  $D'$  equivalent to  $D$  (i.e.  $\llbracket D \rrbracket = \llbracket D' \rrbracket$ ) such that  $D'$  is using a minimal number of queries to each oracle. Since there are several oracles, the definition of the optimal diagrams should be made precise.

First, we define the number of queries to a given oracle:

► **Definition 20.** *Given a  $\mathcal{G}^*$ -diagram  $D$ , for any  $U \in \mathcal{G}$ , let  $\#_U(D)$  be the number of queries to  $U$  in  $D$ , inductively defined as follows:  $\#_U(\overline{a[w]}) = |w|_U$ ,  $\#_U(g) = 0$  for all the other generators, and  $\#_U(D_1 \oplus D_2) = \#_U(D_2 \circ D_1) = \#_U(D_1) + \#_U(D_2)$ ,  $\#_U(Tr_a(D)) = \#_U(D)$ , where  $|w|_U$  is the number of occurrences of  $U$  in the word  $w \in \mathcal{G}^*$ .*

We can now define a query-optimal diagram as follows:

► **Definition 21.** *A  $\mathcal{G}^*$ -diagram  $D$  is query-optimal if  $\forall D' \in \mathbf{Diag}^{\mathcal{G}^*}$ ,  $\forall U \in \mathcal{G}$ ,  $\llbracket D \rrbracket = \llbracket D' \rrbracket$  implies  $\#_U(D) \leq \#_U(D')$ .*

Notice that given a diagram, it is not a priori guaranteed that there exists an equivalent diagram which is query-optimal, if for instance, all the diagrams which minimise the number of queries to some oracle  $U$  do not minimise the number of queries to another oracle  $V$ . We actually show (Proposition 23) that any diagram can be turned into a query-optimal one. To this end, we first need a lower-bound on the number of queries to a given oracle:

► **Proposition 22 (Lowerbound).** *For any  $\mathcal{G}^*$ -diagram  $D : a \rightarrow b$  and any  $U \in \mathcal{G}$ ,  $\#_U(D) \geq \left\lceil \sum_{(c,p) \in X} \frac{|w_{c,p}^D|_U}{2} \right\rceil$  where  $w_{c,p}^D \in \mathcal{G}^*$  is such that  $\llbracket D \rrbracket(c, p) = (c', p', w_{c,p}^D)$ .*

Notice that Proposition 22 provides a lower bound on the minimal number of queries to  $U$  one can reach in optimising a diagram since the right-hand side of the inequality only depends on the semantics of the diagram.

We are now ready to introduce an optimisation procedure that transforms any diagram into an equivalent query-optimal one:

#### Query optimisation procedure of a $\mathcal{G}^*$ -diagram $D$ .

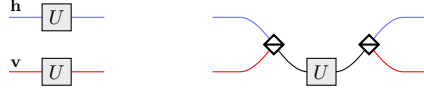
1. Transform  $D$  into its normal form  $D_{NF}$ . A recursive procedure for doing this can easily be deduced from the proof of Theorem 16.
2. Split all gates into elementary gates (that is, gates whose label is a single letter), using the following variants of Equation (2), which are consequences of the equations of Figure 4 (see [13]):  $\forall U \in \mathcal{G}$ ,  $\forall w \in \mathcal{G}^*$ ,  $w \neq I$ :

$$\overline{wU} \rightarrow \overline{U} \overline{w} \quad (18) \qquad \underline{wU} \rightarrow \underline{U} \underline{w} \quad (19) \qquad \overline{wU} \rightarrow \overline{U} \overline{w} \quad (20)$$

3. As long as the diagram contains two nonblack gates with the same label, merge them. To do so, deform the diagram to put one over the other, and apply one of the following equations, which are also consequences of the equations of Figure 4 :

$$\begin{array}{c} \overline{U} \\ \underline{U} \end{array} \rightarrow \begin{array}{c} \overline{U} \\ \underline{U} \end{array} \quad (21) \qquad \begin{array}{c} \underline{U} \\ \overline{U} \end{array} \rightarrow \begin{array}{c} \underline{U} \\ \overline{U} \end{array} \quad (22)$$

$$\begin{array}{c} \overline{U} \\ \overline{U} \end{array} \rightarrow \begin{array}{c} \overline{U} \\ \overline{U} \end{array} \quad (23) \qquad \begin{array}{c} \underline{U} \\ \underline{U} \end{array} \rightarrow \begin{array}{c} \underline{U} \\ \underline{U} \end{array} \quad (24)$$



■ **Figure 6** Two equivalent diagrams: the diagram on the left is optimal in terms of number of polarising beam splitters, the diagram on the right is optimal in terms of queries. Notice there is no equivalent diagram with no polarising beam splitter and at most a single query.

