Combining quantum and classical control: syntax, semantics and adequacy

Kinnari Dave[1,2], Louis Lemonnier[3], Romain Péchoux[2], and Vladimir Zamdzhiev[1]

[1] Université Paris-Saclay, CNRS, ENS Paris-Saclay, Inria, Laboratoire Méthodes Formelles, 91190, Gif-sur-Yvette, France
[2] Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
[3] University of Edinburgh

Abstract. The two main notions of control in quantum programming languages are often referred to as "quantum" control and "classical" control. With the latter, the control flow is based on classical information, potentially resulting from a quantum measurement, and this paradigm is well-suited to mixed state quantum computation. Whereas with quantum control, we are primarily focused on pure quantum computation and there the "control" is based on superposition. The two paradigms have not mixed well traditionally and they are almost always treated separately. In this work, we show that the paradigms may be combined within the same system. The key ingredients for achieving this are: (1) syntactically: a modality for incorporating pure quantum types into a mixed state quantum type system; (2) operationally: an adaptation of the notion of "quantum configuration" from quantum lambda-calculi, where the quantum data is replaced with pure quantum primitives; (3) denotationally: suitable (sub)categories of Hilbert spaces, for pure computation and von Neumann algebras, for mixed state computation in the Heisenberg picture of quantum mechanics.

1 Introduction

There are two important paradigms in the design of quantum programming languages – "classical control" and "quantum control". In the classical control approach (e.g., [18,20,19,5,16,12]) the control flow of a program is conditioned on classical information that may result from quantum measurements. Type systems for quantum programming languages that are based on classical control are able to represent a variety of effects, e.g., quantum state preparation, state evolution through the application of unitary operators, and probabilistic effects induced by quantum measurements. Because of this, it is natural to conceptualise such lambda-calculi using quantum structures and models that are suited to describing mixed-state quantum computation and information (e.g., density matrices, superoperators). In the quantum control approach (e.g., [17,9,10,1,22]), one usually places an emphasis on pure state quantum computation and more specifically on superposition of terms, pure quantum states and unitary operators. In approaches that utilise classical control, one often starts with a selection

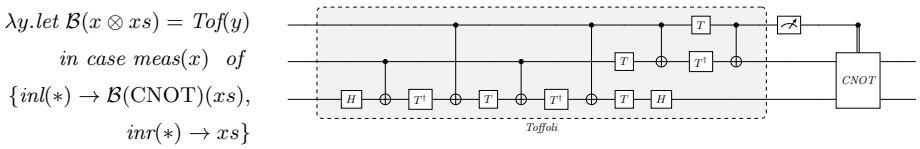


Fig. 1: A program and the corresponding circuit

of constants that represent some unitary operators (e.g., Hadamard, CNOT) and more complex unitary operations can be described in a circuit-like fashion by composing the atomic unitaries in a suitable way. Whereas in the quantum control approach, unitary operators are defined through more fundamental primitives that do not require the programmer to specify a circuit-like decomposition of the unitary operation. Because of this, a considerable economy in terms of syntax can be achieved with quantum control. For example, consider the circuit in Figure 1. It uses the well-known *Toffoli* gate, which would have to be decomposed as a large circuit like the one in the figure. However, the program representing this circuit can instead be written as a simple lambda term in our language given in Figure 1, where $Tof \triangleq \mathcal{B}(\text{ctrl CNOT})$. To highlight this further note that the controlled Hadamard can be defined similarly to the Toffoli in our language, but a typical classically controlled quantum programming language would require two H gates, six T gates and one $CNOT$ gate. Thus, our language offers an ease of writing programs by abstracting away quantum circuits. We aim to shift focus from drawing quantum circuits, which is a more *low-level* approach to writing code, to writing programs directly using syntax, which is a more *high-level* approach. In classical (non-quantum) programming, we can use high-level languages to write algorithms without having to specify boolean circuits or to encode integers using tuples of bits/booleans. With our paper, we are trying to advance quantum programming abstractions towards such a direction. For example, an individual qnat $|n\rangle$ can be used instead of a tensor of qubits $|q_1 q_2 ... q_k\rangle$ whose binary encoding corresponds to n. We view boolean circuits and binary encodings of integers as low-level and likewise for their quantum counterparts. Our aim is to develop higher-level quantum abstractions.

Our Contributions Because of the potential presence of quantum entanglement, it is impossible (in general) to decompose a quantum state into a non-trivial tensor product of two smaller quantum states. Indeed, in the quantum lambda-calculus (QLC) [14], which is based on classical control, the type **qbit** of qubits is an opaque type in the sense that there are no *closed* values of such a type. Program states may be described via *quantum configurations* which are triples $(|\psi\rangle, \ell, M)$, where $|\psi\rangle \in \mathbb{C}^{2^n}$ is a *pure* quantum state (possibly entangled); M is a program possibly containing free variables of type **qbit**; ℓ is a *linking* function that maps the free quantum variables of M to appropriate components of the state $|\psi\rangle$. It is possible to also view ℓ as a unitary permutation acting on $|\psi\rangle$ and this view is important for our development. Such configurations, even though they are not part of the user-facing syntax, allow us to reason

about entangled states and the operational semantics are described via a relation $(|\psi\rangle, \ell, M) \to_p (|\psi'\rangle, \ell', M')$, with p the probability of reduction.

The main idea that allows us to combine both quantum and classical control is to modify the quantum configurations described above by replacing the quantum state $|\psi\rangle$ with a suitable *pure quantum term* $\boldsymbol{t}$ and to replace the linking function ℓ with a suitable *unitary permutation* u_σ, where both $\boldsymbol{t}$ and u_σ are syntactic constructs from our pure quantum language that may be assigned types.

In order to accurately model the situation with the QLC, the term $\boldsymbol{t}$ has to correspond to a *normalised* vector and we replace the unitary constants from the QLC with programmable unitary constructs, so the pure subsystem also has to ensure their *unitarity*. Our pure subsystem is based on ideas described in [3,17], but we further build on this by introducing an *equational theory* for our pure subsystem, which has unique normal forms and we describe a *denotational semantics* for it that is *sound and complete* with respect to the equational theory.

Next, we integrate the pure quantum subsystem into our main calculus, which allows us to describe both quantum and classical information and, therefore also classical control. This is a variant of the QLC with the addition of a modality $\mathcal{B}(\boldsymbol{Q})$ which allows us to view pure quantum types $\boldsymbol{Q}$ as types of mixed state quantum operations in the Heisenberg picture. The intuition behind this is the following: mixed states in the Schrödinger picture can be seen as CPTP maps $(1 \mapsto \rho) \colon \mathbb{C} \to \mathcal{T}(H)$ whereas mixed states in the Heisenberg picture are given by functionals of the form $\mathrm{tr}(\rho -) \colon \mathcal{B}(H) \to \mathbb{C}$[4], both of which are determined by a choice of density operator $\rho \colon H \to H$. This modality allows us to replace the *constants* for state preparation and unitaries acting on qubits in the QLC with *terms and expressions* from the pure subsystem acting on more general quantum types beyond qubits. The operational semantics are described via a suitable adaptation of the aforementioned quantum configurations. The denotational semantics follows previous work on quantum programming semantics based on von Neumann algebras [5,16,12] and we show, in addition, that the assignment $\mathcal{B}(-)$ may be extended to a strict monoidal functor (between the relevant categories of Hilbert spaces and von Neumann algebras) that is crucial for our denotational semantics.

Overall, the two main contributions of our work are: (1) we show that quantum and classical control can be combined in a syntactic, operational and denotational sense by integrating a pure quantum control subsystem as part of the meta-theory and syntax of a quantum lambda-calculus; (2) an equational theory for a (sub)system for quantum control with unique normal forms and a sound and complete denotational semantics. Some supplementary material including extra examples, proofs, and complete figures for formation rule predicates are provided in Appendix.

Related Work We begin by commenting on semantic approaches to classical control. In [14], the authors describe a QLC where the higher-order primitives are interpreted using techniques from the semantics of quantitative models of linear

[4] $\mathcal{T}(H)$ and $\mathcal{B}(H)$ stand for the trace-class and bounded linear operators, respectively, on a Hilbert space H.

	Our work	**Qunity** [24]	**λ-$\mathcal{S}_1$** [9,10,1,8,2]	**Sym** [17]	**Quipper** [11]	**QWIRE** [15]	**Q.λ-calculus** [14]	[7,6,23]	[21]
Quantum control	✓	✓	✓	✓	✗	✗	✗	✗	✗
Classical control	✓	✓	✗	✗	✓	✓	✓	✓	✓
Infinite dimensional quantum type	✓	✗	✗	✓	✗	✗	✓	✓	✓
Unitary function spaces	✓	✗	✗	✓	✗	✓	✗	✗	✗
Normalised quantum terms	✓	✗	✗	✗	✓	✓	✓	✓	✓
Denotational semantics	✓	✓	✓	✗	✗	✓	✓	✓	✓
Adequacy	✓	✗	✓	✗	✗	✓	✓	✓	✓
Termination	✓		✓	✓	✗*	✓	✗*	✗*	✗*

Fig. 2: A comparison of some quantum programming languages.

logic. In [7,6,23], the author(s) work with QLCs whose denotational semantics is given via game semantics. More recently, in [21], the authors have shown how to interpret a QLC with recursive types using presheaves that are enriched over a suitable base category and they also prove full abstraction. The approach that we use in this paper is based on von Neumann algebras which have been previously used for the semantics of quantum programming languages [5,12,16]. In particular, if we disregard the interaction between the pure subsystem and the main calculus, then our denotational model for the main calculus coincides with the one in [5]. The reason that we choose von Neumann algebras over the other approaches is because we think this gives the model which is the closest to mathematical physics out of the ones mentioned. Furthermore, our primary focus is not on developing the semantics of the classical paradigm; instead it is on the *interaction* between the pure quantum control subsystem and the main calculus based on classical control. Another distinguishing feature of our system, compared to the ones above, is the addition of types such as $\mathcal{B}(\mathbf{qnat})$ which allow us to describe states in infinite-dimensional Hilbert spaces, such as $\ell^2(\mathbb{N})$, and which we do not think can be easily included using the other approaches.

For the quantum control paradigm, relevant work includes [9,10,1,8,2]. What all these approaches have in common is that they have specific terms in their syntax for representing superposition. However, these approaches do not ensure unitarity of the relevant function spaces, in general. Our approach, instead, is based on related work first described in [17] and then later in [3]. One of the most distinctive features of this approach to quantum control is that the terms that introduce superposition are subject to strict formation conditions that ensure (linear algebraic) normalisation of the corresponding vectors. Furthermore, this approach to quantum control also imposes strict admissibility criteria on the unitary function spaces that can be formed through the type system and this ensures unitarity of the relevant expressions. Because of these considerations, we choose this approach to model quantum control in our pure subsystem. However, we have made some changes, in particular, we replace the operational semantics from these works with an equational theory instead. This works better for

* These languages admit general recursion.

our approach to *combining* quantum and classical control within the quantum configurations (in QLCs the quantum data is considered modulo equality). We continue building on this development by describing a denotational semantics that is sound and complete with respect to the equational theory. This is the first such result for languages with quantum control.

Another paper which combines quantum and classical control is Qunity [24]. However, compared to our work, Qunity does not have an operational semantics, but instead there is a compilation procedure to quantum circuits. Furthermore, the denotational semantics of Qunity: (1) does not ensure unitarity and normalisation of the pure primitives in its language, whereas ours does; (2) does not ensure trace-preservation of the mixed primitives in the Schrödinger picture, whereas ours does indeed ensure unitality of the mixed primitives in the Heisenberg picture (which corresponds to trace-preservation under the Heisenberg-Schrödinger duality). Other differences include that Qunity is restricted to finite-dimensional quantum data and they do not have higher-order lambda abstractions for dealing with mixed-state primitives. Instead, Qunity relies on a try-catch mechanism to combine quantum and classical control, whereas our approach is based on quantum configurations. The table in Figure 2 gives a comparison of our work with other quantum programming languages. *Normalised quantum terms* refers to the property that terms representing pure/mixed quantum states always correspond to complex vectors with norm one/density operators with trace one. *Adequacy* refers to the property that two observationally distinct programs have distinct denotational interpretations.

2 Quantum Control

This part is adapted from one of the authors' PhD thesis [13, Chapter 3], to which we refer the readers for full detail and proofs.

We write $\mathbf{Unitary} \subseteq \mathbf{Isometry} \subseteq \mathbf{Hilb}$ for the (sub)categories whose objects are the Hilbert spaces and whose morphisms are the unitaries (resp. isometries, bounded linear maps) between them.

2.1 Pure Quantum Syntax

We begin by describing a subsystem for quantum control. The syntax is based on symmetric pattern-matching [17]. The *pure quantum types* are given by the following grammar: $\boldsymbol{Q} ::= \boldsymbol{I} \mid \boldsymbol{Q}_1 \otimes \boldsymbol{Q}_2 \mid \boldsymbol{Q}_1 \oplus \boldsymbol{Q}_2 \mid \mathbf{qnat}$.

