# A Monad for Classically-Controlled Quantum Computation

Rikiya Kashiwagi, Atsushi Igarashi

We study denotational semantics for Knill's QRAM model in which a classical host controls a separate quantum system via unitary operations and measurements. Certain languages model these quantum operations as ordinary functions, interpreting whole programs as linear maps between Hilbert spaces, enforced by linear types. Others model them as commands that act on an external quantum store, closer to classical programs with mutable state. We adopt the latter style, modeling these operations with a parameterized monad on the category of sets. The monad is obtained from a categorical formulation of quantum instruments, built using convex combinations of quantum channels. Based on this monad, we define big-step operational semantics and denotational semantics for a monadic internal language.

## 1 Introduction

The slogan "quantum data, classical control" [19] characterizes most modern quantum programming languages. They differ widely in surface syntax — from higher-order functional calculi to circuit-description languages, — but share a common operational picture: quantum data lives in a separate store and a classical host program issues quantum operations on it. This has been called the *Quantum Random-Access Machine* (QRAM) model [10].

Although they share the same underlying operational model, their interpretations of quantum operations lead to fundamentally different denotational models. Some languages, such as quantum $\lambda$-calculus [20], treat quantum operations in the same way as ordinary functions: a unitary is a function from qubits to qubits, and measurement is a function from qubits to classical data. This uniform functional view enforces linear typing so that quantum data cannot be cloned [25]. Other languages, such as Q# [23] and its formalization $\lambda_{Q\#}$ [21], treat quantum operations as instructions on a quantum store: qubits are not passed as data, but as references into the store. In this setting, the use of linear typing is restricted only to quantum store manipulations.

This store-based approach matches standard accounts of stateful computation: quantum operations act on an implicit store in the same way that read and write operations act on a mutable state in classical languages. From this view point, unitary application and measurement are algebraic operations [16] of quantum effect, and one expects a monad whose algebraic theory captures classical control of quantum computation [17].

Ever since Moggi's seminal work [14] [13], monads have been extensively used to structure computational effects. It is thus natural to view quantum computation as a computational effect and encapsulate it in a monad. Earlier explorations along this line include the Quantum IO monad [1], an algebraic theory of quantum effects [22], and a monadic model of circuit description languages [18]. More recently, this intuition has been refined by constructing monads that explicitly model *quantum instruments*: families of quantum channels indexed by classical measurement outcomes. In particular, Fritz [7] and Booth et al. [3] independently constructed such monads as a quantum analogue of classical state monads. Subsequently, the same

古典制御された量子計算のモナド意味論

This is an unrefereed paper. Copyrights belong to the Authors.

柏木 力哉　五十嵐 淳, 京都大学, Graduate School of Informatics, Kyoto University.

group of authors studied domain-theoretic structures for a similar monad, which can model QRAM-style programs with general recursion. However, a complete justification within categorical quantum mechanics remains to be established. Furthermore, their operational semantics and adequacy with respect to concrete QRAM programming languages have not yet been developed.

This paper addresses these gaps through two primary contributions:

1. Categorical justification via adjunctions: We provide a purely categorical characterization of the quantum instrument monad as a parameterized monad [2]. Specifically, we derive this monad from a chain of adjunctions connecting three distinct domains: classical sets, classical probabilistic mixtures, and quantum channels.
2. A formal language for the QRAM model: We show that this parameterized monad naturally induces an internal language. Extending Atkey's typed command calculus, we introduce the *quantum command calculus* as a formal programming language for the QRAM model. We endow it with a big-step operational semantics, define its denotational semantics using our monad, and state the adequacy theorem relating the two.

## 2 Modeling Classically-Controlled Quantum Computation

### 2.1 The QRAM Operational Model

Quantum computing is fundamentally rooted in the physical principles of quantum mechanics. However, to reason about and execute quantum algorithms, one does not need the full machinery of continuous physical systems; rather, the system is fully characterized by a concise set of operational primitives: quantum registers, classical registers, and a controller that applies operations and reads out measurement results. This operational perspective allows us to treat quantum computation through the lens of a programming architecture.

Knill's Quantum Random-Access Machine (QRAM) [10] embodies this perspective, providing a conventional language for hybrid quantum–classical architectures. In this setup,

- a classical processor issues *recipes* of quantum operations (often in the form of quantum circuits) to a quantum coprocessor, and
- the quantum coprocessor sends measurement results back to the classical processor.

$$\text{Classical} \underset{\text{measurement results}}{\overset{\text{recipes}}{\rightleftarrows}} \text{Quantum}$$

This interaction characterizes what we call *classically-controlled quantum computation*. The rest of this section gives a mathematical model for such an architecture incrementally.

### 2.2 Quantum Channels

The first step towards a mathematical model of QRAM is to formalize the recipes sent to the quantum coprocessor. Recipes can take various forms, such as a set of commands, quantum circuits, or more abstract graphical languages (e.g. ZX-calculus [4]). For these recipes to be physically realizable, their interpretation must be a *quantum channel*: a completely positive and trace-preserving (CPTP) map on Hilbert spaces.

Let $\mathcal{L}(H)$ denote the space of linear operators on a complex Hilbert space $H$, and let $M_n(\mathbb{C})$ be the algebra of $n \times n$ complex matrices. Complete positivity and trace preservation are defined as follows.

**Definition 2.1** (Completely positive and trace-preserving map)**.** Let $H$ and $K$ be complex Hilbert spaces. A linear map $\Phi : \mathcal{L}(H) \to \mathcal{L}(K)$ is *completely positive* and *trace-preserving* if

**(CP)** For all $n \in \mathbb{N}$, the map $\Phi \otimes I_n : \mathcal{L}(H) \otimes M_n(\mathbb{C}) \to \mathcal{L}(K) \otimes M_n(\mathbb{C})$ preserves positive operators, where $I_n$ is the identity map on $M_n(\mathbb{C})$; and

**(TP)** $\mathrm{Tr}(\Phi(\rho)) = \mathrm{Tr}(\rho)$ for all $\rho \in \mathcal{L}(H)$.

Unitary evolution, discarding a subsystem, adding noise, and performing a non-selective measurement are all examples of CPTP maps.