An example of query-optimised diagram is given in Figure 8. The query-optimisation procedure transforms any diagram into an equivalent query-optimal one:

► **Proposition 23.** *The diagram  $D_0$  output by the query optimisation procedure is query-optimal: for any  $U$  and any  $D'$  s.t.  $\llbracket D' \rrbracket = \llbracket D_0 \rrbracket$ , one has  $\#_U(D_0) \leq \#_U(D')$ .*

Notice that the query-optimisation procedure is efficient: one can naturally define the size  $|D|$  of a diagram  $D \in \mathbf{Diag}^{\mathcal{G}^*}$  as follows:  $|\overline{a[w]}| = |w|$ ,  $|g| = 1$  for all the other generators, and  $|D_1 \oplus D_2| = |D_2 \circ D_1| = |D_1| + |D_2|$ ,  $|Tra(D)| = |D| + 1$ . Step 1 of the procedure, which consists in putting the diagram in normal form, can be done using a number of elementary equations of Figure 4 which is quadratic in the size of the diagram, the other two steps being linear. Notice that here we only count the number of basic equations, but it requires also some diagrammatic transformations, which can be handled efficiently using appropriate data structures.

## 5.2 Optimising both queries and PBS

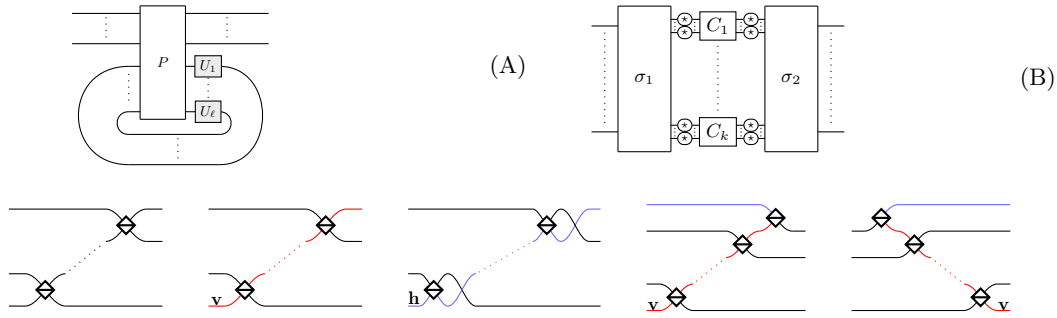
We refine the resource optimisation of a diagram by considering not only the number of queries but also the number of instructions, and in particular the number of polarising beam splitters. Notice that the number of beam splitters and the number of queries cannot be minimised independently, in the sense that there might not exist a diagram that is both query-optimal and PBS-optimal (see such an example in Figure 6). As the implementation of an oracle is a priori more expensive than the implementation of a single PBS, we optimise the number of queries and then the number of PBS in this order, i.e. the measure of complexity is the lexicographic order number of queries, number of polarising beam splitters.

► **Definition 24.** *A diagram  $D$  is query-PBS-optimal if  $D$  is query-optimal and for any query-optimal diagram  $D'$  equivalent to  $D$  (i.e.  $\llbracket D \rrbracket = \llbracket D' \rrbracket$ ),  $\#_{\text{PBS}}(D) \leq \#_{\text{PBS}}(D')$ , where  $\#_{\text{PBS}}(D)$  be the number of PBS of  $D$ .*

We introduce an efficient heuristic, called *PGT procedure* that, when applied on a query-optimal diagram  $D_0$ , preserves the number of queries. Moreover, the produced diagram, called in PGT form (see Figure 7), is query-PBS-optimal when there is at most one query to each oracle:

► **Theorem 25.** *Any query-optimal diagram in PGT form that does not contain two queries to the same oracle (i.e.  $\forall U \in \mathcal{G}, \#_U(D) \leq 1$ ) is query-PBS-optimal.*

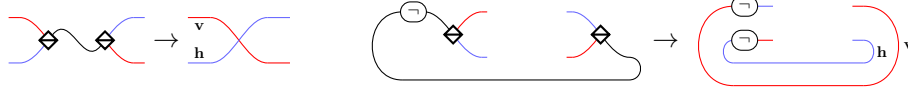
The procedure relies on equations of Figure 4, together with easy to derive variants of these equations. The procedure, with all steps detailed, more pictures and explicit statement of the variants of the equations, is given in the preprint version of the paper [13].