We interpret every quantum type as a separable Hilbert space: the unit type is interpreted as $[\![\boldsymbol{I}]\!] \triangleq \mathbb{C}$; pair types are interpreted as tensor products; (quantum) sum types are interpreted as direct sums; finally, $[\![\mathbf{qnat}]\!] \triangleq \ell^2(\mathbb{N})$. The type of qubits may be defined as $\mathbf{qbit} \triangleq \boldsymbol{I} \oplus \boldsymbol{I}$. We write $\mathbf{qbit}^n$ for the n-fold tensor product $\mathbf{qbit} \otimes \cdots \otimes \mathbf{qbit}$. The Hilbert space $\ell^2(\mathbb{N})$ allows us to form superpositions with respect to a countable orthonormal basis (e.g., $1/\sqrt{2}\,|3\rangle + 1/\sqrt{2}\,|7\rangle$). We may think of $\ell^2(\mathbb{N})$ as a quantum analogue of the natural numbers, to abstract from qubit-level computation. The addition of **qnat** allows us to reason about pure quantum computation while abstracting away from quantum circuits.

$$
\begin{array}{rcl}
\boldsymbol{b} & ::= & \boldsymbol{x} \mid * \mid \mathbf{inl}\ \boldsymbol{b} \mid \mathbf{inr}\ \boldsymbol{b} \mid \boldsymbol{b}_1 \otimes \boldsymbol{b}_2 \mid \mathbf{0} \mid \mathbf{S}\ \boldsymbol{b} \\
\boldsymbol{v} & ::= & \textstyle\sum_{i\in\mathcal{I}} \boldsymbol{\alpha}_i \cdot \boldsymbol{b}_i \\
u & ::= & \{ \mid \boldsymbol{b}_1 \mapsto \boldsymbol{v}_1 \ \ldots \mid \boldsymbol{b}_n \mapsto \boldsymbol{v}_n \ \} \mid u_1 \oplus u_2 \mid u_1 \otimes u_2 \mid u_2 \circ u_1 \mid u^* \mid \mathrm{ctrl}\ u \\
\boldsymbol{t} & ::= & \boldsymbol{x} \mid * \mid \mathbf{inl}\ \boldsymbol{t} \mid \mathbf{inr}\ \boldsymbol{t} \mid \boldsymbol{t}_1 \otimes \boldsymbol{t}_2 \mid \mathbf{0} \mid \mathbf{S}\ \boldsymbol{t} \mid u\,\boldsymbol{t} \mid \textstyle\sum_{i\in\mathcal{I}} \boldsymbol{\alpha}_i \cdot \boldsymbol{t}_i
\end{array}
$$

Fig. 3: Term grammar for the pure quantum subsystem.

The *unitary types*, written as $U(\boldsymbol{Q}_1, \boldsymbol{Q}_2)$, represent the unitary function spaces of our language. They are not interpreted as Hilbert spaces, but as sets of unitary operators: $[\![U(\boldsymbol{Q}_1, \boldsymbol{Q}_2)]\!] \triangleq \{u \colon [\![\boldsymbol{Q}_1]\!] \to [\![\boldsymbol{Q}_2]\!] \mid u \textit{ is a unitary operator}\}$.

We now describe the language of the pure quantum subsystem whose grammar is given in Figure 3. (Pure) ***Terms*** $\boldsymbol{t}$ are written in bold notation in order to distinguish with other syntactic constructs. The symbols $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ denote pure quantum variables. We write $\mathbf{inl}_{\boldsymbol{Q}_1 \oplus \boldsymbol{Q}_2}\ \boldsymbol{t}$ and $\mathbf{inr}_{\boldsymbol{Q}_1 \oplus \boldsymbol{Q}_2}\ \boldsymbol{t}$ for sum type injections, and, for brevity, we often omit writing the subscripts. The term $\boldsymbol{t}_1 \otimes \boldsymbol{t}_2$ is used for forming pairs of terms. The terms $\mathbf{0}$ and $\mathbf{S}\ \boldsymbol{t}$ represent the qnat $|0\rangle$ and the qnat $|n+1\rangle$, provided that $\boldsymbol{t}$ represents the qnat $|n\rangle$. We define $|\underline{\mathbf{n}}\rangle \triangleq \mathbf{S^n}\ \mathbf{0}$ and $\left|\underline{\boldsymbol{t}+\mathbf{n}}\right\rangle \triangleq \mathbf{S^n}\ \boldsymbol{t}$, for $\mathbf{n} \in \mathbb{N}$. The term $u\,\boldsymbol{t}$ represents the application of a unitary u to a term $\boldsymbol{t}$. Superpositions are represented by the term $\sum_{i\in\mathcal{I}} \boldsymbol{\alpha}_i \cdot \boldsymbol{t}_i$. We impose strict conditions on these terms that ensure normalisation in §2.2.

In Figure 3, the ***Basis Terms*** (ranged over by symbols $\boldsymbol{b}, \boldsymbol{b}_i$) are the simplest kind of pure terms. The (pure) ***Values***, ranged over by $\boldsymbol{v}$ and defined in Figure 3, are the normal forms in this subsystem. They are of the form $\sum_{i\in\mathcal{I}} \boldsymbol{\alpha}_i \cdot \boldsymbol{b}_i$, where $\boldsymbol{\alpha}_i \in \mathbb{C}$, $\boldsymbol{b}_i$ are basis terms, and $\mathcal{I}$ is a finite index set. We consider $\mathcal{I}$ as a totally ordered set so that we can work with unique normal forms. We also write $\boldsymbol{\alpha}_1 \cdot \boldsymbol{b}_1 + \cdots + \boldsymbol{\alpha}_n \cdot \boldsymbol{b}_n$ for $\sum_{i\in\{1,\ldots,n\}} \boldsymbol{\alpha}_i \cdot \boldsymbol{b}_i$. The formation conditions (§2.2) ensure that all terms in the sums are pairwise distinct, orthogonal, and of the same type, and thus well-formed values correspond to *normalised* vectors. In our equational theory, every quantum term $\boldsymbol{t}$ can be rewritten to a normalised pure value $\boldsymbol{v}$ (Proposition 3). The values are the canonical forms of this subsystem.

Example 1. The basis term $* \colon \boldsymbol{I}$ represents the complex scalar $1 \in \mathbb{C}$. The basis terms that represent the computational basis for qubits can be defined by $|\mathbf{0}\rangle \triangleq \mathbf{inl}\ * \colon \mathbf{qbit}$ and $|\mathbf{1}\rangle \triangleq \mathbf{inr}\ * \colon \mathbf{qbit}$. Other basis terms that can be defined include $|\mathbf{00}\rangle \triangleq |\mathbf{0}\rangle \otimes |\mathbf{0}\rangle \colon \mathbf{qbit}^2$ and $|\mathbf{11}\rangle \triangleq |\mathbf{1}\rangle \otimes |\mathbf{1}\rangle \colon \mathbf{qbit}^2$. If $\boldsymbol{b}$ is a basis term, we often write it as $|\boldsymbol{b}\rangle$ to distinguish it from other (non-basis) terms, e.g., $|\underline{\mathbf{2}}\rangle \colon \mathbf{qnat}$ represents the qnat $|2\rangle \in \ell^2(\mathbb{N})$. Single-qubit states may be defined by $|\pm\rangle \triangleq (1/\sqrt{2} \cdot |\mathbf{0}\rangle \pm 1/\sqrt{2} \cdot |\mathbf{1}\rangle) : \mathbf{qbit}$. (Entangled) quantum states can be defined, e.g., $\mathbf{Bell} \triangleq (1/\sqrt{2} \cdot |\mathbf{00}\rangle + 1/\sqrt{2} \cdot |\mathbf{11}\rangle) : \mathbf{qbit}^2$. Linear combinations of qnats can also be written in the language, *i.e.* $(1/\sqrt{2} \cdot |\underline{\mathbf{0}}\rangle + 1/\sqrt{6} \cdot |\underline{\mathbf{1}}\rangle + 1/\sqrt{3} \cdot |\underline{\mathbf{2}}\rangle) : \mathbf{qnat}$.

Next, we describe the ***Unitaries*** u of Figure 3. *Unitary pattern-matching* $\{ \mid \boldsymbol{b}_1 \mapsto \boldsymbol{v}_1 \ \ldots \mid \boldsymbol{b}_n \mapsto \boldsymbol{v}_n \ \}$ allows us to build unitary maps out of basis terms and values: every *closed* basis term $\boldsymbol{b}_i$ is mapped to the closed value $\boldsymbol{v}_i$; basis terms with free variables $\boldsymbol{b}_i$ determine a mapping through the use of substitution with respect to the corresponding value $\boldsymbol{v}_i$, wherein both $\boldsymbol{b}_i$ and $\boldsymbol{v}_i$

have the same free variables occurring within them, and the induced mapping is determined by performing all possible substitutions of the free variables on both sides with the same closed basis terms. The formation conditions (§2.2) ensure that the basis terms $\boldsymbol{b}_i$ (resp. the values $\boldsymbol{v}_i$) determine an ONB for the type $\boldsymbol{Q}_1$ (resp. $\boldsymbol{Q}_2$) and therefore $\{ \mid \boldsymbol{b}_1 \mapsto \boldsymbol{v}_1 \ \ldots \mid \boldsymbol{b}_n \mapsto \boldsymbol{v}_n \}$ determines a unitary map. We introduce syntactic sugar for quantum if statements, written **qif**. If u_1 is of the form $\{ \mid \boldsymbol{b}_1 \mapsto \boldsymbol{v}_1 \ \ldots \mid \boldsymbol{b}_n \mapsto \boldsymbol{v}_n \}$ and u_2 of the form $\{ \mid \boldsymbol{b}'_1 \mapsto \boldsymbol{v}'_1 \ \ldots \mid \boldsymbol{b}'_m \mapsto \boldsymbol{v}'_m \}$ then **qif** $\boldsymbol{x}$ **then** $\boldsymbol{x} \otimes u_2$ **else** $\boldsymbol{x} \otimes u_1$ is short for the unitary:

$$\left\{ \begin{array}{ll} |\mathbf{0}\rangle \otimes \boldsymbol{b}_1 \mapsto |\mathbf{0}\rangle \otimes \boldsymbol{v}_1 & |\mathbf{1}\rangle \otimes \boldsymbol{b}'_1 \mapsto |\mathbf{1}\rangle \otimes \boldsymbol{v}'_1 \\ \quad\vdots & \quad\vdots \\ |\mathbf{0}\rangle \otimes \boldsymbol{b}_n \mapsto |\mathbf{0}\rangle \otimes \boldsymbol{v}_n & |\mathbf{1}\rangle \otimes \boldsymbol{b}'_m \mapsto |\mathbf{1}\rangle \otimes \boldsymbol{v}'_m \end{array} \right\}$$

Example 2. The unitaries Had and CNOT below encode the standard Hadamard and CNOT gates, respectively. CNOT makes use of some simple pattern-matching on its last line: it can match either with $|\mathbf{0}\rangle \otimes |\mathbf{0}\rangle$ or with $|\mathbf{0}\rangle \otimes |\mathbf{1}\rangle$. The unitary $\mathrm{Had}_{\mathbf{qnat}}$ defines an extension of the Hadamard gate on the space $\ell^2(\mathbb{N})$.

$$\mathrm{Had} : U(\mathbf{qbit}, \mathbf{qbit})$$
$$\mathrm{Had} \triangleq \left\{ \begin{array}{l} \mid |\mathbf{0}\rangle \mapsto {}^1\!/\!_{\sqrt{2}} \cdot |\mathbf{0}\rangle + {}^1\!/\!_{\sqrt{2}} \cdot |\mathbf{1}\rangle \\ \mid |\mathbf{1}\rangle \mapsto {}^1\!/\!_{\sqrt{2}} \cdot |\mathbf{0}\rangle - {}^1\!/\!_{\sqrt{2}} \cdot |\mathbf{1}\rangle \end{array} \right\}$$

$$\mathrm{CNOT} : U(\mathbf{qbit}^2, \mathbf{qbit}^2)$$
$$\mathrm{CNOT} \triangleq \left\{ \begin{array}{l} \mid |\mathbf{1}\rangle \otimes |\mathbf{0}\rangle \mapsto |\mathbf{1}\rangle \otimes |\mathbf{1}\rangle \\ \mid |\mathbf{1}\rangle \otimes |\mathbf{1}\rangle \mapsto |\mathbf{1}\rangle \otimes |\mathbf{0}\rangle \\ \mid |\mathbf{0}\rangle \otimes |\boldsymbol{x}\rangle \mapsto |\mathbf{0}\rangle \otimes |\boldsymbol{x}\rangle \end{array} \right\}$$

$$\mathrm{Had}_{\mathbf{qnat}} : U(\mathbf{qnat}, \mathbf{qnat})$$
$$\mathrm{Had}_{\mathbf{qnat}} \triangleq \left\{ \begin{array}{ll} \mid |\underline{\mathbf{0}}\rangle & \mapsto {}^1\!/\!_{\sqrt{2}} \cdot |\underline{\mathbf{0}}\rangle + {}^1\!/\!_{\sqrt{2}} \cdot |\underline{\mathbf{1}}\rangle \\ \mid |\underline{\mathbf{1}}\rangle & \mapsto {}^1\!/\!_{\sqrt{2}} \cdot |\underline{\mathbf{0}}\rangle - {}^1\!/\!_{\sqrt{2}} \cdot |\underline{\mathbf{1}}\rangle \\ \mid |\underline{\boldsymbol{x}+\mathbf{2}}\rangle & \mapsto |\underline{\boldsymbol{x}+\mathbf{2}}\rangle \end{array} \right\}$$

The remaining expressions for forming unitaries are easy to understand: $u_2 \circ u_1$ represents composition of unitaries, u^* represents the adjoint of a unitary, $u_1 \otimes u_2$ and $u_1 \oplus u_2$ represent tensor products and direct sums of unitaries, and finally, ctrl u represents a qubit-controlled unitary operator. For simplicity, we sometimes write $\{\ldots \mid \boldsymbol{b}_i \mapsto \boldsymbol{v}_i \ldots\}$ as a shorthand notation for the unitary $\{ \mid \boldsymbol{b}_1 \mapsto \boldsymbol{v}_1 \ \ldots \mid \boldsymbol{b}_n \mapsto \boldsymbol{v}_n \}$, when n is clear from context or unimportant. We also point out that unitaries depend only on values and basis terms and that general pure terms cannot be used for the construction of unitaries.