In what follows, we restrict our attention to finite-dimensional Hilbert spaces, where these maps behave nicely without analytical subtleties. We write **CPTP** for the category whose objects are finite-dimensional complex Hilbert spaces and whose morphisms are CPTP maps. This is the category we use to model the pure quantum part in the QRAM model.

### 2.3 Quantum Instruments

The second step is to formalize selective measurement, i.e., measurement with classical outcomes.

The same motivation applies to the problem of *dynamic lifting* in the literature around Proto-Quipper [5]. Mathematically, this is captured by *quantum instruments* [6] [15]: families of maps indexed by classical outcomes whose sum is itself a channel.

Let $X$ be a set of classical outcomes (for example, the set of bit strings of a fixed length). Concretely, an instrument from $H$ to $K$ with outcomes $X$ assigns each $x \in X$ a map $\Phi_x : \mathcal{L}(H) \to \mathcal{L}(K)$ such that:

- for an input state $\rho$, the probability of obtaining outcome $x$ is $\mathrm{Tr}(\Phi_x(\rho))$;
- the post-measurement state $\rho'$ conditioned on outcome $x$ is $\rho' = \Phi_x(\rho)/\mathrm{Tr}(\Phi_x(\rho))$; and
- if we forget which outcome occurred, the overall evolution is still a valid quantum channel.

Each $\Phi_x$ is completely positive and trace-non-increasing (CPTNI); we call such maps *quantum subchannels*. This leads to the following definition.

**Definition 2.2** ((Finite) quantum instrument)**.** Let $H$ and $K$ be finite-dimensional complex Hilbert spaces, and $X$ a set. A *quantum instrument* with classical outcomes $X$ is a family of subchannels $\{\Phi_x : \mathcal{L}(H) \to \mathcal{L}(K)\}_{x\in X}$ with finite support (i.e., $\Phi_x \neq 0$ for only finitely many $x \in X$) such that their sum $\sum_{x\in X} \Phi_x$ is trace-preserving.

**Example 2.3.** Projective measurement is an example of a quantum instrument. Let $H$ be a Hilbert space and $\{P_x\}_{x\in X}$ a complete family of orthogonal projectors on $H$, i.e.,

- $P_x P_y = \delta_{xy} P_x$ for all $x, y \in X$ and
- $\sum_{x\in X} P_x = id_H$.

Then the family of maps $\{\Phi_x : \mathcal{L}(H) \to \mathcal{L}(H)\}_{x\in X}$ defined by $\Phi_x(\rho) = P_x \rho P_x$ is a quantum instrument with outcomes $X$.

### 2. 4 Composing Instruments

So far, a quantum instrument has been a single step of classically-controlled quantum computation: issuing a recipe of quantum operations and yielding a classical outcome. What remains is to formalize the sequence of such steps, which requires a composition of instruments.

For an input state $\rho$, consider the following procedure in QRAM:

1. Perform some operations on $\rho$ and get the result $x \in X$ with probability $p_x$;
2. Depending on the outcome $x$, perform another operation and get the result $y \in Y$ with probability $q_{x,y}$.

This procedure is interpreted by composing two instruments:

1. $\{\Phi_x : \mathcal{L}(H) \to \mathcal{L}(K)\}_{x\in X}$ where $p_x = \mathrm{Tr}(\Phi_x(\rho))$;
2. $\{\Psi_{x,y} : \mathcal{L}(K) \to \mathcal{L}(L)\}_{y\in Y}$ where $q_{x,y} = \mathrm{Tr}(\Psi_{x,y}(\Phi_x(\rho))/p_x)$.

The overall procedure is then described by

$$\left\{ \sum_{x\in X} (\Psi_{x,y} \circ \Phi_x) : \mathcal{L}(H) \to \mathcal{L}(L) \right\}_{y\in Y}.$$

Composite instruments are well-defined because of the closure of complete positivity under composition and trace preservation. Consequently, complex QRAM control structure can, in principle, be represented as valid composite instruments.

In this low-level representation, however, pure classical computation and quantum operations are deeply intertwined. Applying a classical function $f : X \to Y$ requires lifting $f$ into identity-channel maps and embedding it directly into the summation over quantum channels. This manual folding eliminates any clear structural boundary between classical control and quantum state manipulation.

Such a loss of modularity directly motivates a monadic framework. Monads provide the required modularity: they isolate quantum effects from classical computation, allowing pure classical functions to combine naturally with quantum operations.

## 3 The Quantum Instrument Monad

### 3. 1 Instruments as Distributions of Channels

This section characterizes quantum instruments by the distribution monad. Recall that finite distributions on a set $X$ is

$$\mathcal{D}(X) = \left\{ \mu : X \to [0,1] \;\middle|\; |\mathrm{supp}(\mu)| < \infty, \sum_{x\in X} \mu(x) = 1 \right\}.$$

There is a unique map from $\mathcal{D}(X \times \mathbf{CPTP}(H,K))$ to a quantum instrument: $\iota(\nu) = \{\Phi_x : \mathcal{L}(H) \to \mathcal{L}(K)\}_{x\in X}$ where

$$\Phi_x = \sum_{(x,\Phi)\in \mathrm{supp}(\nu)} \nu(x,\Phi)\Phi.$$

Quantum instruments with outcomes $X$ are therefore the quotient of $\mathcal{D}(X \times \mathbf{CPTP}(H,K))$ by the equivalence relation identifying distributions with the same image under $\iota$.

