# Algebraic, operational, and denotational semantics for classically controlled quantum communication

THEO WANG, University of Oxford, United Kingdom

SAM STATON, University of Oxford, United Kingdom

We systematically study the semantics of classically controlled quantum communication. We present a parameterised algebraic theory for classically controlled quantum I/O, and give two sound models: a quantum-stream-based operational semantics, and a monadic denotational semantics. We further show that the two models correspond to the same notion of communication in the sense that the latter is adequate and fully abstract with respect to the former.

## 1 Introduction

One of the most promising avenues of scaling quantum computing is by distributed quantum computing (e.g. [1]). When this happens, we might want to run quantum programs which send and receive qubits, like:

```
q: qbit = in(); q = apply(hadamard, q); out(q)
```

where in receives a qubit from a fixed channel, and out sends a qubit to a fixed channel. Moreover, to enable a natural programming model, we need to express classically controlled communication (sending/receiving qubits based on measurement outcomes), like:

```
p: qbit = in(); q: qbit = new(|0>);
if (measure(p) == |1>) { out(q); } else { skip; }
```

It is thus imperative to study such quantum communication primitives in the lense of programming languages. This could ultimately allow us to develop verification methods for distributed quantum programs, propose admissible compiler rewrites, and suggest good language designs. To date, quantum communication has mostly been studied in the context of quantum concurrency and, in particular, quantum process calculi [2, 4, 8, 15] (QPC), which extend classical process calculi with quantum computing primitives and quantum message-passing (sending/receiving qubits). However, the inherent complexity of concurrency makes it difficult to isolate the challenges of communication itself.

In this work (preprint at [14]), we study the individual operations of quantum communication *in isolation* as algebraic effects, following the tradition of Plotkin and Power [11]. We propose a new algebraic theory for quantum communication by building on [13] a complete theory for quantum computing. We then study two, complementary models of the theory. The first one is an operational semantics (Section 3.1), demonstrating the computational model we have in mind. The second one is an elegant monadic denotational

Authors' Contact Information: Theo Wang, University of Oxford, Oxford, United Kingdom, theo_ck_wang@hotmail.com; Sam Staton, University of Oxford, Oxford, United Kingdom, sam.staton@cs.ox.ac.uk.

semantics (Section 3.2). Finally, we show that the latter is adequate and fully abstract with respect to the former (Theorem 3.2).

## 2 An Algebraic Theory for Quantum Communication

Our starting point is [13], a semantics for qubit quantum computation *without* communication as a parameterised algebraic theory (a collection of quantum operations along with an equational theory). The language of well-formed terms is defined by the judgement $\Gamma \mid \Delta \vdash c$, where $\Delta = q_1...q_p$ is a list of parameters (qubit names) and $\Gamma = x_1 : m_1...x_k : m_k$ is a list of computation variables (continuations) along with their valences (the number of qubits each continuation takes). The judgement is the least one closed under the following:

$$\frac{(x : p) \in \Gamma}{\Gamma \mid a_1...a_p \vdash x(a_1...a_p)} \qquad \frac{\Gamma \mid \Delta \vdash t}{\Gamma \mid \pi(\Delta) \vdash t} \qquad \frac{\{\Gamma \mid \Delta, b_1...b_{m_i} \vdash t_i\}_{i=1...k} \qquad \mathbf{op} : (p \mid m_1...m_k)}{\Gamma \mid \Delta, a_1...a_p \vdash \mathbf{op}(a_1...a_p; \{b_1...b_{m_i}.t_i\}_i)}$$

where $\pi$ is a permutation. Intuitively, we think of such a term $c$ as a computation which takes $p$ qubits, classically branches, and gives one of $m_1...m_k$ qubits; we can assign it a 'type' $(p \mid m_1...m_k)$. This algebraic syntax is convenient because it avoids syntactic sugar, but it can be straightforwardly made into ordinary programming syntax (see [13] Section 5). Finally, theories can be combined: the disjoint union of two algebraic theories is called the *sum* of theories, and the commutative combination (such that every operation in one commutes with every operation in the other) is called the *tensor* [6, 14].