■ **Figure 7** Schematic description of a diagram in PGT form (for Permutation, Gates and Traces). A diagram is in PGT form if it is of the form (A), with  $P$  of the form (B), and the  $C_i$  of the forms depicted on the second line.  $\text{-(}\ast\text{)}$  denotes either  $\frac{a}{b}$  or  $\frac{a}{\neg}$  with  $a \in \{\mathbf{v}, \mathbf{h}\}$ , and  $\sigma_1, \sigma_2$  are permutations of the wires.

**PGT procedure.** Given a query optimal diagram  $D_0$ :

0. During all the procedure, every time there are two consecutive negations, we remove them using Equation (7), (8) or their all-black version.
1. Deform the gate-optimal diagram  $D_0$  to put it in the form (A) with  $P$  gate-free. The goal of the following steps is to put  $P$  in stair form.
2. Split all PBS of the form  $\frac{a}{b}$  into combinations of  $\frac{a}{b}$ ,  $\frac{a}{\neg}$ ,  $\frac{a}{\neg}$  and  $\frac{a}{\neg}$ , using Equations (13) to (17).
3. As long as there are two PBS connected by a black wire, with possibly a black negation on this wire, push the possibly remaining negation out using Equation (4), and cancel the PBS together using Equation (10) and its variants. For example:



When there are not two such PBS anymore, all black wires are connected to at least one side of  $P$  (possibly through negations), and the PBS are connected together with red and blue wires with possibly negations on them.

4. Remove all isolated loops. Note that since  $D_0$  is query-optimal, there cannot be loops containing gates at this point.

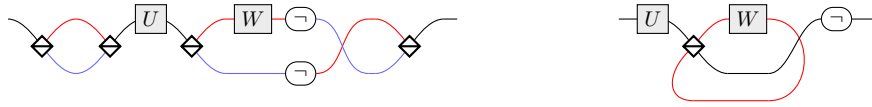
5. Deform  $P$  to put it in the form (B) with the  $C_i$  of the form  $\frac{a}{b}$  and  $\sigma_1$  and  $\sigma_2$  being wire permutations, where  $\text{-(}\ast\text{)}$  is either  $\frac{a}{b}$  or  $\frac{a}{\neg}$  with  $a \in \{\mathbf{v}, \mathbf{h}\}$ ,  $\frac{a}{b}$  is either  $\frac{a}{b}$  or  $\frac{a}{\neg}$  and  $\frac{a}{\neg}$  is either  $\frac{a}{b}$  or  $\frac{a}{\neg}$ .

6. Remove the negations in the middle of the  $C_i$  by pushing them to the bottom by means of variants of Equation (4).

7. Transform each  $C_i$ , which is now, up to deformation, a ladder of PBS without negations, into one of the five kinds of stairs depicted in Figure 7, depending on its type. To do so, deform it and apply Equations (11) and (12) appropriately, and repeatedly apply the appropriate equation among (14), (15), (16), and a variant of (13). This gives us  $D_1$ .

An example of diagram produced by the PGT procedure is given in Figure 8.

Since the PGT procedure consists in putting a subdiagram of  $D_0$  in stair form (except Step 1 which is just deformation and does not change the number of PBS), this procedure does not increase the number of PBS in  $D_0$ :



■ **Figure 8** The diagram on the left is the obtained by applying the query-optimisation procedure on the example of Figure 5. The diagram on the right is (up to deformation) obtained by applying the PGT procedure to the diagram on the left. Notice that this diagram is both query- and PBS-optimal.

► **Proposition 26.** *The diagram  $D_1$  output by the PGT procedure contains at most as many PBS as the initial diagram  $D_0$ .*

This also implies that given any diagram  $D$ , there exists an equivalent query-PBS-optimal diagram in PGT form. Indeed, by Proposition 23, there exist query-optimal diagrams equivalent to  $D$ , and among these diagrams, some of them have minimal number of PBS and are therefore query-PBS-optimal. Finally, applying the PGT procedure to one of these diagrams gives us an equivalent diagram in PGT form, which, since the PGT procedure does not change the gates or increase the number of PBS, is still query-PBS-optimal.