This quotient has a categorical description as a

coequalizer, which we now describe. Since $\mathcal{D}$ forms a monad on **Set**, and the convex combination of CPTP maps is again a CPTP map, $\mathbf{CPTP}(H,K)$ carries a canonical $\mathcal{D}$-algebra structure

$$a_{H,K}\colon \mathcal{D}(\mathbf{CPTP}(H,K)) \to \mathbf{CPTP}(H,K),$$
$$a_{H,K}(\nu) = \sum_{\Phi\in\mathrm{supp}(\nu)} \nu(\Phi)\Phi.$$

Let $T$ be a monad on **Set**, and denote by $\mathbf{Set}^T$ the Eilenberg–Moore category of $T$-algebras. The forgetful functor $G\colon \mathbf{Set}^T \to \mathbf{Set}$ admits a left adjoint $F\colon \mathbf{Set} \to \mathbf{Set}^T$, yielding the free-forgetful adjunction $F \dashv G$. The following proposition shows that a quantum instrument with outcome set $X$ can be characterized as the underlying set of the $X$-fold coproduct of the $\mathcal{D}$-algebra defined above.

**Proposition 3.1.** *The set*

$$G\left(\coprod_{x\in X} (\boldsymbol{CPTP}(H,K), a_{H,K})\right)$$

*is isomorphic to the set of quantum instruments from $H$ to $K$ with outcome set $X$.*

*Proof.* Every $\mathcal{D}$-algebra has a canonical presentation as coequalizer of free algebras (e.g., Chapter VI in [12]) Then, taking the $X$-fold coproduct of the diagram, we obtain the following:

$$\mathcal{D}(X\times\mathcal{D}(A)) \underset{\mu_{X\times A}}{\overset{\mathcal{D}(\mathrm{id}_X\times a)}{\rightrightarrows}} \mathcal{D}(X\times A) \xrightarrow{q} G\left(\coprod_{x\in X}(A,a)\right)$$

where $(A,a) = (\mathbf{CPTP}(H,K), a_{H,K})$.

An element $\mathcal{D}(X\times\mathbf{CPTP}(H,K))$ is a formal convex combination of pairs $(x,\Phi)$. The quotient map $q$ identifies two such formal distributions if and only if they yield the same channel for each $x\in X$ under the map $a_{H,K}$. Therefore, an equivalence class $[\nu]\in G\left(\coprod_{x\in X}(A,a)\right)$ is uniquely determined by the family of subchannels $(\mathcal{E}_x)_{x\in X}$ given by:

$$\mathcal{E}_x = \sum_{\Phi\in\mathbf{CPTP}} \nu(x,\Phi)\Phi.$$

Since $\sum_{(x,\Phi)\in\mathrm{supp}(\nu)}\nu(x,\Phi) = 1$ and each $\Phi$ is CPTP, the family $(\mathcal{E}_x)_{x\in X}$ forms a valid quantum instrument. Conversely, any instrument $(\mathcal{E}_x)_{x\in X}$ determines a unique equivalence class in the coequalizer, yielding the required bijection. □

### 3. 2 Conv-valued Quantum Channels

There is an alternative presentation of quantum instruments using the convex structure of quantum channels.

**Definition 3.2** (Convex space)**.** A convex space is a set $X$ equipped with a family of binary operations

$$+_p : X\times X\to X \text{ for } p\in[0,1]$$

such that

- **(unit)** $x +_0 y = y$,
- **(idempotence)** $x +_p x = x$,
- **(parametric commutativity)** $x +_p y = y +_{1-p} x$, and
- **(parametric associativity)** $(x +_p y) +_q z = x +_{pq} (y +_r z)$ for all $p,q,r\in[0,1]$ satisfying $(1-p)qr = 1-q$.

A convex map is a function $f : X\to Y$ that preserves convex combinations, i.e.,

$$f(x +_p y) = f(x) +_p f(y)$$

Convex spaces and convex map form a category, denoted **Conv**.

As is well-known, **Conv** is equivalent to the Eilenberg–Moore category of $\mathcal{D}$.

**Proposition 3.3.** $\boldsymbol{Set}^{\mathcal{D}} \cong \boldsymbol{Conv}$.

*Proof.* See e.g. Theorem 4 in [9]. □

Thus, $\mathcal{D}$-algebra used in proposition Proposition 3.1 can be replaced by a **Conv**-object.

Under this equivalence, a $\mathcal{D}$-algebra $(\mathcal{D}(\mathbf{CPTP}(H,K)), a_{H,K})$ in Section 3. 1 is the same data as a convex space defined below.

**Lemma 3.4.** *The hom-set $\boldsymbol{CPTP}(H,K)$ forms a convex space with the family of operations $\{+_p\}_{p\in[0,1]}$ defined by*

$$\Phi +_p \Psi = p\Phi + (1-p)\Psi.$$

**Definition 3.5.** The **Conv**-valued hom-functor of **CPTP**,

$$\mathbf{CPTP}_{\mathbf{Conv}} : \mathbf{CPTP}^{\mathrm{op}}\times\mathbf{CPTP}\to\mathbf{Conv}$$

sends $(H,K)$ to $\mathbf{CPTP}(H,K)$ with the canonical convex structure of Lemma 3.4, and $(\Phi : H'\to H, \Psi : K\to K')$ to the convex map $\Theta\mapsto\Phi;\Theta;\Psi$.

By Proposition 3.3, the construction of Proposition 3.1 can equally be carried out in **Conv** rather than $\mathbf{Set}^{\mathcal{D}}$, applied to $\mathbf{CPTP}_{\mathbf{Conv}}(H,K)$ in place of $(\mathbf{CPTP}(H,K), a_{H,K})$. This gives the definition we use from here on.

**Definition 3.6** (Quantum Instrument Functor)**.** Define the functor $\mathcal{I} : \mathbf{CPTP}^{op}\times\mathbf{CPTP}\times\mathbf{Set}\to\mathbf{Set}$ by

$$\mathcal{I}(H,K,X) = G(X\cdot\mathbf{CPTP}_{\mathbf{Conv}}(H,K))$$

with the evident action on morphisms, where $G : \mathbf{Conv}\to\mathbf{Set}$ is the forgetful functor.