The $\mathsf{QUANTUM}$ theory is the one defined by operations for state preparation $\mathsf{new} : (0 \mid 1)$, unitary evolution $\mathsf{apply}_U : (n \mid n)$ for each $2^n \times 2^n$ unitary $U$, and measurement + classical control $\mathsf{measure} : (1 \mid 0, 0)$, along with suitable equations. For example, the following equation capturing the principle of deferred measurement, where for any unitaries $U_0, U_1$, $D(U_0, U_1)$ is the controlled unitary such that $D(U_0, U_1)|i\rangle|\psi\rangle = |i\rangle|U_i\psi\rangle$ for $i = 0, 1$.

$$\begin{aligned}&\mathsf{measure}(a, \mathsf{apply}_U(\vec{b}, \vec{b}.x(\vec{b}), \mathsf{apply}_V(\vec{b}, \vec{b}.y(\vec{b})))) \\ &\qquad = \mathsf{apply}_{D(U,V)}((a, \vec{b}), (a, \vec{b}).\mathsf{measure}(a, x(\vec{b}), y(\vec{b}))) \qquad (1)\end{aligned}$$

We now build the theory for quantum communication. We extend $\mathsf{QUANTUM}$ with an operation for receiving qubits, $\mathsf{in} : (0 \mid 1)$, and an operation for sending qubits $\mathsf{out} : (1 \mid 0)$. This way, we are able to express the example programs as $x : 0 \mid \cdot \vdash \mathsf{in}(q.\mathsf{apply}_H(q, q.\mathsf{out}(q, x)))$ and $x : 0 \mid \cdot \vdash \mathsf{in}(p.\mathsf{new}(q.\mathsf{measure}(p, x, \mathsf{out}(q, x))))$, respectively. Furthermore, we add equations such that $\mathsf{in}$ and $\mathsf{out}$ commute with every existing operation in $\mathsf{QUANTUM}$. We thus obtain $\mathsf{I/O} \otimes \mathsf{QUANTUM}$, the tensor of the $\mathsf{QUANTUM}$ theory with the free $\mathsf{I/O}$ theory, which will be our main object of study. The commutativity equations are crucial to making the I/O operations compatible with quantum theory. For example, the equation

$$\mathsf{measure}(a, \mathsf{out}(b, x), \mathsf{out}(b, y)) = \mathsf{out}(b, \mathsf{measure}(a, x, y)) \qquad (2)$$

which makes $\mathsf{out}$ commute with $\mathsf{measure}$ is crucial to proving the following equation, which is a slight tweak of a well-known equivalence in quantum theory:

$$\begin{aligned}&\mathsf{new}(q.\mathsf{apply}_H(q, q.\mathsf{new}(p.\mathsf{apply}_{CX}((q, p), (q, p).\mathsf{out}(p, \mathsf{measure}(q, x, x)))))) \\ &= \mathsf{new}(q.\mathsf{apply}_H(q, q.\mathsf{measure}(q, \mathsf{new}(p.\mathsf{out}(p, x)), \mathsf{new}(p.\mathsf{apply}_X(p, p.\mathsf{out}(p, x))))))) \qquad (3)\end{aligned}$$

## 3 Two Models

We now study two of I/O ⊗ QUANTUM's sound models, one operational, one denotational. The point is to give insight into the theory itself and connect to other methods. Moreover, there is a risk that tensors of theories collapse (e.g. see the Eckmann-Hilton argument [9]) and so our models show that there is no collapse in this setting.

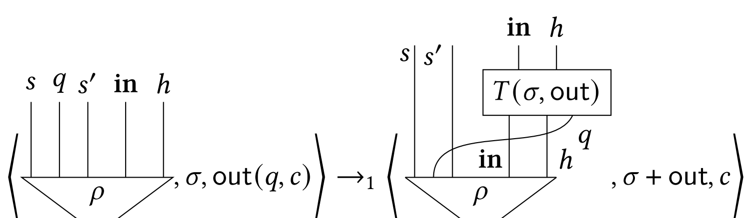
### 3.1 Operational Semantics

*Strategy.* The operational semantics we give to the terms of I/O ⊗ QUANTUM is based on a *quantum stream* used to handle I/O. The idea is to have a global quantum state $\rho : \mathbb{C} \to S \otimes \mathbf{qbit} \otimes H$, where $S$ is the state accessible to the program or the *program state*, and $H$ is the state accessible to the stream, or the *stream/hidden state*. Then, we define a stream as a state machine which reacts to the program requesting an input qubit or presenting an output qubit.

*Category of CP maps.* We interpret quantum states and processes as morphisms in the category **CP**, which is the finite biproduct completion of **CPM** [12], the category of (finite dimensional) matrix algebras and complete positive maps. In particular, we write $\mathbf{qbit} \triangleq \mathcal{M}_2(\mathbb{C})$ as the space of qubits. Conveniently, **CP** is a biproduct compact closed category, allowing us to note its morphisms as string diagrams (cf. [3, 5, 14]). Its monoidal structure defined by the tensor product ⊗ and its unit $I$ defined as the space of 1D matrices $\mathbb{C}$. CP maps $\mathbb{C} \to A$ are called 'states' and correspond to (unnormalised) density operators, while morphisms $A \to \mathbb{C}$ are called 'effects' and correspond to the positive quantum observables.