2.2 Formation Conditions

In order to guarantee (linear-algebraic) normalisation for quantum terms and unitarity for functions, we rely on formulating: (1) a suitable *orthogonality relation* $\perp \subseteq \boldsymbol{Terms} \times \boldsymbol{Terms}$, which is used as part of the formation condition of some terms; (2) additional predicates $\mathrm{ONB}_{\boldsymbol{Q}}$ and $\mathrm{ONB}^{val}_{\boldsymbol{Q}}$ which ensure that a set of basis terms/values of type $\boldsymbol{Q}$ determines an ONB which is used as part of the formation conditions for the atomic introduction rule of unitaries.

The relation $\perp$ is defined as the smallest binary relation $\perp \subseteq \boldsymbol{Terms} \times \boldsymbol{Terms}$ that is closed under the rules in Figure 4. Quantum terms that are orthogonal via this relation are orthogonal in a denotational sense as well (§2.3).

$$\frac{}{\mathbf{inl}\ \boldsymbol{t}_1 \perp \mathbf{inr}\ \boldsymbol{t}_2} \qquad \frac{}{\mathbf{0} \perp \mathbf{S}\ \boldsymbol{t}} \qquad \frac{\boldsymbol{t}_1 \perp \boldsymbol{t}_2}{u\,\boldsymbol{t}_1 \perp u\,\boldsymbol{t}_2} \qquad \frac{j \neq k.\ \boldsymbol{t}_j \perp \boldsymbol{t}_k \qquad \sum_{i \in \mathcal{J} \cup \mathcal{K}} \overline{\boldsymbol{\alpha}_i}\boldsymbol{\beta}_i = 0}{\sum_{j \in \mathcal{J}}(\boldsymbol{\alpha}_j \cdot \boldsymbol{t}_j) \perp \sum_{k \in \mathcal{K}}(\boldsymbol{\beta}_k \cdot \boldsymbol{t}_k)}$$

Fig. 4: Rules for the orthogonality relation (excerpt).

$$\frac{}{\mathrm{ONB}_{\boldsymbol{Q}}(\{\boldsymbol{x}\})} \qquad \frac{}{\mathrm{ONB}_{\boldsymbol{I}}(\{*\})} \qquad \frac{\mathrm{ONB}_{\boldsymbol{Q}_1}(S) \qquad \mathrm{ONB}_{\boldsymbol{Q}_2}(T)}{\mathrm{ONB}_{\boldsymbol{Q}_1 \oplus \boldsymbol{Q}_2}(\{\mathbf{inl}\ \boldsymbol{b} \mid \boldsymbol{b} \in S\} \cup \{\mathbf{inr}\ \boldsymbol{b} \mid \boldsymbol{b} \in T\})}$$

$$\frac{\mathrm{ONB}_{\boldsymbol{Q}_i}(\pi_i(S)) \qquad \forall \boldsymbol{b} \in \pi_i(S),\ \mathrm{ONB}_{\boldsymbol{Q}_j}(S^i_{\boldsymbol{b}})}{\mathrm{ONB}_{\boldsymbol{Q}_1 \otimes \boldsymbol{Q}_2}(S)}\ i,j \in \{1,2\}, i \neq j$$

$$\frac{\mathrm{ONB}_{\mathbf{qnat}}(S)}{\mathrm{ONB}_{\mathbf{qnat}}(\{\mathbf{0}\} \cup \{\mathbf{S}\ \boldsymbol{b} \mid \boldsymbol{b} \in S\})} \qquad \frac{(\boldsymbol{\alpha}_{\boldsymbol{b},\boldsymbol{b}'})_{(\boldsymbol{b},\boldsymbol{b}') \in S \times S} \text{ is a unitary matrix} \quad \mathrm{ONB}_{\boldsymbol{Q}}(S)}{\mathrm{ONB}^{val}_{\boldsymbol{Q}}(\{\sum^*_{\boldsymbol{b}' \in S}\ \boldsymbol{\alpha}_{\boldsymbol{b},\boldsymbol{b}'} \cdot \boldsymbol{b}' \mid \boldsymbol{b} \in S\})}$$

Fig. 5: Orthonormal basis predicates.

This relation allows us to enforce the orthogonality of terms appearing within a superposition $\sum_{i \in \mathcal{I}}(\boldsymbol{\alpha}_i \cdot \boldsymbol{t}_i)$. Under the assumption that all terms $\boldsymbol{t}_i$ represent normalised vectors that are pairwise orthogonal, the norm of the superposition is easy to calculate (statically), and it is given by $\sum_{i \in \mathcal{I}} |\boldsymbol{\alpha}_i|^2$. This justifies the formulation of the introduction rule for superposition (Figure 6) and guarantees that the vector represented by this superposition of terms is normalised.

Example 3. The rules for the orthogonality relation are not restricted to closed quantum terms, e.g., $\mathbf{0} \perp \mathbf{S}\ \boldsymbol{x}$ and $\mathbf{inl}\ \boldsymbol{x} \perp \mathbf{inr}\ \boldsymbol{y}$. One can also show that $|+\rangle \perp |-\rangle$ (defined in Example 1). However, we cannot show that Had $|\mathbf{0}\rangle \perp |-\rangle$ even though Had $|\mathbf{0}\rangle$ is equal to $|+\rangle$ with respect to our equational logic and also with respect to our denotational semantics (both introduced later). The reason for this is that we wish to define the orthogonality relation *statically*.

In order to determine whether a finite set S of (potentially open) basis terms determines an ONB for a type $\boldsymbol{Q}$, we introduce the *orthonormal basis predicates* $\mathrm{ONB}_{\boldsymbol{Q}}(S)$ in Figure 5, originally known as *orthogonal decomposition* [17]. Note that there is a mistake with the rule for the tensor product in the original paper, *e.g.* it does not allow for the typing of a CNOT.

Given a type $\boldsymbol{Q}$ and a finite set of basis terms S, we say that S determines an *orthonormal basis of* $\boldsymbol{Q}$, which we write as $\mathrm{ONB}_{\boldsymbol{Q}}(S)$, if this can be derived via the rules in Figure 5. We now explain some of the notation used therein. If S is a set of the form $\{\boldsymbol{b}_1 \otimes \boldsymbol{b}'_1, \ldots, \boldsymbol{b}_n \otimes \boldsymbol{b}'_m\}$ then we define $\pi_1(S) \triangleq \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ and $\pi_2(S) \triangleq \{\boldsymbol{b}'_1, \ldots, \boldsymbol{b}'_m\}$. In this situation, we also define $S^1_{\boldsymbol{b}} \triangleq \{\boldsymbol{b}' \mid \boldsymbol{b} \otimes \boldsymbol{b}' \in S\}$ and $S^2_{\boldsymbol{b}'} \triangleq \{\boldsymbol{b} \mid \boldsymbol{b} \otimes \boldsymbol{b}' \in S\}$. Furthermore, the ONB predicate implies pairwise orthogonality of the associated values. Note that the predicate $\mathrm{ONB}_{\boldsymbol{Q}}(S)$ is defined for a finite set S consisting of *basis* terms. But, in order to be able to define unitaries that make use of superposition, we also work with other ONBs. Given a finite set S consisting of *values*, we say that S determines an ONB of type $\boldsymbol{Q}$, and we write $\mathrm{ONB}^{val}_{\boldsymbol{Q}}(S)$ to indicate this, whenever this can be derived via the rules in Figure 5. We also introduce syntactic sugar: the value $\sum^*_{i \in \mathcal{I}} \boldsymbol{\alpha}_i \cdot \boldsymbol{b}_i$ is

$$\frac{\Vdash u\colon U(\boldsymbol{Q}_1,\boldsymbol{Q}_2) \quad \boldsymbol{\Gamma}\vdash \boldsymbol{t}\colon \boldsymbol{Q}_1}{\boldsymbol{\Gamma}\vdash u\,\boldsymbol{t}\colon \boldsymbol{Q}_2} \qquad \frac{\forall k,\ \boldsymbol{\Gamma}\vdash \boldsymbol{t}_k\colon \boldsymbol{Q} \quad \forall k\neq j,\ \boldsymbol{t}_k\perp \boldsymbol{t}_j \quad \sum_{i\in\mathcal{I}}|\boldsymbol{\alpha}_i|^2=1}{\boldsymbol{\Gamma}\vdash \sum_{i\in\mathcal{I}}\boldsymbol{\alpha}_i\cdot\boldsymbol{t}_i\colon \boldsymbol{Q}}$$

$$\frac{\begin{array}{l}\forall i,\ \boldsymbol{\Gamma}_i\vdash \boldsymbol{b}_i\colon \boldsymbol{Q}_1 \quad \mathrm{ONB}_{\boldsymbol{Q}_1}\{\boldsymbol{b}_1,\dots,\boldsymbol{b}_n\}\\ \forall i,\ \boldsymbol{\Gamma}_i\vdash \boldsymbol{v}_i\colon \boldsymbol{Q}_2 \quad \mathrm{ONB}^{val}_{\boldsymbol{Q}_2}\{\boldsymbol{v}_1,\dots,\boldsymbol{v}_n\}\end{array}}{\Vdash \{\ |\ \boldsymbol{b}_1\mapsto\boldsymbol{v}_1\ \dots\ |\ \boldsymbol{b}_n\mapsto\boldsymbol{v}_n\ \}\colon U(\boldsymbol{Q}_1,\boldsymbol{Q}_2)} \qquad \frac{\Vdash u_1\colon U(\boldsymbol{Q}_1,\boldsymbol{Q}_2) \quad \Vdash u_2\colon U(\boldsymbol{Q}_3,\boldsymbol{Q}_4)}{\Vdash u_1\oplus u_2\colon U(\boldsymbol{Q}_1\oplus\boldsymbol{Q}_3,\boldsymbol{Q}_2\oplus\boldsymbol{Q}_4)}$$

$$\frac{\Vdash u_1\colon U(\boldsymbol{Q}_1,\boldsymbol{Q}_2) \quad \Vdash u_2\colon U(\boldsymbol{Q}_3,\boldsymbol{Q}_4)}{\Vdash u_1\otimes u_2\colon U(\boldsymbol{Q}_1\otimes\boldsymbol{Q}_3,\boldsymbol{Q}_2\otimes\boldsymbol{Q}_4)} \qquad \frac{\Vdash u_1\colon U(\boldsymbol{Q}_1,\boldsymbol{Q}_2) \quad \Vdash u_2\colon U(\boldsymbol{Q}_2,\boldsymbol{Q}_3)}{\Vdash u_2\circ u_1\colon U(\boldsymbol{Q}_1,\boldsymbol{Q}_3)}$$

$$\frac{\Vdash u\colon U(\boldsymbol{Q}_1,\boldsymbol{Q}_2)}{\Vdash u^*\colon U(\boldsymbol{Q}_2,\boldsymbol{Q}_1)} \qquad \frac{\Vdash u\colon U(\boldsymbol{Q},\boldsymbol{Q})}{\Vdash \mathrm{ctrl}\ u\colon U(\mathbf{qbit}\otimes\boldsymbol{Q},\mathbf{qbit}\otimes\boldsymbol{Q})}$$

Fig. 6: Formation rules for terms and unitaries (excerpt).

the finite linear combination where elements with scalar 0 are omitted. In other words, the indices $i\in\mathcal{I}$ such that $\boldsymbol{\alpha}_i=0$ are ignored, e.g., if $\boldsymbol{\alpha}_1=1$ and $\boldsymbol{\alpha}_2=0$, $\sum^*_{i\in\{1,2\}}\boldsymbol{\alpha}_i\cdot\boldsymbol{b}_i$ is the value $\sum_{i\in\{*\}}1\cdot\boldsymbol{b}_1$.