Hereafter, we write $X\cdot-$ for the $X$-fold coproduct

in **Conv**.

### 3.3 Parameterized Monad Structure

Section 2.4 required composing instruments across different Hilbert spaces $H$, $K$, and $L$. Rather than a standard monad on a fixed Hilbert space, we pursue a *parameterized monad* [2] to capture this state-changing composition. A parameterized monads is a generalization of a standard monad indexed by a category $\mathcal{S}$ of "states" or "resources", where the unit and multiplication align with the action on indices.

**Definition 3.7.** Let $\mathcal{C}$ and $\mathcal{S}$ be a category. A *$\mathcal{S}$-parameterized monad* is the triple $(T, \eta, \mu, \tau)$ where

- $T : \mathcal{S}^{op} \times \mathcal{S} \times \mathcal{C} \to \mathcal{C}$ is a functor,
- the *unit* $\eta_{S,A} : A \to T(S, S, A)$ is a family of morphisms, natural in $A$ and dinatural in $S$,
- the *multiplication* $\mu_{S_1,S_2,S_3,A} : T(S_1, S_2, T(S_2, S_3, A)) \to T(S_1, S_3, A)$ is a family of morphisms, natural in $S_1$, $S_3$, and $A$ and dinatural in $S_2$, and

satisfying the standard monad laws: $\eta; \mu = T(S_1, S_2, \eta); \mu = id$ and $T(S_1, S_2, \mu); \mu = \mu; \mu$.

Suppose $\mathcal{C}$ has finite products. A parameterized monad is *strong* when also equipped with a family of morphisms $\tau_{A,S_1,S_2,B} : A \times T(S_1, S_2, B) \longrightarrow T(S_1, S_2, A \times B)$, natural in $A$, $B$, $S_1$, and $S_2$, subject to the natural adaptation of the axioms for non-parameterized strong monads [14].

Just as with standard monads, any parameterized monad induces a parameterized Kleisli category $\mathbf{Set}_T$, whose objects are pairs $(S \in \mathcal{S}, A \in \mathcal{C})$ and whose morphisms from $(S_1, A)$ to $(S_2, B)$ are given by $A \to T(S_1, S_2, B)$ in $\mathcal{C}$. As the standard Kleisli category of strong monads is used to model computational $\lambda$-calculus [13], the parameterized Kleisli category of strong parameterized monads is used to model Atkey's "typed command calculus".

In our case, $\mathcal{S}$ is **CPTP** and $\mathcal{C}$ is **Set**. The functor $\mathcal{I}$ in Definition 3.6 with the following structure is called the *quantum instrument monad*.

**Theorem 3.8** (Quantum Instrument Monad)**.** *$(\mathcal{I}, \eta, \mu)$ is a parameterized monad on **Set** where*

- *the unit $\eta_{H,X} : X \to \mathcal{I}(H, H, X)$ is*
$$\eta_{H,X} = X \xrightarrow{in} G(X \cdot I) \xrightarrow{G(X \cdot \lceil id_H \rceil)} G(X \cdot \boldsymbol{CPTP_{Conv}}(H, H))$$
*where $\lceil id_H \rceil$ is the convex structure of $id_H \in \boldsymbol{CPTP}(H, H)$; and*
- *the multiplication $\mu : \mathcal{I}(H, K, \mathcal{I}(K, L, X)) \to \mathcal{I}(H, L, X)$ is*
$$\mu = G(G(X \cdot C_2) \cdot C_1) \xrightarrow{G(\epsilon \cdot id)} G((X \cdot C_2) \otimes C_1) \xrightarrow{\cong} G(X \cdot (C_2 \otimes C_1)) \xrightarrow{G(X \cdot c_{H,K,L})} G(X \cdot \boldsymbol{CPTP_{Conv}}(H, L))$$
*where $C_1 = \boldsymbol{CPTP_{Conv}}(H, K)$, $C_2 = \boldsymbol{CPTP_{Conv}}(K, L)$, and $c_{H,K,L}$ is the composition of convex maps.*

We defer the proof of Theorem 3.8 to Section 3.4, where we obtain them as a consequence of a stronger structural fact, namely that $\mathcal{I}$ arises from a parameterized adjunction.

### 3.4 A Parameterized Adjunction Inducing the Monad

As standard monads arise from adjunctions, parameterized monads arise from parameterized adjunctions (Proposition 1 in [2]).

**Definition 3.9.** An *$\mathcal{S}$-parameterized adjunction* from $\mathcal{C}$ to $\mathcal{D}$ is a 4-tuple $(L, R, \eta, \epsilon)$ where

- $L : \mathcal{S} \times \mathcal{C} \to \mathcal{D}$ and $R : \mathcal{S}^{op} \times \mathcal{D} \to \mathcal{C}$ are functors, and
- $\eta_{S,A} : A \to R(S, L(S, A))$ and $\epsilon_{S,A} : L(S, R(S, A)) \to D$ are families of morphisms, natural in $A$ and dinatural in $S$, satisfying the triangle identities.

We now show that the quantum instrument monad arises from a parameterized adjunction.

The construction combines two adjunctions. The first is the free-forgetful adjunction described above.

$$F \dashv G : \mathbf{Set} \rightleftarrows \mathbf{Conv}$$

The second is the family of adjunctions between **Conv** and the **Conv**-valued presheaf category $\widehat{\mathbf{CPTP}}_{\mathbf{Conv}} = [\mathbf{CPTP}^{op}, \mathbf{Conv}]$ for each pair $H, K \in \mathbf{CPTP}$:

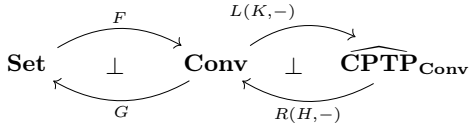
$$L(K, -) \dashv R(H, -) : \mathbf{Conv} \rightleftarrows \widehat{\mathbf{CPTP}}_{\mathbf{Conv}}$$

where the left/right adjoints are

$$L(K, A) = A \otimes \mathbf{CPTP}_{\mathbf{Conv}}(-, K),$$
$$R(H, P) = PH.$$

This second adjunction closely mirrors the enriched presheaf approach of [24], which define quantum operations as modules over $\Sigma$-monoids. In our case, the presheaf category is **Conv**-valued rather than $\Sigma$**Mon**-enriched and the base category is **CPTP** rather than **Q** of superoperators.