Applying the PGT procedure after the query optimisation procedure produces an interesting heuristic: the output diagram is necessarily query-optimal and can even be query-PBS-optimal when it does not contain two queries to the same oracle.

Notice that, like the query optimisation procedure, the PGT procedure is efficient, i.e. it can be done using a number of elementary graphical transformations (those of Figure 4) which is linear in the size of the diagram. Moreover, it also requires some diagrammatic transformations, which can be handled using appropriate data structures, leading to a quadratic algorithm.

### 5.3 Hardness

We show in this section that the query-PBS optimisation problem is actually NP-hard.

► **Theorem 27.** *The problem of, given an abstract diagram, finding an equivalent query-PBS-optimal diagram, is NP-hard.*

The proof, given in [13], is based on a reduction from the maximum Eulerian cycle decomposition problem (MAX-ECD) which is known to be NP-hard [8]. The MAX-ECD problem consists, given a graph, in finding a partition of its set of edges into the maximum number of cycles. Intuitively, the reduction goes as follows: given an Eulerian graph  $G = (V = \{v_0, \dots, v_{n-1}\}, E)$ , let  $\sigma$  be a permutation of the vertices of the graph s.t.  $\forall i, (v_i, \sigma(v_i)) \in E$  (such a  $\sigma$  exists since  $G$  is Eulerian), we construct a  $V^*$ -diagram  $D$  such that the number of occurrences of each  $v_i$  in  $D$  is half its degree in  $G$ ; and such that  $\forall i, \llbracket D \rrbracket (\mathbf{V}, i) = ((\mathbf{V}, i), v_i)$  and  $\llbracket D \rrbracket (\mathbf{H}, i) = ((\mathbf{H}, i), \sigma(v_i))$ . Roughly speaking, we show that the edge-partitions of  $G$  into cycles correspond to the possible implementations of  $D$ , and that a partition with a maximal number of cycles leads to an implementation with a minimal number of PBS.

In the following, we explore a few variants of the problem, which remain NP-hard.

First, query-PBS optimisation is still hard when restricted to negation-free diagrams:

► **Corollary 28.** *The problem of, given a negation-free abstract diagram, finding an equivalent diagram which is query-PBS-optimal among negation-free diagrams, is NP-hard.*

Additionally, it is also hard, in a query-optimal diagram, to optimise the gates and the negations together by, respectively, defining a cost function (at least in the case where the negation cost is not less than the PBS), prioritising the negations over the PBS, and

prioritising the PBS over the negations. Note that the NP-hardness is clear in the third case since the considered problem is a refinement of the query-PBS-optimisation problem addressed in Theorem 27.

► **Corollary 29.** *For any  $\alpha \geq 1$ , the problem of, given an abstract diagram  $D$ , finding an equivalent query-optimal diagram  $D'$  such that  $\#_{\text{PBS}}(D') + \alpha\#_{-}(D')$  is minimal, is NP-hard, where  $\#_{-}(D)$  is the number of negations in  $D$ .*

► **Corollary 30.** *The problem of, given an abstract diagram  $D$ , finding an equivalent query- $\neg$ -PBS-optimal<sup>1</sup> diagram is NP-hard.*

## 6 Discussions and Future Work

The power and limits of quantum coherent control is an intriguing question. Maybe surprisingly, we have proved that coherently controlled quantum computations, when expressed in the PBS-calculus, can be efficiently optimised: any PBS-diagram can be transformed in polynomial time into a diagram that is optimal in terms of oracle queries. We have refined the procedure to also decrease the number of polarising beam splitters. It leads to an optimal diagram when each oracle is queried only once, but the corresponding optimisation problem is NP-hard in general. We leave to future work an experimental evaluation of the PGT procedure when each oracle is not necessarily queried only once.

To perform the resource optimisation, we have introduced a few add-ons to the framework of the PBS-calculus. First, we have refined the syntax in order to allow the representation of unsaturated (or 3-leg) polarising beam splitters. They are essential ingredients for resource optimisation, as they provide a way to decompose a diagram into elementary components and then remove the useless ones. However, notice that one can perform resource optimisation of vanilla PBS-diagrams, using the refined one only as an intermediate language. Indeed, given a vanilla PBS-diagram (where all wires are black), one can apply the optimisation procedures described in this paper. The resulting optimised PBS-diagram may contain some unsaturated PBS, but all these 3-leg PBS can be saturated by adding useless traces and then one can make the diagram monochromatic. The resulting vanilla PBS-diagram keeps the same number of queries and PBS.