Example 4. For **qbit**, the rules from Figure 5 show that $\mathrm{ONB}_{\mathbf{qbit}}(\{|\mathbf{0}\rangle,|\mathbf{1}\rangle\})$ and that $\mathrm{ONB}^{val}_{\mathbf{qbit}}(\{|+\rangle,|-\rangle\})$ hold (with $|\pm\rangle$ defined in Example 1). We also have that $\mathrm{ONB}_{\mathbf{qbit}}(\{\boldsymbol{x}\})$, where $\boldsymbol{x}$ is a variable. In other words, we determine an ONB for the type **qbit** by substituting $\boldsymbol{x}$ with all possible closed computational *basis* terms of type **qbit** (see Figure 3). For **qnat**, any set S with the property that $\mathrm{ONB}_{\mathbf{qnat}}(S)$ holds, must contain a term that is not closed. We have that $\mathrm{ONB}_{\mathbf{qnat}}(\{|\underline{\mathbf{x}}\rangle\})$, $\mathrm{ONB}_{\mathbf{qnat}}(\{|\underline{\mathbf{0}}\rangle,|\underline{\boldsymbol{x}+\mathbf{1}}\rangle\})$, and $\mathrm{ONB}_{\mathbf{qnat}}(\{|\underline{\mathbf{0}}\rangle,|\underline{\mathbf{1}}\rangle,|\underline{\boldsymbol{x}+\mathbf{2}}\rangle\})$ hold true and determine the same ONB for the type **qnat** (it represents the computational basis of $\ell^2(\mathbb{N})$). The set used in the last predicate allows us to conclude that $\mathrm{ONB}^{val}_{\mathbf{qnat}}\big(\{{}^1\!/\!_{\sqrt{2}}\cdot|\underline{\mathbf{0}}\rangle+{}^1\!/\!_{\sqrt{2}}\cdot|\underline{\mathbf{1}}\rangle,{}^1\!/\!_{\sqrt{2}}\cdot|\underline{\mathbf{0}}\rangle-{}^1\!/\!_{\sqrt{2}}\cdot|\underline{\mathbf{1}}\rangle,|\underline{\boldsymbol{x}+\mathbf{2}}\rangle\}\big)$ holds. This last set of values is used to type $\mathrm{Had}_{\mathbf{qnat}}$ from Example 2.

Judgements for pure quantum terms have the form $\boldsymbol{\Gamma}\vdash\boldsymbol{t}\colon\boldsymbol{Q}$ where $\boldsymbol{\Gamma}$ stands for a linear context of the form $\boldsymbol{\Gamma}\triangleq\boldsymbol{x}_1\colon\boldsymbol{Q}_1,\dots,\boldsymbol{x}_n\colon\boldsymbol{Q}_n$. The formation rules for terms are presented in Figure 6; the first two lines are standard for a linearly-typed term calculus. The rule for superposition depends on the orthogonality relation $\perp$, and the premise ensures that the resulting superposition corresponds to a normalised vector. This is made precise later (Theorem 1). Finally, the formation rule for $u\,\boldsymbol{t}$, i.e. the application of a unitary u to a term $\boldsymbol{t}$, is akin to function application in lambda-calculi. However, the complexity is primarily relegated to the judgement $\Vdash u\colon U(\boldsymbol{Q}_1,\boldsymbol{Q}_2)$, which ensures the unitarity of u.

The formation rules for unitaries are presented in Figure 6. In the introduction rule for atomic unitaries $\{\ |\ \boldsymbol{b}_1\mapsto\boldsymbol{v}_1\ \dots\ |\ \boldsymbol{b}_n\mapsto\boldsymbol{v}_n\ \}$, we have that the potentially open basis terms $\{\boldsymbol{b}_1,\dots,\boldsymbol{b}_n\}$ (resp. values $\{\boldsymbol{v}_1,\dots,\boldsymbol{v}_n\}$) determine an ONB for $\boldsymbol{Q}_1$ (resp. $\boldsymbol{Q}_2$) in the sense explained above. Then the resulting bijective mapping that associates $\boldsymbol{b}_i$ to $\boldsymbol{v}_i$ uniquely determines a unitary. The formation rules for the remaining unitaries are straightforward, and they align easily with the mathematical intuition that motivates them. We also wish to note that unitaries do not depend on general terms (see Figure 3).

2.3 Denotational Semantics for Pure Quantum System

Recall (§2.1) that the interpretation of a pure quantum type $\boldsymbol{Q}$ is a Hilbert space $[\![\boldsymbol{Q}]\!]$, i.e., an object of the category **Isometry**, and the interpretation of a unitary type $U(\boldsymbol{Q}_1, \boldsymbol{Q}_2)$ is the set $[\![U(\boldsymbol{Q}_1, \boldsymbol{Q}_2)]\!]$ of unitaries in $[\![\boldsymbol{Q}_1]\!] \to [\![\boldsymbol{Q}_2]\!]$.

A linear context is interpreted in the usual way $[\![\boldsymbol{\Gamma}]\!] \triangleq [\![\boldsymbol{Q}_1]\!] \otimes \cdots \otimes [\![\boldsymbol{Q}_n]\!]$. The interpretation of a judgement $\boldsymbol{\Gamma} \vdash \boldsymbol{t} \colon \boldsymbol{Q}$ is given by a morphism $[\![\boldsymbol{\Gamma} \vdash \boldsymbol{t} \colon \boldsymbol{Q}]\!] \in \mathbf{Isometry}([\![\boldsymbol{\Gamma}]\!], [\![\boldsymbol{Q}]\!])$. We also sometimes abuse notation and write $[\![\boldsymbol{t}]\!]$ for brevity instead of $[\![\boldsymbol{\Gamma} \vdash \boldsymbol{t} \colon \boldsymbol{Q}]\!]$. For closed quantum terms $\cdot \vdash \boldsymbol{t} \colon \boldsymbol{Q}$, we interpret the empty context as $\mathbb{C}$ (the tensor unit in **Isometry**) and so we can identify $[\![\cdot \vdash \boldsymbol{t} \colon \boldsymbol{Q}]\!](1)$ with a normalised vector.

Every value $\boldsymbol{v}$ is of the form $\sum_{i \in \mathcal{I}} \boldsymbol{\alpha}_i \cdot \boldsymbol{b}_i$, but the sum is subject to strict admissibility conditions imposed by the orthogonality relation $\perp$ from §2.2. The denotational significance of this relation is provided by the statement: given $\boldsymbol{\Gamma}_1 \vdash \boldsymbol{t}_1 \colon \boldsymbol{Q}$ and $\boldsymbol{\Gamma}_2 \vdash \boldsymbol{t}_2 \colon \boldsymbol{Q}$, if $\boldsymbol{t}_1 \perp \boldsymbol{t}_2$ then $[\![\boldsymbol{t}_1]\!]^* [\![\boldsymbol{t}_2]\!] = 0_{[\![\boldsymbol{\Gamma}_2]\!], [\![\boldsymbol{\Gamma}_1]\!]}$; which is central in proving that the interpretation of values is well-defined as isometries.

The most challenging unitaries to interpret are the unitary pattern-matchings, constructed through the use of basis terms and values (but not general terms). Another key role is played by the $\mathrm{ONB}_{\boldsymbol{Q}}$ and $\mathrm{ONB}^{val}_{\boldsymbol{Q}}$ predicates: they can be understood as determining suitable ONBs of the corresponding types. The denotational significance of these predicates is that they give us a resolution of the identity at type $\boldsymbol{Q}$, as made precise by the following proposition.

Proposition 1. *Let* $\boldsymbol{\Gamma}_1 \vdash \boldsymbol{v}_1 \colon \boldsymbol{Q}, \ldots, \boldsymbol{\Gamma}_n \vdash \boldsymbol{v}_n \colon \boldsymbol{Q}$ *be well-formed values. If* $\mathrm{ONB}^{\mathrm{val}}_{\boldsymbol{Q}}\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$, *then* $\mathrm{id}_{[\![\boldsymbol{Q}]\!]} = \sum_{i=1}^{n} [\![\boldsymbol{v}_i]\!] \circ [\![\boldsymbol{v}_i]\!]^*$.

Pattern-matching is then interpreted by $[\![\Vdash \{\ldots \mid \boldsymbol{b}_i \mapsto \boldsymbol{v}_i \ldots\}]\!] \triangleq \sum_i [\![\boldsymbol{v}_i]\!] \circ [\![\boldsymbol{b}_i]\!]^*$, and the previous proposition ensures it is unitary. Finally, we can show that the interpretation of all pure quantum terms is well-defined as isometries.

Theorem 1. *If* $\boldsymbol{\Gamma} \vdash \boldsymbol{t} \colon \boldsymbol{Q}$, *then* $[\![\boldsymbol{t}]\!] \in \mathbf{Isometry}([\![\boldsymbol{\Gamma}]\!], [\![\boldsymbol{Q}]\!])$.

Moreover, any canonical symmetric monoidal isomorphism can be represented by the unitaries of our language. The canonical symmetric monoidal isomorphisms are simply those that can be defined using the structure of a symmetric monoidal category only, *i.e.* isomorphisms that are given by: left and right unitors, symmetric swaps, associators, closed under composition and tensor products. We use this result later for quantum configurations.

Proposition 2. *Let* $\boldsymbol{Q}_1$ *and* $\boldsymbol{Q}_2$ *be two pure types, and let* $f \colon [\![\boldsymbol{Q}_1]\!] \to [\![\boldsymbol{Q}_2]\!]$ *be a canonical symmetric monoidal isomorphism. Then, there exists a unitary* $\Vdash u \colon U(\boldsymbol{Q}_1, \boldsymbol{Q}_2)$ *such that* $[\![u]\!] = f$.

2.4 Equational Theory

We now describe the equational theory of the pure quantum subsystem. Given two pure quantum terms $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$, we write $\boldsymbol{\Gamma} \vdash \boldsymbol{t}_1 = \boldsymbol{t}_2 \colon \boldsymbol{Q}$ to indicate that $\boldsymbol{t}_1$ equals $\boldsymbol{t}_2$ as terms of type $\boldsymbol{Q}$ in context $\boldsymbol{\Gamma}$. As usual if $\boldsymbol{\Gamma} \vdash \boldsymbol{t}_1 = \boldsymbol{t}_2 \colon \boldsymbol{Q}$, the rules imply that $\boldsymbol{\Gamma} \vdash \boldsymbol{t}_1 \colon \boldsymbol{Q}$ and $\boldsymbol{\Gamma} \vdash \boldsymbol{t}_2 \colon \boldsymbol{Q}$. We now explain some of the more

interesting rules in greater detail. The rules of the equational theory are divided into four groups. The first one is concerned with the usual rules for reflexivity, symmetry and transitivity. Secondly, we provide rules to handle linear algebraic identities, such as exchanging sums and the linearity of injections and unitaries; this also contains the rule $\boldsymbol{\Gamma} \vdash \sum_{i=1}^{1} 1 \cdot \boldsymbol{t} = \boldsymbol{t} : \boldsymbol{Q}$. A small excerpt is found below.

$$\frac{\boldsymbol{\Gamma} \vdash \sum_i \boldsymbol{\alpha}_i \cdot \left(\sum_j \boldsymbol{\beta}_{i,j} \cdot \boldsymbol{t}_j \right) : \boldsymbol{Q}}{\boldsymbol{\Gamma} \vdash \sum_i \boldsymbol{\alpha}_i \cdot \left(\sum_j \boldsymbol{\beta}_{i,j} \cdot \boldsymbol{t}_j \right) = \sum_j \left(\sum_i \boldsymbol{\alpha}_i \boldsymbol{\beta}_{i,j} \right) \cdot \boldsymbol{t}_j : \boldsymbol{Q}} \qquad \frac{\Vdash u : U(\boldsymbol{Q}_1, \boldsymbol{Q}_2) \qquad \boldsymbol{\Gamma} \vdash \sum_i \boldsymbol{\alpha}_i \cdot \boldsymbol{t}_i : \boldsymbol{Q}_1}{\boldsymbol{\Gamma} \vdash u \left(\sum_i \boldsymbol{\alpha}_i \cdot \boldsymbol{t}_i \right) = \sum_i \boldsymbol{\alpha}_i \cdot (u\, \boldsymbol{t}_i) : \boldsymbol{Q}_2}$$

Congruence rules comprise the third group, e.g., $\boldsymbol{\Gamma} \vdash \mathbf{S}\, \boldsymbol{t}_1 = \mathbf{S}\, \boldsymbol{t}_2 : \mathbf{qnat}$ given that $\boldsymbol{\Gamma} \vdash \boldsymbol{t}_1 = \boldsymbol{t}_2 : \mathbf{qnat}$. Last but not least are the *computational* rules, which, read left to right, can be seen as providing an operational semantics. The most important rule and our version of β-reduction, is the rule for unitary pattern-matching applied to a basis term.

$$\frac{\cdot \vdash \boldsymbol{b}' : \boldsymbol{Q}_1 \qquad \Vdash \{ \mid \boldsymbol{b}_1 \mapsto \boldsymbol{v}_1 \;\; \ldots \mid \boldsymbol{b}_n \mapsto \boldsymbol{v}_n \;\} : U(\boldsymbol{Q}_1, \boldsymbol{Q}_2) \qquad \mathtt{s}(\boldsymbol{b}_j) = \boldsymbol{b}'}{\cdot \vdash \{ \mid \boldsymbol{b}_1 \mapsto \boldsymbol{v}_1 \;\; \ldots \mid \boldsymbol{b}_n \mapsto \boldsymbol{v}_n \;\}\, \boldsymbol{b}' = \mathtt{s}(v_j) : \boldsymbol{Q}_2}$$

The formation conditions on unitary pattern-matching ensure that $\mathrm{ONB}_{\boldsymbol{Q}_1}(\{\boldsymbol{b}_i\}_i)$. With the latter, and given $\cdot \vdash \boldsymbol{b}' : \boldsymbol{Q}_1$, there exists a unique substitution $\mathtt{s}$ and a unique $\boldsymbol{b}_j$ such that $\mathtt{s}(\boldsymbol{b}_j) = \boldsymbol{b}'$. The resulting term of the rule is then the same substitution applied to the corresponding output $\boldsymbol{v}_j$. Such substitutions were first defined in [17], and can be deduced via an inference system.