**Lemma 3.10.** *The family* $F; L(K,-) \dashv R(H,-); G$ *for* $H, K \in \boldsymbol{CPTP}$ *forms a* ***CPTP****-parameterized adjunction.*




By the construction of parameterized monads from parameterized adjunctions [2], Theorem 3.8 immediately follows from this lemma.

## 4 Quantum Command Calculus

### 4.1 Syntax

In this section, we introduce the *quantum command calculus*, an internal language for the QRAM model explained in Section 2.1. To capture the QRAM architecture, where a classical controller emits "recipes" to an underlying quantum store, the calculus strictly distinguishes between a pure classical part and a stateful quantum part.

This language design is achieved by Atkey's *typed command calculus*, an internal language for parameterized monads. The calculus has syntactic distinction between values and computations following Levy's Fine-grain call-by-value [11] in addition to the third syntactic category for state manipulations. The state manipulations are interpreted as morphisms of state category $\mathcal{S}$, which is **CPTP** in our case. Recalling that CPTP maps are the model of QRAM recipes, they correspond to the syntax of state manipulations. As explained in Section 2.1, we can take richer representations for recipes, but here we only expect symmetric monoidal structure for them. Assuming the state calculus to have a symmetric monoidal structure was already investigated by Atkey (Section 5 in [2]), which is called "symmetric monoidal typed command calculus".

$x, y, \ldots$ and $a, b, \ldots$ ranges over variable names and quantum system labels, respectively. Types and typing context are defined for values and state spaces, respectively.

$$\begin{aligned} A, B, \ldots &::= \top \mid A; H \to B; K \\ \Gamma &::= \cdot \mid \Gamma, x : A \\ H, K, \ldots &::= n \in \mathbb{N} \mid H \otimes K \\ \Delta &::= \cdot \mid \Delta, a : H \end{aligned}$$

Here, $H$ represents a Hilbert space of dimension $|H|$, given by $|n| = n$ and $|H \otimes K| = |H||K|$. Because systems are modeled by a symmetric monoidal category, the context of state spaces $\Delta$ is consumed linearly. Consequently, the no-cloning property of quantum data is enforced within the type system of recipes rather than that of values and computations.

Three typing judgments are defined for values, computations, and recipes, respectively:

$$\Gamma \vdash^v V : A, \quad \Gamma; \Delta \vdash^c M : A; H, \quad \Delta \vdash^r P : H.$$

The complete typing rules are summarized in Figure 1.

The quantum-specific additions are the following rules:

- Meas handles selective measurement. It measures an $n$-dimensional quantum system at $a : n$ and branches to computation $M_i$ based on the classical outcome $i \in \{1, \ldots, n\}$.
- Prim incorporates primitive quantum operations $\Psi : H \to K$, interpreted as CPTP maps $\mathcal{L}(\mathbb{C}^{|H|}) \to \mathcal{L}(\mathbb{C}^{|K|})$. This covers unitary gates, state preparation, non-selective measurements, as well as syntactic conversions of state types (e.g., merge operations $\Psi_{\text{merge}} : m \otimes n \to mn$).

### 4.2 Operational Semantics

The operational semantics is defined via a big-step reduction relation over a single execution branch. As discussed in Section 2.3, each execution branch corresponds to an individual component of a quantum instrument — that is, a quantum subchannel. Accordingly, the big-step relation records an accumulated execution trace $T$, which formally models the quantum subchannel from the system typed by $\Delta$ to that typed by $\Delta'$, as governed by the formation judgment $\Delta \vdash^t T : \Delta'$ (Figure 2).

A raw recipe $P$ lacks an explicit output system label, so $[P]_a$ binds its output to a label $a : H$.

The big-step relation is defined as

$$M \Downarrow V, T$$

stating that evaluating a closed computation $M$ yields a value $V$ and an accumulated execution trace $T$. For a measurement $\mathsf{meas}(a)[M_1, \ldots, M_n]$, evaluation non-deterministically branches into one of $n$ execution paths corresponding to outcome $i \in \{1, \ldots, n\}$, prepending the projection $\pi_i(a)$ to the resulting trace. The inference rules are presented in Figure 3. We use the informal notations $P_\square$ and $D_1[a]$ to denote the substitution of the pending output label $\square$ in $D_1$ with $a$.

Syntactic result for the big-step relation is stated in the following lemma.

**Values** $\Gamma \vdash^v V : A$

$$\textsc{Ax}\ \frac{}{\Gamma, x : A \vdash^v x : A} \qquad \top\text{I}\ \frac{}{\Gamma \vdash^v * : \top} \qquad \to\text{I}\ \frac{\Gamma, x : A; H \vdash^c M : B; K}{\Gamma \vdash^v \lambda x.M : A; H \to B; K}$$