We have also generalised the gates of the diagrams, by considering arbitrary monoids. This is a natural abstraction that allows one to consider various examples and in particular the one of the free monoid which is appropriate to model the oracle queries. The query complexity is a convenient model to prove lower bounds, but note that the optimisation procedures described in this paper can be applied with any arbitrary monoid (for instance using Proposition 8). However, there is no guarantee of minimality with an arbitrary monoid.

Another direction of research is to consider resource optimisation in a more expressive language for quantum control. Indeed, the polarisation of a particle can only be flipped within a PBS-diagram. The PBS-calculus is well suited for most applications of coherent control in quantum computing, by allowing the description of superpositions of classical controls (in particular superposition of causal orders) since the input particle can be in any superposition of polarisations. However, it would be interesting to develop resource optimisation techniques for quantum computation involving arbitrary quantum control.

<sup>1</sup> A diagram is query- $\neg$ -PBS-optimal if it is optimal according to the lexicographic order: the number of queries then the number of negations and finally the number of polarising beam splitters.

## References

- 1 Alastair A. Abbott, Julian Wechs, Dominic Horsman, Mehdi Mhalla, and Cyril Branciard. Communication through coherent control of quantum channels. *Quantum*, 4:333, September 2020. doi:10.22331/q-2020-09-24-333.
- 2 Thorsten Altenkirch and Jonathan Grattage. A functional quantum programming language. In *20th Annual IEEE Symposium on Logic in Computer Science (LICS'05)*, pages 249–258. IEEE, 2005.
- 3 Matthew Amy, Dmitri Maslov, and Michele Mosca. Polynomial-time T-depth optimization of Clifford+T circuits via matroid partitioning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(10):1476–1489, 2014.
- 4 Mateus Araújo, Fabio Costa, and Časlav Brukner. Computational advantage from quantum-controlled ordering of gates. *Physical Review Letters*, 113(25):250402, 2014. doi:10.1103/PhysRevLett.113.250402.
- 5 Pablo Arrighi, Christopher Cedzich, Marin Costes, Ulysse Rémond, and Benoît Valiron. Addressable quantum gates. *arXiv preprint*, 2021. arXiv:2109.08050.
- 6 Costin Badescu and Prakash Panangaden. Quantum alternation: Prospects and problems. In Chris Heunen, Peter Selinger, and Jamie Vicary, editors, *Proceedings 12th International Workshop on Quantum Physics and Logic, QPL 2015, Oxford, UK, July 15-17, 2015*, volume 195 of *EPTCS*, pages 33–42, 2015. doi:10.4204/EPTCS.195.3.
- 7 Cyril Branciard, Alexandre Clément, Mehdi Mhalla, and Simon Perdrix. Coherent control and distinguishability of quantum channels via PBS-diagrams. In Filippo Bonchi and Simon J. Puglisi, editors, *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021)*, volume 202 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:20, Dagstuhl, Germany, August 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2021.22.
- 8 Alberto Caprara. Sorting permutations by reversals and Eulerian cycle decompositions. *SIAM Journal on Discrete Mathematics*, 12(1):91–110, 1999. doi:10.1137/S089548019731994X.
- 9 Giulio Chiribella. Perfect discrimination of no-signalling channels via quantum superposition of causal structures. *Physical Review A*, 86(4):040301, 2012. doi:10.1103/PhysRevA.86.040301.
- 10 Giulio Chiribella, Giacomo Mauro D’Ariano, Paolo Perinotti, and Benoît Valiron. Quantum computations without definite causal structure. *Physical Review A*, 88:022318, August 2013. doi:10.1103/PhysRevA.88.022318.
- 11 Alexandre Clément, Nicolas Heurtel, Shane Mansfield, Simon Perdrix, and Benoît Valiron. LOv-calculus: A graphical language for linear optical quantum circuits. Accepted at MFCS’22, 2022. arXiv:2204.11787.
- 12 Alexandre Clément and Simon Perdrix. PBS-calculus: A graphical language for coherent control of quantum computations. In Javier Esparza and Daniel Král, editors, *45th International Symposium on Mathematical Foundations of Computer Science (MFCS 2020)*, volume 170 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:14, Dagstuhl, Germany, August 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2020.24.
- 13 Alexandre Clément and Simon Perdrix. Minimising resources of coherently controlled quantum computations, 2022. doi:10.48550/ARXIV.2202.05260.
- 14 Timoteo Colnaghi, Giacomo Mauro D’Ariano, Stefano Facchini, and Paolo Perinotti. Quantum computation with programmable connections between gates. *Physics Letters A*, 376(45):2940–2943, 2012. doi:10.1016/j.physleta.2012.08.028.
- 15 Alejandro Díaz-Caro, Mauricio Guillermo, Alexandre Miquel, and Benoît Valiron. Realizability in the unitary sphere. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13. IEEE, 2019.
- 16 Gilles Dowek and Pablo Arrighi. Lineal: A linear-algebraic lambda-calculus. *Logical Methods in Computer Science*, 13, 2017.