Proposition 3. *Given $\cdot \vdash \boldsymbol{t} : \boldsymbol{Q}$, there is a unique value $\boldsymbol{v}$ s.t. $\cdot \vdash \boldsymbol{t} = \boldsymbol{v} : \boldsymbol{Q}$.*

This proposition shows that pure terms have a unique *normal form*. It is crucial for the proof of the completeness theorem, and comes in handy in the rest of the paper, especially to define the reduction system for the main calculus (§3).

Theorem 2 (Completeness). *$\cdot \vdash \boldsymbol{t}_1 = \boldsymbol{t}_2 : \boldsymbol{Q}$ iff $[\![\cdot \vdash \boldsymbol{t}_1 : \boldsymbol{Q}]\!] = [\![\cdot \vdash \boldsymbol{t}_2 : \boldsymbol{Q}]\!]$.*

3 Combining Classical and Quantum Control

This section lays down the syntax and semantics of the main calculus, which combines the quantum control fragment with a classically controlled layer, a variant of a call-by-value linear lambda-calculus. The subsystem which we presented in the previous section is designed to represent pure quantum information and computation, whereas in this section, we are concerned with mixed quantum information and computation. Our denotational approach towards this uses von Neumann algebras, which provide an appropriate mathematical setting to achieve this treatment. Operationally, this is achieved by adapting the configurations (also called closures) $(|\psi\rangle, \ell, M)$ from an effectful quantum lambda-calculus, e.g., [5,20], by replacing the quantum state $|\psi\rangle$ with a pure term $\boldsymbol{t}$, replacing the linking function ℓ with a unitary u that can be described via our syntax and which represents a suitable permutation, and finally by defining suitable formation conditions for the configuration.

$$
\begin{array}{lll}
\text{(Vars)} & x, y, z & \\
\text{(Types)} & \mathrm{A}, \mathrm{B} & ::= \mathrm{I} \mid \mathrm{A} + \mathrm{B} \mid \mathrm{A} \otimes \mathrm{B} \mid !\mathrm{A} \mid \mathrm{A} \multimap \mathrm{B} \mid \mathrm{Nat} \mid \mathcal{B}(\boldsymbol{Q}) \\
\text{(Terms)} & L, M, N & ::= * \mid x \mid \mathit{inl}(M) \mid \mathit{inr}(N) \mid \mathit{case}\, L\, \mathit{of}\, \{\mathit{inl}(x) \to M,\ \mathit{inr}(y) \to N\} \\
 & & \mid M \otimes N \mid \mathit{let}\ x \otimes y = M\ \mathit{in}\ N \mid \mathit{lift}(M) \mid \mathit{force}(M) \\
 & & \mid \lambda x.M \mid M\, N \\
 & & \mid \mathit{zero} \mid \mathit{succ}(M) \mid \mathit{match}\, L\, \mathit{with}\, \{\mathit{zero} \to M,\ \mathit{succ}(x) \to N\} \\
 & & \mid \mathit{pure}(\boldsymbol{t}) \mid \mathit{meas}(M) \mid \mathcal{B}(u)(M) \\
 & & \mid \mathit{let}\ \mathcal{B}(z) = M\ \mathit{in}\ N \mid \mathit{let}\ \mathcal{B}(x \otimes y) = M\ \mathit{in}\ N \\
\text{(Values)} & V, W & ::= * \mid x \mid \mathit{inl}(V) \mid \mathit{inr}(W) \mid V \otimes W \mid \mathit{lift}(M) \mid \lambda x.M \\
 & & \mid \mathit{zero} \mid \mathit{succ}(V)
\end{array}
$$

Fig. 7: Syntax of the classically controlled system.

3.1 Syntax and Typing Rules

The *types* of the main calculus, ranged over by capital Latin letters (e.g., A, B) are given in Figure 7. All types except Nat and $\mathcal{B}(\boldsymbol{Q})$ are standard in a system for a classical call-by-value linear lambda-calculus. Nat is a ground type representing classical natural numbers and $\mathcal{B}(\boldsymbol{Q})$ is the type that represents *mixed state* quantum computation on the Hilbert space determined by the type $\boldsymbol{Q}$. As we see later, this type plays an important role by introducing quantum and probabilistic effects into the system. The modality $\mathcal{B}$ is inspired from the denotational model, where $\mathcal{B}(H)$ represents the von Neumann algebra of bounded operators on the Hilbert space H.

Terms (and values) of the calculus are described in Figure 7. The term $\mathit{pure}(\boldsymbol{t})$ models the preparation of a pure quantum state represented by the quantum term $\boldsymbol{t}$. Measurement is modelled by the term $\mathit{meas}(M)$. The term $\mathcal{B}(u)(M)$ represents the application of the unitary u to a term M. Additionally, we have two syntactic constructs to model the isomorphism between the types $\mathcal{B}(\boldsymbol{Q}_1 \otimes \boldsymbol{Q}_2)$ and $\mathcal{B}(\boldsymbol{Q}_1) \otimes \mathcal{B}(\boldsymbol{Q}_2)$: the term $\mathit{let}\ \mathcal{B}(z) = M\ \mathit{in}\ N$ allows one to construct a term of type $\mathcal{B}(\boldsymbol{Q}_1 \otimes \boldsymbol{Q}_2)$ from a term of type $\mathcal{B}(\boldsymbol{Q}_1) \otimes \mathcal{B}(\boldsymbol{Q}_2)$ and the term $\mathit{let}\, \mathcal{B}(x \otimes y) = M\ \mathit{in}\ N$ deals with the opposite direction. This isomorphism arises because the functor $\mathcal{B}(\cdot)$ used in the denotational model is *strict monoidal*. Both constructs are necessary to allow the language to manipulate composed quantum systems and also terms possibly representing *entangled states*.

Typing contexts (ranged over by symbols $\Delta, \Sigma_1, \Sigma_2$) are finite sequences $\Delta \triangleq x_1 : \mathrm{A}_1, \ldots, x_n : \mathrm{A}_n$ mapping variables to types. We use the notation $!\Delta$ for contexts of the form $x_1 : !\mathrm{A}_1, \ldots, x_n : !\mathrm{A}_n$. *Typing judgements* have the form $\Delta \vdash M : \mathrm{A}$ and follow a linear typing discipline to deal with quantum data. The typing rules are given in Figure 8 where, as usual, Σ_1, Σ_2 denotes the disjoint union of Σ_1 and Σ_2. Classical bits correspond to the type $\mathit{bit} \triangleq I + I$ and mixed state qubits have type $\mathit{qbit} \triangleq \mathcal{B}(\mathbf{qbit})$. The rule for the term $\mathit{pure}(\boldsymbol{t})$ introduces a state $\boldsymbol{t}$ from the quantum control fragment as a term of type $\mathcal{B}(\boldsymbol{Q})$ into the main calculus. The typing rule for measurement maps a term M of quantum type $\mathcal{B}(\boldsymbol{Q})$ to a term $\mathit{meas}(M)$ of classical type $\overline{\boldsymbol{Q}}$, where the type $\overline{\boldsymbol{Q}}$ is defined inductively on the structure of the pure quantum type $\boldsymbol{Q}$:

$$\overline{\boldsymbol{I}} \triangleq I, \quad \overline{\boldsymbol{Q}_1 \otimes \boldsymbol{Q}_2} \triangleq \overline{\boldsymbol{Q}_1} \otimes \overline{\boldsymbol{Q}_2}, \quad \overline{\boldsymbol{Q}_1 \oplus \boldsymbol{Q}_2} \triangleq \overline{\boldsymbol{Q}_1} + \overline{\boldsymbol{Q}_2}, \quad \overline{\mathbf{qnat}} \triangleq \mathrm{Nat}.$$

$$\frac{!\Delta, \Sigma_1 \vdash L : \mathrm{A} + \mathrm{B} \qquad !\Delta, \Sigma_2, x : \mathrm{A} \vdash M : \mathrm{C} \qquad !\Delta, \Sigma_2, y : \mathrm{B} \vdash N : \mathrm{C}}{!\Delta, \Sigma_1, \Sigma_2 \vdash \mathit{case}\, L\, \mathit{of}\{\mathit{inl}(x) \to M,\ \mathit{inr}(y) \to N\} : \mathrm{C}}$$

$$\frac{\Delta, x : \mathrm{A} \vdash M : \mathrm{B}}{\Delta \vdash \lambda x.M : \mathrm{A} \multimap \mathrm{B}} \qquad \frac{!\Delta, \Sigma_1 \vdash M : \mathrm{A} \multimap \mathrm{B} \quad !\Delta, \Sigma_2 \vdash N : \mathrm{A}}{!\Delta, \Sigma_1, \Sigma_2 \vdash M\, N : \mathrm{B}}$$

$$\frac{!\Delta, \Sigma_1 \vdash L : \mathrm{Nat} \quad !\Delta, \Sigma_2 \vdash M : \mathrm{A} \quad !\Delta, \Sigma_2, x : \mathrm{Nat} \vdash N : \mathrm{A}}{!\Delta, \Sigma_1, \Sigma_2 \vdash \mathit{match}\, L\, \mathit{with}\{\mathit{zero} \to M,\ \mathit{succ}(x) \to N\} : \mathrm{A}} \qquad \frac{\cdot \vdash \boldsymbol{t} : \boldsymbol{Q}}{!\Delta \vdash \mathit{pure}(\boldsymbol{t}) : \mathcal{B}(\boldsymbol{Q})}$$

$$\frac{\Delta \vdash M : \mathcal{B}(\boldsymbol{Q})}{\Delta \vdash \mathit{meas}(M) : \overline{\boldsymbol{Q}}} \qquad \frac{\Vdash u : U(\boldsymbol{Q}_1, \boldsymbol{Q}_2) \qquad !\Delta, \Sigma_1 \vdash M : \mathcal{B}(\boldsymbol{Q}_1)}{!\Delta, \Sigma_1 \vdash \mathcal{B}(u)(M) : \mathcal{B}(\boldsymbol{Q}_2)}$$

$$\frac{!\Delta, \Sigma_1 \vdash M : \mathcal{B}(\boldsymbol{Q}_1) \otimes \mathcal{B}(\boldsymbol{Q}_2) \qquad !\Delta, \Sigma_2, z : \mathcal{B}(\boldsymbol{Q}_1 \otimes \boldsymbol{Q}_2) \vdash N : \mathrm{A}}{!\Delta, \Sigma_1, \Sigma_2 \vdash \mathit{let}\, \mathcal{B}(z) = M\ \mathit{in}\ N : \mathrm{A}}$$

$$\frac{!\Delta, \Sigma_1 \vdash M : \mathcal{B}(\boldsymbol{Q}_1 \otimes \boldsymbol{Q}_2) \qquad !\Delta, \Sigma_2, x : \mathcal{B}(\boldsymbol{Q}_1), y : \mathcal{B}(\boldsymbol{Q}_2) \vdash N : \mathrm{A}}{!\Delta, \Sigma_1, \Sigma_2 \vdash \mathit{let}\, \mathcal{B}(x \otimes y) = M\ \mathit{in}\ N : \mathrm{A}}$$

Fig. 8: Typing rules for terms (excerpt).