**Computations** $\Gamma; \Delta \vdash^c M : A; K$

$$\to\text{E}\ \frac{\Gamma \vdash^v V : A; H \to B; K \qquad \Gamma \vdash^v W : A}{\Gamma; a : H \vdash^c V\ (W; a) : B; K} \qquad \textsc{Ret}\ \frac{\Gamma \vdash^v V : A \qquad \Delta \vdash^r P : H}{\Gamma; \Delta \vdash^c \mathsf{return}\ (V; P) : A; H}$$

$$\textsc{Seq}\ \frac{\Gamma; \Delta \vdash^c M : A; H \qquad \Gamma, x : A; \Delta', a : H \vdash^c N : B; K}{\Gamma; \Delta, \Delta' \vdash^c M\ \mathsf{to}\ (x; a).N : B; K} \qquad \textsc{Meas}\ \frac{\Gamma; \Delta, a : n \vdash^c M_i : A; K\ \text{for } i = 1, \ldots, n}{\Gamma; \Delta, a : n \vdash^c \mathsf{meas}(a)[M_1, \ldots, M_n] : A; K}$$

**Recipes** $\Delta \vdash^r P : H$

$$\textsc{Ax}\ \frac{}{a : H \vdash^r a : H} \qquad 1\text{I}\ \frac{}{\cdot \vdash^r I : 1} \qquad 0\text{E}\ \frac{\Delta \vdash^r P : H}{\Delta, a : 0 \vdash^r \mathsf{let}\ I = a\ \mathsf{in}\ P : H} \qquad \otimes\text{I}\ \frac{\Delta \vdash^r P : H \qquad \Delta' \vdash^r Q : H}{\Delta, \Delta' \vdash^r P \otimes Q : H \otimes K}$$

$$\otimes\text{E}\ \frac{\Delta \vdash^r P : H \otimes K \qquad \Delta', a : H, b : K \vdash^r Q : L}{\Delta, \Delta' \vdash^r \mathsf{let}\ a \otimes b = P\ \mathsf{in}\ Q : L} \qquad \textsc{Prim}\ \frac{\Delta \vdash^r P : H \qquad \Psi : H \to K}{\Delta \vdash^r \Psi\ P : K}$$

**Fig. 1 Terms for the quantum command calculus**

$$\frac{}{\Delta \vdash^t I : \Delta} \qquad \frac{\Delta \vdash^r P : H}{\Delta \vdash^t [P]_a : a : H}$$

$$\frac{1 \le i \le n}{\Delta, a : n \vdash^t \pi_i(a) : \Delta, a : n}$$

$$\frac{\Delta_1 \vdash^t T_1 : \Delta_1' \qquad \Delta_2 \vdash^t T_2 : \Delta_2'}{\Delta_1, \Delta_2 \vdash^t T_1 \otimes T_2 : \Delta_1', \Delta_2'}$$

$$\frac{\Delta \vdash^t T : \Delta' \qquad \Delta' \vdash^t T' : \Delta''}{\Delta \vdash^t T; T' : \Delta''}$$

**Fig. 2 Execution trace formation rules**

$$\frac{M[V/x] \Downarrow W, D}{(\lambda x.M)\ (V; a) \Downarrow W, D} \qquad \frac{}{\mathsf{return}\ (V; P) \Downarrow V, [P]_\square}$$

$$\frac{M \Downarrow V, D_1 \qquad N[V/x] \Downarrow W, D_2}{M\ \mathsf{to}\ (x; a).N \Downarrow N[V/x], (I \otimes D_1[a]); D_2}$$

$$\frac{M_i \Downarrow V_i, D_i}{\mathsf{meas}(a)[M_1, \ldots, M_n] \Downarrow V_i, (\pi_i(a); D_i)}\ (i = 1, \ldots, n)$$

**Fig. 3 Big-step reduction relation**

**Lemma 4.1** (Type Soundness)**.** *If* $\Gamma; \Delta \vdash^c M : A; H$ *and* $M \Downarrow D, V$*, then* $\Gamma \vdash^v V : A$ *and* $\Delta \vdash^t D : H$*.*

## 5 Denotational Semantics

### 5.1 Semantic Domains and Interpretation of Types

The semantic domains of quantum command calculus consists of three categories: $\mathcal{V}$ for pure values, $\mathcal{C}$ for stateful computations, and $\mathcal{S}$ for quantum states.

$$\mathcal{V} = \mathbf{Set}, \quad \mathcal{C} = \mathbf{Set}_{\mathcal{I}}, \quad \mathcal{S} = \mathbf{CPTP}$$

where $\mathbf{Set}_{\mathcal{I}}$ is the parameterized Kleisli category of the quantum instrument monad $\mathcal{I}$ defined in Definition 3.6. Note that the objects of $\mathbf{Set}_{\mathcal{I}}$ are pairs $(A, H)$ where $A \in \mathbf{Set}$ and $H \in \mathbf{CPTP}$ and morphisms are given by

$$\mathbf{Set}_{\mathcal{I}}((A, H), (B, K)) = \mathbf{Set}(A, \mathcal{I}(H, K, B)).$$

The interpretation of types is defined inductively as follows:

$$[\![\top]\!] = \mathbf{1}$$
$$[\![A; H \to B; K]\!] = \mathbf{Set}_{\mathcal{I}}(([\![A]\!], [\![H]\!]), ([\![B]\!], [\![K]\!]))$$
$$[\![n]\!] = \mathbb{C}^n$$
$$[\![H \otimes K]\!] = [\![H]\!] \otimes [\![K]\!]$$

where $\mathbf{1}$ is the terminal object in $\mathbf{Set}$.

The typing contexts are interpreted as cartesian products in $\mathbf{Set}$:

$$[\![\cdot]\!] \triangleq \mathbf{1}, \quad [\![\Gamma, x : A]\!] \triangleq [\![\Gamma]\!] \times [\![A]\!]$$

and the state spaces are interpreted as tensor prod-

ucts in **CPTP**:

$$[\![\cdot]\!] \triangleq I, \quad [\![\Delta, a : H]\!] \triangleq [\![\Delta]\!] \otimes [\![H]\!]$$

where $I = \mathbb{C}$ is the monoidal unit.

### 5. 2 Interpretation of Judgments

The three syntactic categories — values, computations, recipes — are interpreted as morphisms in $\mathcal{V}$, $\mathcal{C}$, and $\mathcal{S}$, respectively. The interpretation is given in full for values and recipes in Figure 4, but Seq and Meas is left for future work.