- 17 Daniel Ebler, Sina Salek, and Giulio Chiribella. Enhanced communication with the assistance of indefinite causal order. *Physical Review Letters*, 120(12):120502, March 2018. doi:10.1103/PhysRevLett.120.120502.
- 18 Stefano Facchini and Simon Perdrix. Quantum circuits for the unitary permutation problem. In *International Conference on Theory and Applications of Models of Computation*, pages 324–331. Springer, 2015. doi:10.1007/978-3-319-17142-5\_28.
- 19 Adrien Feix, Mateus Araújo, and Časlav Brukner. Quantum superposition of the order of parties as a communication resource. *Physical Review A*, 92(5):052326, 2015. doi:10.1103/PhysRevA.92.052326.
- 20 Philippe Allard Guérin, Adrien Feix, Mateus Araújo, and Časlav Brukner. Exponential communication complexity advantage from quantum superposition of the direction of communication. *Physical Review Letters*, 117(10):100502, 2016. doi:10.1103/PhysRevLett.117.100502.
- 21 Lucien Hardy. Probability theories with dynamic causal structure: a new framework for quantum gravity. *arXiv preprint gr-qc/0509120*, 2005.
- 22 Masahito Hasegawa, Martin Hofmann, and Gordon Plotkin. Finite dimensional vector spaces are complete for traced symmetric monoidal categories. In *Pillars of computer science*, pages 367–385. Springer, 2008.
- 23 Vadym Kliuchnikov and Dmitri Maslov. Optimization of Clifford circuits. *Physical Review A*, 88(5):052307, 2013.
- 24 Saunders MacLane. Categorical algebra. *Bulletin of the American Mathematical Society*, 71(1):40–106, 1965.
- 25 Yunseong Nam, Neil J. Ross, Yuan Su, Andrew M. Childs, and Dmitri Maslov. Automated optimization of large quantum circuits with continuous parameters. *npj Quantum Information*, 4(1):1–12, 2018.
- 26 Ognjan Oreshkov, Fabio Costa, and Časlav Brukner. Quantum correlations with no causal order. *Nature communications*, 3(1):1–8, 2012.
- 27 Martin J. Renner and Časlav Brukner. Reassessing the computational advantage of quantum-controlled ordering of gates. *Physical Review Research*, 3(4):043012, 2021.
- 28 Peter Selinger. Finite dimensional hilbert spaces are complete for dagger compact closed categories. *Electronic Notes in Theoretical Computer Science*, 270(1):113–119, 2011.
- 29 Augustin Vanrietvelde, Hlér Kristjánsson, and Jonathan Barrett. Routed quantum circuits. *Quantum*, 5:503, 2021.
- 30 Julian Wechs, Hippolyte Dourdent, Alastair A. Abbott, and Cyril Branciard. Quantum circuits with classical versus quantum control of causal order. *arXiv preprint*, 2021. arXiv:2101.08796.
- 31 Matt Wilson and Giulio Chiribella. A diagrammatic approach to information transmission in generalised switches. *arXiv preprint*, 2020. arXiv:2003.08224.
- 32 Mingsheng Ying, Nengkun Yu, and Yuan Feng. Defining quantum control flow. *arXiv preprint*, 2012. arXiv:1209.4379.
- 33 Magdalena Zych, Fabio Costa, Igor Pikovski, and Časlav Brukner. Bell’s theorem for temporal order. *Nature communications*, 10(1):1–10, 2019.