This gives the classical analogue of the type $\boldsymbol{Q}$, e.g., $\overline{\mathbf{qbit}} = \mathit{bit}$. For a quantum basis term $\boldsymbol{b}$ of type $\boldsymbol{Q}$, $\overline{\boldsymbol{b}}$ (defined below) denotes its translation to a classical value of type $\overline{\boldsymbol{Q}}$, which we use to represent measurement outcomes.

$$\overline{*} \triangleq *, \quad \overline{\boldsymbol{b}_1 \otimes \boldsymbol{b}_2} \triangleq \overline{\boldsymbol{b}_1} \otimes \overline{\boldsymbol{b}_2}, \quad \overline{\mathbf{inl}\ \boldsymbol{b}} \triangleq \mathit{inl}(\overline{\boldsymbol{b}}), \quad \overline{\mathbf{0}} \triangleq \mathit{zero}, \quad \overline{\mathbf{S}\ \boldsymbol{b}} \triangleq \mathit{succ}(\overline{\boldsymbol{b}}).$$

3.2 Operational Semantics

Quantum configurations. We lay down the operational semantics of the calculus using an adaptation of the notion of a *quantum configuration*.

Definition 1 (Quantum Configuration). *Let* $\mathtt{Conf}$ *be the set of* quantum configurations $\mathcal{C} \triangleq (\boldsymbol{t}, u_\sigma, M)$, *where:*

- $\boldsymbol{t} \in \boldsymbol{Terms}$ *represents a* pure quantum state*;*
- $u_\sigma \in \boldsymbol{Unitaries}$ *represents a symmetric monoidal isomorphism;*
- $M \in$ Terms *is a term from the main calculus.*

In the above definition, we can think of the quantum term $\boldsymbol{t}$ as representing a quantum state, because the quantum control fragment ensures that $\boldsymbol{t}$ can be rewritten (in its equational theory) to a unique quantum value (Proposition 3). Next, u_σ permutes the components of $\boldsymbol{t}$ so that they appear in the same order as variables that represent them in M. We slightly abuse terminology and refer to u_σ as a permutation, but it is meant to be understood as a canonical symmetric monoidal isomorphism (see Proposition 2). If we fix the domain and codomain of u_σ, then the essential data is given by a choice of permutation σ and this makes it easier to understand how each unitary acts on the coordinates, and the contextual rules formulation becomes precise. Note that, thanks to Proposition 2, the pure quantum syntax is expressive enough to represent all the monoidal permutations that we need in the sequel.

Example 5. Consider the quantum configuration $(\boldsymbol{t}, u_{id}, x \otimes y)$, where we have $\boldsymbol{t} \triangleq * \otimes ({}^1\!/\!\sqrt{2} \cdot |\mathbf{0}\rangle \otimes |\mathbf{0}\rangle + {}^1\!/\!\sqrt{2} \cdot |\mathbf{1}\rangle \otimes |\mathbf{1}\rangle)$ and $u_{id} \triangleq \{\!| * \otimes \boldsymbol{w} \otimes \boldsymbol{z} \mapsto \boldsymbol{w} \otimes \boldsymbol{z} \}$

is a unitary with type $\Vdash u_{id} \colon U(\boldsymbol{I} \otimes \mathbf{qbit} \otimes \mathbf{qbit}, \mathbf{qbit} \otimes \mathbf{qbit})$. It conveys the information that the variable x in $x \otimes y$ keeps track of the first qubit in $\boldsymbol{t}$ and that the variable y keeps track of the second one. Whereas in the configuration $(\boldsymbol{t}, u_{swap}, x \otimes y)$ with $u_{swap} \triangleq \{\mid * \otimes \boldsymbol{w} \otimes \boldsymbol{z} \mapsto \boldsymbol{z} \otimes \boldsymbol{w}\}$, the variable x keeps track of the second qubit and y of the first one. Note that, due to quantum entanglement, the term $\boldsymbol{t}$ cannot be decomposed into a non-trivial tensor product, and the variables x and y should be thought of as identifying qubit components of $\boldsymbol{t}$, instead of storing quantum data themselves.

A quantum configuration $\mathcal{V} = (\boldsymbol{v}, u_\sigma, V)$, when both the quantum term $\boldsymbol{v}$ and term V are values, is called a *value configuration*. For example, $(|0\rangle, u_{id}, x)$ is a value configuration. A quantum configuration $\mathcal{SV} = (\boldsymbol{v}, u_\sigma, M)$, where $\boldsymbol{v}$ is a quantum value, is a *semi-value configuration*. We write `VConf` (resp. `SVConf`) for the set of (resp. semi-)value configurations. The reason we introduce semi-value configurations, is because they allow us to define our operational semantics for the main calculus modulo equality of the pure quantum terms $\boldsymbol{t}$ (analogous to the quantum lambda-calculus). We already proved that values $\boldsymbol{v}$ are normal forms for the pure terms (Proposition 3), so this makes them a natural choice.

Definition 2. *A quantum configuration* $(\boldsymbol{t}, u_\sigma, M)$ *is* well-formed *with type* A*, written as* $(\boldsymbol{t}, u_\sigma, M) \colon \mathrm{A}$ *if the following judgments can be derived:*

- $\cdot \vdash \boldsymbol{t} \colon \boldsymbol{Q}_1 \otimes \cdots \otimes \boldsymbol{Q}_n$*;*
- $\cdot \Vdash u_\sigma \colon U(\boldsymbol{Q}_1 \otimes \cdots \otimes \boldsymbol{Q}_n, \boldsymbol{Q}'_1 \otimes \cdots \otimes \boldsymbol{Q}'_m)$*;*
- $x_1 : \mathcal{B}(\boldsymbol{Q}'_1), \ldots, x_m : \mathcal{B}(\boldsymbol{Q}'_m) \vdash M : \mathrm{A}$*.*

We write $\mathrm{WF}_X(A)$ for the set of well-formed configurations in X with type A, where $X \in \{\mathtt{Conf}, \mathtt{VConf}, \mathtt{SVConf}\}$.

Example 6. The configuration $((1/\sqrt{2} \cdot |00\rangle + 1/\sqrt{2} \cdot |11\rangle) \otimes |0\rangle, u_{\mathrm{id}}, x \otimes y)$ with:

- $\cdot \vdash (1/\sqrt{2} \cdot |00\rangle + 1/\sqrt{2} \cdot |11\rangle) \otimes |0\rangle \ : \mathbf{qbit}^3$;
- $\cdot \Vdash u_{\mathrm{id}} \colon U(\mathbf{qbit}^3, \mathbf{qbit}^3)$;
- $x : \mathcal{B}(\mathbf{qbit}^2), y : \mathit{qbit} \vdash x \otimes y : \mathcal{B}(\mathbf{qbit}^2) \otimes \mathit{qbit}$.

In this configuration, u_{id} is just the identity, x points to the Bell state in $\boldsymbol{t}$ whereas y points to $|\mathbf{0}\rangle$. We have $n = 3, m = 2$, with $\boldsymbol{Q}_1 = \boldsymbol{Q}_2 = \boldsymbol{Q}_3 = \mathbf{qbit}$, $\boldsymbol{Q}'_1 = \mathbf{qbit}^2$ and $\boldsymbol{Q}'_2 = \mathbf{qbit}$. The configuration is well-formed.

The above example highlights the need for m and n (from Definition 2) to be different values. The intuition behind this difference is that the unitary u_σ partitions the n quantum types into m blocks, where each block is represented by a variable in M. Furthermore, in situations where we wish to combine two such blocks which are not next to each other in the tensor expression of the n types, we allow the possibility of permuting these blocks and merging them so that now two blocks that were initially represented by two variables are represented by one. This is formally expressed in the reduction rules we define next. The role of u_σ in a well-formed configuration is illustrated in Figure 9. The *small-step reduction* $\cdot \to_{\cdot} \cdot \subseteq \mathtt{Conf} \times [0,1] \times \mathtt{Conf}$ is the relation defined by the rules of Figure 10: the reduction $\mathcal{C} \to_p \mathcal{C}'$ holds when the quantum configuration $\mathcal{C}$ reduces to $\mathcal{C}'$ with probability $p \in [0,1]$.

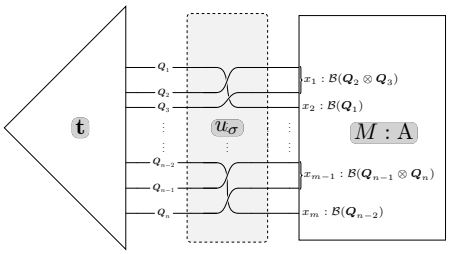


Fig. 9: Graphical representation of a well-formed quantum configuration.

The reduction rules along with the unitaries required to state them are given in Figure 10. $\{\boldsymbol{b}_{ij}\}_{i,j}$ stand for basis terms and s in the reduction rule for $\mathit{meas}(x)$ corresponds to the position of $\boldsymbol{b}_i$ in the tensor product appearing in the quantum term. All sums are finite. The unitary $u_{\sigma_{gather}}$ appearing in the reduction rule for the term $\mathit{let}\,\mathcal{B}(z) = x \otimes y\ \mathit{in}\ N$ merges two consecutive variables into one block, so that now a single variable can represent it. Similarly, in the reduction rule for the term $\mathit{let}\,\mathcal{B}(x \otimes y) = z\ \mathit{in}\ N$, the unitary $u_{\sigma_{divide}}$ partitions a block into two, so that two variables can represent them. The two rules are symmetric as expected, since these terms only allow switching between types $\mathcal{B}(\boldsymbol{Q}_1 \otimes \boldsymbol{Q}_2)$ and $\mathcal{B}(\boldsymbol{Q}_1) \otimes \mathcal{B}(\boldsymbol{Q}_2)$. The last rule of Figure 10 is a *contextual rule* which holds for any *evaluation context* E. Evaluation contexts are defined by the following grammar:

$$\begin{aligned} E := {} & [.] \mid E \otimes N \mid V \otimes E \mid \mathit{let}\ x \otimes y = E\ \mathit{in}\ N \mid \mathit{inl}(E) \mid \mathit{inr}(E) \\ & \mid \mathit{case}\, E\, \mathit{of}\{\mathit{inl}(x) \to M,\ \mathit{inr}(y) \to N\} \mid \mathit{force}(E) \mid \mathit{succ}(E) \mid E\, N \\ & \mid V\, E \mid \mathit{meas}(E) \mid \mathcal{B}(u)(E) \mid \mathit{let}\ \mathcal{B}(z) = E\ \mathit{in}\ N \mid \mathit{let}\ \mathcal{B}(x \otimes y) = E\ \mathit{in}\ N \end{aligned}$$

One of the premises of the contextual rule states that if σ is a permutation which does not act on the same set of indices as σ_1, σ_2, then we can form a "union" of these permutations by composing an *extension* of the two. For a permutation $\sigma : S \to S$, we can define its extension $\sigma^{ext} : S \sqcup S' \to S \sqcup S'$ by $\sigma^{ext}(s) \triangleq \sigma(s)$, if $s \in S$, $\sigma^{ext}(s') \triangleq s'$, if $s' \in S'$. We now introduce the reduction relation we use to define the operational semantics.

Definition 3. *The* reduction relation $\cdot \rightsquigarrow_{\cdot} \cdot \subseteq \mathtt{SVConf} \times [0,1] \times \mathtt{SVConf}$ *is defined in the following way: we write* $(\boldsymbol{v}, u_\sigma, M) \rightsquigarrow_p (\boldsymbol{v}', u_{\sigma'}, M')$ *whenever*

$$\frac{\cdot \vdash \boldsymbol{v} = \boldsymbol{t} : \boldsymbol{Q} \qquad (\boldsymbol{t}, u_\sigma, M) \to_p (\boldsymbol{t}', u_{\sigma'}, M') \qquad \cdot \vdash \boldsymbol{t}' = \boldsymbol{v}' : \boldsymbol{Q}'}{(\boldsymbol{v}, u_\sigma, M) \rightsquigarrow_p (\boldsymbol{v}', u_{\sigma'}, M')}$$

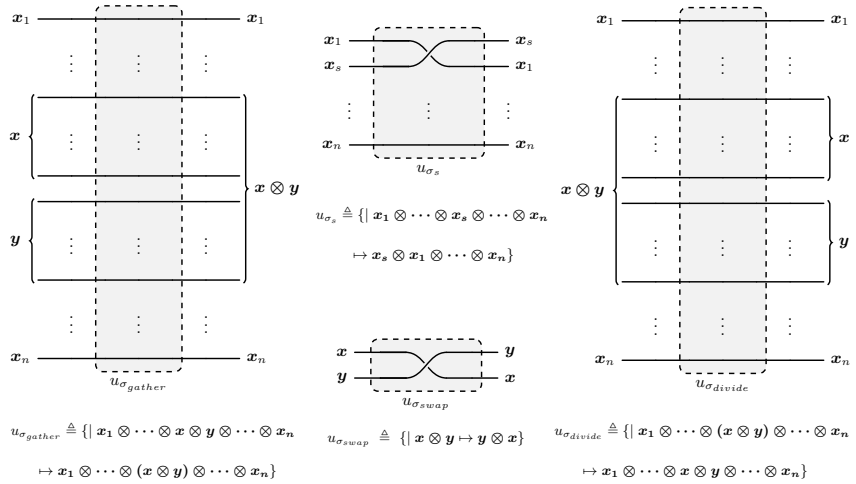
The intuition behind the above definition is that it allows us to reason modulo equality of the pure terms $\boldsymbol{t}$, as a consequence of Proposition 3.