### 5. 3 Adequacy

In order to state the adequacy theorem without recursion, we add a base type $\mathsf{Bool}$ and values $\mathsf{tt}$ and $\mathsf{ff}$ to the calculus.

Let $M$ be typed as $\vdash^c M : A; H$ and The probability of evaluating to a value $V$ under an initial quantum state $\rho \in \mathcal{L}([\![\Delta]\!])$ is the sum of the traces of all non-deterministic paths leading to $V$:

$$Pr(M \Downarrow V \mid \rho) = \sum_{M \Downarrow V, T} Tr([\![T]\!](\rho))$$

The adequacy theorem asserts that the probabilistic predictions of both the operational and denotational semantics perfectly coincide.

**Theorem 5.1.** *If* $\vdash^c M : \mathsf{Bool}; H$*, then for each* $V \in \{\mathsf{tt}, \mathsf{ff}\}$*,*

$$Tr(\Phi_V(1)) = Pr(M \Downarrow V \mid 1)$$

*where* $[\![M]\!](*) = \{\Phi_V\}_{V \in \{\mathsf{tt}, \mathsf{ff}\}}$*.*

## 6 Conclusions and Prospects

This paper gave a categorical construction of a quantum instrument monad, presented independently via distributions and via an adjunction with convex spaces, and used it to sketch a denotational semantics for a QRAM-style command calculus via a parameterized Freyd category. The interpretation of sequencing and measurement, and the corresponding adequacy proof, remain to be completed; we expect the proof to follow the standard logical-relations argument for algebraic effects, adapted to track the accumulated execution trace of Section 4. 2.

A more substantial extension is general recursion. The command calculus of Section Section 4 is terminating by construction, but recursive quantum programs — e.g. repeat-until-success circuits — require a fixed-point operator on computations. Following [8], this can be obtained by equipping the parameterized Kleisli category $\mathbf{Set}_{\mathcal{I}}$ with an $\omega$-cppo structure on hom-sets and requiring the monad operations to be Scott-continuous; a least fixed-point operator on $\mathbf{Set}_{\mathcal{I}}$'s hom-sets then interprets a fix construct on computations, exactly as in the domain-theoretic treatment of the untyped lambda calculus. Since $\mathcal{I}(H, K, X)$ is built from copowers of convex spaces, transporting this $\omega$-cppo structure requires the copower construction itself to be compatible with directed joins — we leave the details, together with the resulting extension of adequacy to a fixed-point induction principle, to future work.

### References


[ 1 ] Altenkirch, T. and Green, A. S.: The Quantum IO Monad, *Semantic Techniques in Quantum Computation*, Mackie, I. and Gay, S.(eds.), Cambridge University Press, Cambridge, 2009, pp. 173–205.

[ 2 ] Atkey, R.: Parameterised Notions of Computation, *Journal of Functional Programming*, Vol. 19, No. 3-4(2009), pp. 335–376.

[ 3 ] Booth, R. I., Leichtle, D., Rice, A., and Worrall, K.: Composing Quantum Instruments, June 2026.

[ 4 ] Coecke, B. and Duncan, R.: Interacting Quantum Observables, *Automata, Languages and Programming*, Aceto, L., Damgård, I., Goldberg, L. A., Halldórsson, M. M., Ingólfsdóttir, A., and Walukiewicz, I.(eds.), Berlin, Heidelberg, Springer, 2008, pp. 298–310.

[ 5 ] Colledan, A. and Dal Lago, U.: On Dynamic Lifting and Effect Typing in Circuit Description Languages, *28th International Conference on Types for Proofs and Programs (TYPES 2022)*, Kesner, D. and Pédrot, P.-M.(eds.), Leibniz International Proceedings in Informatics (LIPIcs), Vol. 269, Dagstuhl, Germany, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, pp. 3:1–3:21.

[ 6 ] Davies, E. B. and Lewis, J. T.: An Operational Approach to Quantum Probability, *Communications in Mathematical Physics*, Vol. 17, No. 3(1970), pp. 239–260.

[ 7 ] Fritz, T.: The Quantum Instrument Monad, June 2026.

[ 8 ] Goncharov, S. and Schroder, L.: A Relatively Complete Generic Hoare Logic for Order-Enriched Effects, *Proceedings of the 2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '13, USA, IEEE Computer Society, June 2013, pp. 273–282.

[ 9 ] Jacobs, B.: *Introduction to Coalgebra: Towards Mathematics of States and Observation*, Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, Cambridge, 2016.

[10] Knill, E.: Conventions for Quantum Pseudocode, No. LA-UR–96-2724, 366453, June 1996.

**Values**

$$\frac{}{[\![\Gamma, x : A \vdash^v x : A]\!] = \pi_2 : [\![\Gamma]\!] \times [\![A]\!] \to [\![A]\!]} \qquad \frac{}{[\![\Gamma \vdash^v * : \top]\!] = ! : [\![\Gamma]\!] \to \mathbf{1}}$$

$$\frac{[\![\Gamma, x : A; H \vdash^c M : B; K]\!] = f : ([\![\Gamma]\!] \times [\![A]\!], [\![H]\!]) \to ([\![B]\!], [\![K]\!])}{[\![\Gamma \vdash^v \lambda x.M : A; H \to B; K]\!] = \Lambda(f) : [\![\Gamma]\!] \to \mathcal{C}(([\![A]\!], [\![H]\!]), ([\![B]\!], [\![K]\!]))}$$