3.3 Main properties

We show that well-formedness is preserved by the reduction relation $\cdot \rightsquigarrow_{\cdot} \cdot$.

Theorem 3 (Subject Reduction). *For a configuration* $\mathcal{C}_1 \in \mathrm{WF}_{\mathtt{SVConf}}(A)$*, if* $\mathcal{C}_1 \rightsquigarrow_p \mathcal{C}_2$ *for some probability* $p \in [0,1]$*, then* $\mathcal{C}_2 \in \mathrm{WF}_{\mathtt{SVConf}}(A)$*.*

Next, we show that progress and strong normalisation hold. More specifically, a well-formed semi-value configuration $\mathcal{C}$ is either a value configuration or it reduces to a finite number of semi-value configurations with total probability 1, *i.e.* if we have not reached a normal form then the probability to be stuck is 0.



$$
\begin{aligned}
(\boldsymbol{t}, u_\sigma, \mathit{case\ inl}(V)\ \mathit{of}\{\mathit{inl}(x){\to}M,\ \ \mathit{inr}(y){\to}N\}) &\to_1 (\boldsymbol{t}, u_\sigma, M[V/x]) \\
(\boldsymbol{t}, u_\sigma, \mathit{case\ inr}(V)\ \mathit{of}\{\mathit{inl}(x){\to}M,\ \ \mathit{inr}(y){\to}N\}) &\to_1 (\boldsymbol{t}, u_\sigma, N[V/y]) \\
(\boldsymbol{t}, u_\sigma, \mathit{let}\ x \otimes y = V \otimes W\ \ \mathit{in}\ M) &\to_1 (\boldsymbol{t}, u_\sigma, M[V/x, W/y]) \\
(\boldsymbol{t}, u_\sigma, \mathit{force}(\mathit{lift}(M))) &\to_1 (\boldsymbol{t}, u_\sigma, M) \\
(\boldsymbol{t}, u_\sigma, (\lambda x.M)\, V) &\to_1 (\boldsymbol{t}, u_\sigma, M[V/x]) \\
(\boldsymbol{t}, u_\sigma, \mathit{match\ zero\ with}\{\mathit{zero}{\to}M,\ \ \mathit{succ}(x){\to}N\}) &\to_1 (\boldsymbol{t}, u_\sigma, M) \\
(\boldsymbol{t}, u_\sigma, \mathit{match\ succ}(V)\ \mathit{with}\{\mathit{zero}{\to}M,\ \ \mathit{succ}(x){\to}N\}) &\to_1 (\boldsymbol{t}, u_\sigma, N[V/x]) \\
(\boldsymbol{t}, u_\sigma, \mathit{pure}(\boldsymbol{t}')) &\to_1 (\boldsymbol{t} \otimes \boldsymbol{t}', u_{\sigma_{swap}} \circ (u_\sigma \otimes u_{id}), x) \\
(\Sigma_i p_i{\cdot}\Sigma_j \boldsymbol{\alpha}_{ij}{\cdot}\boldsymbol{b}'_{ij} \otimes \cdots \otimes \boldsymbol{b}_i \otimes \cdots, u_{\sigma_s}, \mathit{meas}(x)) &\to_{|p_k|^2} (\Sigma_j \boldsymbol{\alpha}_{kj}{\cdot}\boldsymbol{b}'_{kj} \otimes \cdots, u_{id}, \overline{\boldsymbol{b}_k}) \\
(\boldsymbol{t}, u_\sigma, \mathcal{B}(u)(z)) &\to_1 ((u^*_\sigma \circ (u \otimes id) \circ u_\sigma)\, \boldsymbol{t}, u_\sigma, z) \\
(\boldsymbol{t}, u_\sigma, \mathit{let}\ \mathcal{B}(z) = x \otimes y\ \ \mathit{in}\ N) &\to_1 (\boldsymbol{t}, u_{\sigma_{gather}} \circ u_\sigma, N) \\
(\boldsymbol{t}, u_\sigma, \mathit{let}\ \mathcal{B}(x \otimes y) = z\ \ \mathit{in}\ N) &\to_1 (\boldsymbol{t}, u_{\sigma_{divide}} \circ u_\sigma, N)
\end{aligned}
$$

$$
\frac{(\boldsymbol{t}_1, u_{\sigma_1^{ext}}, M_1) \to_p (\boldsymbol{t}_2, u_{\sigma_2^{ext}}, M_2) \qquad dom(\sigma) \cap (dom(\sigma_1) \cup dom(\sigma_2)) = \emptyset}{(\boldsymbol{t}_1, u_{\sigma^{ext} \circ \sigma_1^{ext}}, E[M_1]) \to_p (\boldsymbol{t}_2, u_{\sigma^{ext} \circ \sigma_2^{ext}}, E[M_2])}
$$

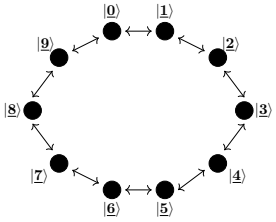
Fig. 10: Reduction rules and their unitary permutations.

Theorem 4 (Progress). *If $\mathcal{C} \in \mathrm{WF}_{\mathtt{SVConf}}(A)$, then either $\mathcal{C} \in \mathtt{VConf}$, or there exists $\mathcal{C}_i \in \mathrm{WF}_{\mathtt{SVConf}}(A)$ such that $\mathcal{C} \rightsquigarrow_{p_i} \mathcal{C}_i$. Moreover, if $\{\mathcal{C}_i\}_{i \in I}$ is the set of all such distinct (not α-equivalent) configurations, then I is finite, and $\sum_i p_i = 1$.*

Theorem 5 (Strong Normalisation). *For a configuration $\mathcal{C} \in \mathrm{WF}_{\mathtt{SVConf}}(A)$, there is no infinite sequence of reductions $\mathcal{C} \rightsquigarrow_{p_0} \mathcal{C}_1 \rightsquigarrow_{p_1} \mathcal{C}_2 \rightsquigarrow_{p_2} \cdots$.*

3.4 Illustrating Example: Quantum Walk Search

Given a graph with some marked vertices, a quantum walk search is the quantum analogue of a random walk [4], which looks for these marked vertices. Each node in the graph represents the state of two quantum registers. A condition on the application of a coin operator on the first register decides the direction in which a walker should take the next step. A step in this direction is represented by the application of a unitary on the second register. After a given number of steps, measurement is performed on the second register to reveal whether the walker has arrived on a marked node. Here, we present a k-step quantum walk on a cycle with 10 nodes for simplicity in our language. Representing the second register with qubits, in this case, would require 4 qubits, whereas we represent the state of the second register with a single **qnat**. In the unitary $\text{CNOT}_{\mathbf{qnat}}$ below, we use some (hopefully obvious) syntactic sugar with the expressions involving **n** in order to avoid writing all the cases; the letter **n** here should not be seen as a variable.



$$u_1 \triangleq \left\{ \begin{array}{ll} | \ |\underline{\mathbf{2n}}\rangle & \mapsto |\underline{\mathbf{2n{+}1}}\rangle, \mathbf{n} \leq 4 \\ | \ |\underline{\mathbf{2n{+}1}}\rangle & \mapsto |\underline{\mathbf{2n}}\rangle, \quad \mathbf{n} \leq 4 \\ | \ |\underline{\mathbf{y{+}10}}\rangle & \mapsto |\underline{\mathbf{y{+}10}}\rangle \end{array} \right\} \quad u_2 \triangleq \left\{ \begin{array}{ll} | \ |\underline{\mathbf{0}}\rangle & \mapsto |\underline{\mathbf{9}}\rangle \\ | \ |\underline{\mathbf{2n{+}2}}\rangle & \mapsto |\underline{\mathbf{2n{+}1}}\rangle, \mathbf{n} \leq 3 \\ | \ |\underline{\mathbf{2n{+}1}}\rangle & \mapsto |\underline{\mathbf{2n{+}2}}\rangle, \mathbf{n} \leq 3 \\ | \ |\underline{\mathbf{9}}\rangle & \mapsto |\underline{\mathbf{0}}\rangle \\ | \ |\underline{\mathbf{y{+}10}}\rangle & \mapsto |\underline{\mathbf{y{+}10}}\rangle \end{array} \right\}$$

$$\text{CNOT}_{\mathbf{qnat}} : U(\mathbf{qbit} \otimes \mathbf{qnat}, \mathbf{qbit} \otimes \mathbf{qnat})$$
$$\text{CNOT}_{\mathbf{qnat}} \triangleq \mathbf{qif}\ \boldsymbol{x}\ \mathbf{then}\ \boldsymbol{x} \otimes u_1\ \mathbf{else}\ \boldsymbol{x} \otimes u_2$$

We define the unitary S corresponding to one step of the walk as $S \triangleq \text{CNOT}_{\mathbf{qnat}} \circ (\text{Had} \otimes \text{Id})$. Finally, the k-step walk can be represented in our language as follows, where S^k represents k compositions of S with itself:

$$\text{walk} : (\mathcal{B}(\mathbf{qbit} \otimes \mathbf{qnat}) \multimap \mathit{qbit} \otimes \text{Nat})$$
$$\text{walk} \triangleq \lambda z.\mathit{let}\ \mathcal{B}(x_1 \otimes x_2) = \mathcal{B}(S^k)(z)\ \mathit{in}\ x_1 \otimes \mathit{meas}(x_2)$$

3.5 Denotational Semantics: Soundness and Adequacy

In this section we describe the denotational semantics of the main calculus. Figure 11 succinctly describes the interaction between the pure fragment and the mixed fragment. We present the functor which facilitates the main contribution of the paper, i.e combining quantum and classical control.

The $\mathcal{B}$ functor. In order to model the interaction between the pure quantum subsystem and the main calculus, we define the functor $\mathcal{B} : \mathbf{Isometry} \to \mathbf{NCPSU}$, mapping a Hilbert space H to $\mathcal{B}(H)$. For an isometry $f \colon H_1 \to H_2$, the map $(f^* \circ (-) \circ f)$ is an NCPSU morphism $B(H_2) \to B(H_1)$, so by defining $\mathcal{B}(f) \triangleq (f^* \circ (-) \circ f)^{op}$, we get that $\mathcal{B}(f) \colon \mathcal{B}(H_1) \to \mathcal{B}(H_2)$ in **NCPSU**. We remark that this functor restricts to a functor $\mathcal{B} \colon \mathbf{Unitary} \to \mathbf{NMIU}$ and we show that the functor $\mathcal{B} : \mathbf{Isometry} \to \mathbf{NCPSU}$ is strict monoidal.

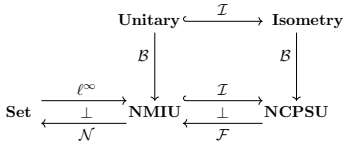


Fig. 11: Categorical relations.

$$\begin{aligned}
[\![!\Delta \vdash \mathit{pure}(\boldsymbol{t}) : \mathcal{B}(\boldsymbol{Q})]\!] &\triangleq \mathcal{B}[\![\boldsymbol{t}]\!] \circ \diamond_{!\Delta} \\
[\![\Delta \vdash \mathit{meas}(M) : \overline{\boldsymbol{Q}}]\!] &\triangleq m^{\boldsymbol{Q}} \circ [\![\Delta \vdash M : \mathcal{B}(\boldsymbol{Q})]\!] \\
[\![\Delta \vdash \mathcal{B}(u)(M) : \mathcal{B}(\boldsymbol{Q}_2)]\!] &\triangleq \mathcal{B}[\![u]\!] \circ [\![\Delta \vdash M : \mathcal{B}(\boldsymbol{Q}_1)]\!]
\end{aligned}$$

Fig. 12: Interpretation of typing judgements (an excerpt).

Interpretation of Types. We interpret types as von Neumann algebras. The types corresponding to a linear call-by-value lambda-calculus are interpreted in the standard way: $[\![\mathrm{I}]\!] \triangleq \mathbb{C}$, $[\![\mathrm{A} + \mathrm{B}]\!] \triangleq [\![\mathrm{A}]\!] \oplus [\![\mathrm{B}]\!]$, $[\![\mathrm{A} \otimes \mathrm{B}]\!] \triangleq [\![\mathrm{A}]\!] \otimes [\![\mathrm{B}]\!]$, $[\![!\mathrm{A}]\!] \triangleq !{}[\![\mathrm{A}]\!]$ and $[\![\mathrm{A} \multimap \mathrm{B}]\!] \triangleq [\![\mathrm{A}]\!] \multimap [\![\mathrm{B}]\!]$. The type for natural numbers is interpreted as the commutative von Neumann algebra $\ell^\infty(\mathbb{N})$, i.e $[\![\mathrm{Nat}]\!] \triangleq \ell^\infty(\mathbb{N})$. Finally, the interpretation of the type $\mathcal{B}(\boldsymbol{Q})$ uses the $\mathcal{B}$ functor that allows us to incorporate pure quantum primitives into our semantics, i.e $[\![\mathcal{B}(\boldsymbol{Q})]\!] \triangleq \mathcal{B}[\![\boldsymbol{Q}]\!]$.

Interpretation of Typing judgements. A typing judgment of the form $\Delta \vdash M \colon \mathrm{A}$ is interpreted as a morphism $[\![\Delta \vdash M \colon \mathrm{A}]\!] \colon [\![\Delta]\!] \to [\![\mathrm{A}]\!]$ in **NCPSU** and we often abbreviate this by writing $[\![M]\!]$. An excerpt of the interpretation is defined in Figure 12.

Theorem 6. *For every pure quantum type $\boldsymbol{Q}$, there exist a set X, an NMIU isomorphism $\alpha_{\overline{\boldsymbol{Q}}} : [\![\overline{\boldsymbol{Q}}]\!] \cong \ell^\infty(X)$, and an isometric isomorphism $\beta_{\boldsymbol{Q}} : [\![\boldsymbol{Q}]\!] \cong \ell^2(X)$. Moreover, $\forall \boldsymbol{b} \in$ **Basis Terms**, $\exists x \in X$ s.t. $\beta_{\boldsymbol{Q}}([\![\boldsymbol{b}]\!]) = \alpha_{\overline{\boldsymbol{Q}}}([\![\overline{\boldsymbol{b}}]\!]) = |x\rangle$.*

Using the above theorem, measurement is interpreted as the map $m^{\boldsymbol{Q}} : \mathcal{B}([\![\boldsymbol{Q}]\!]) \to [\![\overline{\boldsymbol{Q}}]\!]$ defined as $m^{\boldsymbol{Q}} \triangleq \alpha^{op}_{\overline{\boldsymbol{Q}}} \circ m^{op}_X \circ \mathcal{B}(\beta_{\boldsymbol{Q}})$.

Interpretation of Configurations. A well-formed configuration $(\boldsymbol{t}, u_\sigma, M)$ with $\cdot \vdash \boldsymbol{t} \colon \boldsymbol{Q}$, $\Vdash u_\sigma \colon (\boldsymbol{Q}, \boldsymbol{Q}'_1 \otimes \cdots \otimes \boldsymbol{Q}'_m)$, and $x_1 \colon \mathcal{B}(\boldsymbol{Q}'_1), \ldots, x_m \colon \mathcal{B}(\boldsymbol{Q}'_m) \vdash M \colon \mathrm{A}$ is interpreted as the NCPSU morphism

$$[\![(\boldsymbol{t}, u_\sigma, M) : \mathrm{A}]\!] \triangleq \left(\mathbb{C} \xrightarrow{\mathcal{B}([\![u_\sigma \boldsymbol{t}]\!])} \begin{array}{c} \mathcal{B}(\boldsymbol{Q}'_1 \otimes \cdots \otimes \boldsymbol{Q}'_m) \\ = \\ \mathcal{B}(\boldsymbol{Q}'_1) \otimes \cdots \otimes \mathcal{B}(\boldsymbol{Q}'_m) \end{array} \xrightarrow{[\![M]\!]} \mathrm{A} \right).$$

We can now show that our semantic interpretation is sound with respect to single-step reduction.

Theorem 7 (Soundness). *For $\mathcal{C} \in \mathrm{WF}_{\mathit{SVConf}}(A)$, if $\mathcal{C} \notin$ VConf, then*

$$[\![\mathcal{C} : \mathrm{A}]\!] = \sum_{\mathcal{C} \rightsquigarrow_p \mathcal{C}'} p [\![\mathcal{C}' : \mathrm{A}]\!],$$

where the sum ranges over all possible reducts $\mathcal{C} \rightsquigarrow_p \mathcal{C}'$.

Next, we can show that our interpretation is sound in a big-step sense as well, which follows easily using the previous result and strong normalisation.

Theorem 8 (Strong Adequacy). *For* $\mathcal{C} \in \mathrm{WF}_{\mathit{SVConf}}(A)$,

$$[\![\mathcal{C}]\!] = \sum_{\mathcal{V} \in \mathit{VConf}} P(\mathcal{C} \to_* \mathcal{V})[\![\mathcal{V}]\!],$$

where $P(\mathcal{C} \to_* \mathcal{V})$ *indicates the overall probability that* $\mathcal{C}$ *reduces to* $\mathcal{V}$.

This implies that, for any well-formed configuration $\mathcal{C}$, its interpretation $[\![\mathcal{C}]\!]$ is an NCPU map (i.e., it is not merely subunital) and therefore it corresponds to a quantum channel in the Heisenberg picture of quantum mechanics.

4 Conclusion and Perspectives

We described a programming language which has support for both pure state quantum computation and mixed state quantum computation. We began by describing the pure quantum subsystem (§2), for which we introduced an equational theory and proved its completeness w.r.t. the denotational semantics (§2.4). Then, we described the main calculus (§3.1) which uses a new adaptation of quantum configurations (§3.2), which we used to define the operational semantics of the language. Here, the interaction between the pure quantum subsystem and the main calculus becomes apparent. We showed that our denotational semantics (§3.5) has a clear and appropriate interpretation as channels in the Heisenberg picture of quantum mechanics and we showed it is sound and adequate with respect to the operational semantics (§3.5).

Although our language has support for Nat, an inductive type, an obvious extension would be to include more general inductive types. Additionally, support for higher-order *pure* quantum computation is another feature that would be interesting to add in a future work.

Another matter that could be tackled as future work is recursion. In the pure system, adding general recursion is an open and difficult problem [13, Section 5.2]. In the main calculus, our denotational model supports recursion for first-order quantum functions [16,12]. However, our model does not support general recursion when quantum lambda abstractions are involved. There are other denotational models that support general recursion and quantum lambda abstractions, but they do not support types like qnat that correspond to $\ell^2(\mathbb{N})$, instead they support recursive types that intuitively correspond to infinite classical combinations of spaces involving finite-dimensional Hilbert spaces, which is not the same as $\ell^2(\mathbb{N})$. A model for quantum lambda abstractions, qnat and general recursion is an open problem to the best of our knowledge.

Acknowledgements

The authors would like to thank Benoît Valiron for helpful comments and discussions. Louis Lemonnier's research was funded by the Engineering and Physical Sciences Research Council (EPSRC) under project EP/X025551/1 "Rubber DUQ: Flexible Dynamic Universal Quantum programming". This work is also supported by the HORIZON 2020 project NEASQC, by the Inria associate team TC(Pro)^3, by the PEPR integrated project EPiQ ANR-22-PETQ-0007, and by the HQI initiative ANR-22-PNCQ-0002.

References

1. Andrés-Martínez, P.: Unbounded loops in quantum programs: categories and weak-while loops. Ph.D. thesis, Laboratory for Foundations of Computer Science, School of Informatics, University of Edinburgh (2022). https://doi.org/https://doi.org/10.48550/arXiv.2212.05371
2. Arrighi, P., Dowek, G.: Lineal: A linear-algebraic lambda-calculus. Log. Methods Comput. Sci. **13**(1) (2017). https://doi.org/10.23638/LMCS-13(1:8)2017, `https://doi.org/10.23638/LMCS-13(1:8)2017`
3. Chardonnet, K.: Towards a Curry-Howard Correspondence for Quantum Computation. (Vers une correspondance de Curry-Howard pour le calcul quantique). Ph.D. thesis, Université Paris-Saclay, France (2023), `https://tel.archives-ouvertes.fr/tel-03959403`
4. Childs, A.M., Cleve, R., Deotto, E., Farhi, E., Gutmann, S., Spielman, D.A.: Exponential algorithmic speedup by a quantum walk. In: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing. pp. 59–68 (2003)
5. Cho, K., Westerbaan, A.: Von Neumann Algebras form a Model for the Quantum Lambda Calculus (2016). https://doi.org/https://doi.org/10.48550/arXiv.1603.02133
6. Clairambault, P., de Visme, M.: Full abstraction for the quantum lambda-calculus. Proc. ACM Program. Lang. **4**(POPL), 63:1–63:28 (2020). https://doi.org/10.1145/3371131, `https://doi.org/10.1145/3371131`
7. Clairambault, P., de Visme, M., Winskel, G.: Game semantics for quantum programming. PACMPL **3**(POPL), 32:1–32:29 (2019). https://doi.org/10.1145/3290345
8. Díaz-Caro, A., Malherbe, O.: Semimodules and the (syntactically-)linear lambda calculus. CoRR **abs/2205.02142** (2022). https://doi.org/10.48550/ARXIV.2205.02142, `https://doi.org/10.48550/arXiv.2205.02142`
9. Díaz-Caro, A., Guillermo, M., Miquel, A., Valiron, B.: Realizability in the unitary sphere. In: LICS 2019. IEEE (jun 2019). https://doi.org/10.1109/lics.2019.8785834, `https://doi.org/10.1109%2Flics.2019.8785834`
10. Díaz-Caro, A., Malherbe, O.: Quantum control in the unitary sphere: Lambda-s1 and its categorical model. Logical Methods in Computer Science **Volume 18, Issue 3** (2022). https://doi.org/10.46298/lmcs-18(3:32)2022, `https://doi.org/10.46298%2Flmcs-18%283%3A32%292022`
11. Green, A.S., Lumsdaine, P.L., Ross, N.J., Selinger, P., Valiron, B.: Quipper: a scalable quantum programming language. In: Proceedings of the 34th ACM SIGPLAN conference on Programming language design and implementation. pp. 333–342 (2013)
12. Jia, X., Kornell, A., Lindenhovius, B., Mislove, M.W., Zamdzhiev, V.: Semantics for variational quantum programming. In: POPL 2022. vol. 6, pp. 1–31. ACM (2022). https://doi.org/10.1145/3498687
13. Lemonnier, L.: The Semantics of Effects : Centrality, Quantum Control and Reversible Recursion. Theses, Université Paris-Saclay (Jun 2024), `https://theses.hal.science/tel-04625771`
14. Pagani, M., Selinger, P., Valiron, B.: Applying quantitative semantics to higher-order quantum computing. In: POPL 2014. pp. 647–658. ACM (2014). https://doi.org/10.1145/2535838.2535879

15. Paykin, J., Rand, R., Zdancewic, S.: Qwire: a core language for quantum circuits. ACM SIGPLAN Notices **52**(1), 846–858 (2017)
16. Péchoux, R., Perdrix, S., Rennela, M., Zamdzhiev, V.: Quantum programming with inductive datatypes: Causality and affine type theory. In: FoSSaCS 2020. Lecture Notes in Computer Science, vol. 12077, pp. 562–581. Springer (2020). https://doi.org/10.1007/978-3-030-45231-5_29
17. Sabry, A., Valiron, B., Vizzotto, J.K.: From symmetric pattern-matching to quantum control. In: FoSSaCS 2018). Lecture Notes in Computer Science, vol. 10803, pp. 348–364. Springer (2018). https://doi.org/10.1007/978-3-319-89366-2_19
18. Selinger, P.: Towards a quantum programming language. Mathematical Structures in Computer Science **14**(4), 527–586 (2004). https://doi.org/https://doi.org/10.1017/S0960129504004256
19. Selinger, P., Valiron, B.: A lambda calculus for quantum computation with classical control. Mathematical Structures in Computer Science **16**(3), 527–552 (2006). https://doi.org/10.1017/S0960129506005238
20. Selinger, P., Valiron, B., et al.: Quantum lambda calculus. Semantic techniques in quantum computation pp. 135–172 (2009)
21. Tsukada, T., Asada, K.: Enriched presheaf model of quantum FPC. Proc. ACM Program. Lang. **8**(POPL), 362–392 (2024). https://doi.org/10.1145/3632855, `https://doi.org/10.1145/3632855`
22. Valiron, B.: Semantics of quantum programming languages: Classical control, quantum control. Journal of Logical and Algebraic Methods in Programming **128** (2022). https://doi.org/https://doi.org/10.1016/j.jlamp.2022.100790
23. de Visme, M.: Quantum Game Semantics. (Sémantique des Jeux Quantique). Ph.D. thesis, University of Lyon, France (2020), `https://tel.archives-ouvertes.fr/tel-03045844`
24. Voichick, F., Li, L., Rand, R., Hicks, M.: Qunity: A unified language for quantum and classical computing. Proc. ACM Program. Lang. **7**(POPL), 921–951 (2023). https://doi.org/10.1145/3571225, `https://doi.org/10.1145/3571225`