**Computations**

$$\frac{[\![\Gamma \vdash^v V : A; H \to B; K]\!] = f : [\![\Gamma]\!] \to \mathcal{C}(([\![A]\!], [\![H]\!]), ([\![B]\!], [\![K]\!])) \qquad [\![\Gamma \vdash^v W : A]\!] = g : [\![\Gamma]\!] \to [\![A]\!]}{[\![\Gamma; a : H \vdash^c V\ (W; a) : B; K]\!] = \Lambda^{-1}(\langle f, g \rangle) : ([\![\Gamma]\!], [\![H]\!]) \to ([\![B]\!], [\![K]\!])}$$

$$\frac{[\![\Gamma \vdash^v V : A]\!] = f : [\![\Gamma]\!] \to [\![A]\!] \qquad [\![\Delta \vdash^r P : H]\!] = g : [\![\Delta]\!] \to [\![H]\!]}{[\![\Gamma; \Delta \vdash^c \mathsf{return}\ (V; P) : A; H]\!] = J(f, g) : ([\![\Gamma]\!], [\![\Delta]\!]) \to ([\![A]\!], [\![H]\!])}$$

**Recipes**

$$\frac{}{[\![a : H \vdash^r a : H]\!] = id : [\![H]\!] \to [\![H]\!]} \qquad \frac{[\![\Delta \vdash^r P : H]\!] = f : [\![\Delta]\!] \to [\![H]\!] \qquad [\![\Theta \vdash^r Q : K]\!] = g : [\![\Theta]\!] \to [\![K]\!]}{[\![\Delta, \Theta \vdash^r P \otimes Q : H \otimes K]\!] = f \otimes g : [\![\Delta]\!] \otimes [\![\Theta]\!] \to [\![H]\!] \otimes [\![K]\!]}$$

$$\frac{[\![\Delta \vdash^r P : H \otimes K]\!] = f : [\![\Delta]\!] \to [\![H]\!] \otimes [\![K]\!] \qquad [\![\Theta, a : H, b : K \vdash^r Q : L]\!] = g : [\![\Theta]\!] \otimes [\![H]\!] \otimes [\![K]\!] \to [\![L]\!]}{[\![\Delta, \Theta \vdash^r \mathsf{let}\ a \otimes b = P\ \mathsf{in}\ Q : L]\!] = (f \otimes id); \sigma; g : [\![\Delta]\!] \otimes [\![\Theta]\!] \to [\![L]\!]}$$

$$\frac{[\![\Delta \vdash^r P : H]\!] = f : [\![\Delta]\!] \to [\![H]\!] \qquad \Psi \in \mathcal{S}([\![H]\!], [\![K]\!])}{[\![\Delta \vdash^r \underline{\Psi}\ P : K]\!] = f; \Psi : [\![\Delta]\!] \to [\![K]\!]}$$

**Fig. 4 Interpretation of terms except for Seq and Meas**


[11] Levy, P., Power, J., and Thielecke, H.: Modelling Environments in Call-by-Value Programming Languages, *Information and Computation*, Vol. 185, No. 2(2003), pp. 182–210.

[12] Mac Lane, S.: *Categories for the Working Mathematician*, Graduate Texts in Mathematics, Vol. 5, Springer, New York, NY, 1971.

[13] Moggi, E.: Computational Lambda-Calculus and Monads, *Proceedings. Fourth Annual Symposium on Logic in Computer Science*, June 1989, pp. 14–23.

[14] Moggi, E.: Notions of Computation and Monads, *Information and Computation*, Vol. 93, No. 1(1991), pp. 55–92.

[15] Ozawa, M.: Quantum Measuring Processes of Continuous Observables, *Journal of Mathematical Physics*, Vol. 25, No. 1(1984), pp. 79–87.

[16] Plotkin, G. and Power, J.: Adequacy for Algebraic Effects, *Foundations of Software Science and Computation Structures*, Honsell, F. and Miculan, M.(eds.), Berlin, Heidelberg, Springer, 2001, pp. 1–24.

[17] Plotkin, G. and Power, J.: Notions of Computation Determine Monads, *Foundations of Software Science and Computation Structures*, Nielsen, M. and Engberg, U.(eds.), Berlin, Heidelberg, Springer, 2002, pp. 342–356.

[18] Sakayori, K., Colledan, A., and Dal Lago, U.: On Circuit Description Languages, Indexed Monads, and Resource Analysis, *Proc. ACM Program. Lang.*, Vol. 10, No. POPL(2026), pp. 24:688–24:717.

[19] Selinger, P.: Towards a Quantum Programming Language, *Mathematical Structures in Computer Science*, Vol. 14, No. 4(2004), pp. 527–586.

[20] Selinger, P. and Valiron, B.: A Lambda Calculus for Quantum Computation with Classical Control, *Proceedings of the 7th International Conference on Typed Lambda Calculi and Applications*, TLCA'05, Berlin, Heidelberg, Springer-Verlag, April 2005, pp. 354–368.

[21] Singhal, K., Hietala, K., Marshall, S., and Rand, R.: Q# as a Quantum Algorithmic Language, *Electronic Proceedings in Theoretical Computer Science*, Vol. 394(2023), pp. 170–191.

[22] Staton, S.: Algebraic Effects, Linearity, and Quantum Programming Languages, *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '15, New York, NY, USA, Association for Computing Machinery, January 2015, pp. 395–406.

[23] Svore, K., Geller, A., Troyer, M., Azariah, J., Granade, C., Heim, B., Kliuchnikov, V., Mykhailova, M., Paz, A., and Roetteler, M.: Q#: Enabling Scalable Quantum Computing and Development with a High-level DSL, *Proceedings of the Real World Domain Specific Languages Workshop 2018*, RWDSL2018, New York, NY, USA, Association for Computing Machinery, February 2018, pp. 1–10.

[24] Tsukada, T. and Asada, K.: Enriched Presheaf Model of Quantum FPC, *Proc. ACM Program. Lang.*, Vol. 8, No. POPL(2024), pp. 13:362–13:392.

[25] Wootters, W. K. and Zurek, W. H.: A Single Quantum Cannot Be Cloned, *Nature*, Vol. 299, No. 5886(1982), pp. 802–803.