*Quantum Stream.* Let $X_0 \in \mathbf{CP}$ be the shape of the initial hidden state. Then a stream $t$ in $\mathsf{Stream}(X_0)$ consists of: a map $q : \{\mathsf{in}, \mathsf{out}\}^* \to \mathbf{Ob}\ \mathbf{CP}$ (such that $q(\epsilon) = X_0$) defining the shape of stream state at every step; and for every $\sigma \in \{\mathsf{in}, \mathsf{out}\}^*$, trace-preserving CP maps $T(\sigma, \mathsf{in}) : q(\sigma) \to \mathbf{qbit} \otimes q(\sigma + \mathsf{in})$ and $T(\sigma, \mathsf{out}) : \mathbf{qbit} \otimes q(\sigma) \otimes \mathbf{qbit} \to \mathbf{qbit} \otimes q(\sigma + \mathsf{out})$ which determine the evolution of the stream state after an I/O operation.

*Operational Semantics.* A configuration is a triple $\langle \rho, \sigma, c \rangle$, where $\rho$ is a normalised global state along with an list of qubit labels $\Delta_\rho$, $\sigma \in \{\mathsf{in}, \mathsf{out}\}^*$ is the current I/O trace, and $c$ is the running program. Given a stream $t \in \mathsf{Stream}(X_0)$, the small-step operational semantics is given by a transition relation $\to_p^t$ (where $p$ indicates the transition probability), defined inductively with respect to the structure of the program in a standard way. A term is evaluated probabilistically until a continuation call is reached in which case the program terminates. For example, for the output operation we have the following rule




(out)

which performs the output and updates the stream state. The full definition is given in [14] Figure 4.4.

*Program Equivalence.* We say that $\Gamma \mid \Delta \vdash c_1 \simeq_{qu} c_2$ if for all $X_0 \in \mathbf{CP}$, $t = (q, T) \in \mathsf{Stream}(X_0)$, $\sigma_0 \in \{\mathsf{in}, \mathsf{out}\}^*$, and $\rho : \mathbb{C} \longrightarrow \mathbf{qbit}^{\otimes|\Delta|} \otimes \mathbf{qbit} \otimes q(\sigma_0)$,

$$\sum_{\psi} \Pr(\langle \rho, \sigma, c_1 \rangle \longrightarrow^* \langle \psi, \sigma_f, x(\Delta_\psi) \rangle) \cdot \psi = \sum_{\psi} \Pr(\langle \rho, \sigma, c_2 \rangle \longrightarrow^* \langle \psi, \sigma_f, x(\Delta_\psi) \rangle) \cdot \psi \tag{4}$$

for any $(x : n) \in \Gamma$ and I/O trace $\sigma_f \in \{\mathsf{in}, \mathsf{out}\}^*$. Intuitively, this is the 'right' definition because it sums over all the execution paths which give the same 'type level' outcome, i.e. presenting the same I/O trace and calling the same continuation. And indeed we have:

**Theorem 3.1** (Operational Semantics gives a model ([14] Theorem 4.3.25))**.** *Terms of* $\mathsf{I/O} \otimes \mathsf{QUANTUM}$ *modulo* $\simeq_{qu}$ *gives a sound model of* $\mathsf{I/O} \otimes \mathsf{QUANTUM}$.

Rather interestingly, our program equivalence has a 'denotational' flavour: it relies on introspecting the program states at termination instead of using a syntactic, contextual equivalence. This motivates seeking for a denotational semantics.

### 3.2 Denotational Semantics

*Strategy.* Following the traditions of algebraic theories of effects [11], we construct a monad and interpret $\mathsf{I/O} \otimes \mathsf{QUANTUM}$ in its Kleisli category. In particular, our theory is a *tensor* between $\mathsf{I/O}$ and $\mathsf{QUANTUM}$. Therefore, instead of using the standard resumptions monad [6] (which corresponds to a sum of algebraic theories), we construct a free I/O monad ($T(X) = \mu Y.[I, Y] + (O \otimes Y) + X$) directly on top of $\mathbf{CP}$, which in some sense embodies the models of $\mathsf{QUANTUM}$.

*Infinite biproducts for* $\mathbf{CP}$. To enable the definition of an I/O monad in $\mathbf{CP}$, we need to extend it with infinite coproducts (and thus biproducts). To this end, we construct $\mathbf{CP}^\infty$ as the infinite biproduct completion of $\mathbf{CPM}$, following [10]. $\mathbf{CPM}$ is first completed to dcpo-enriched category $\overline{\mathbf{CPM}}$ using [16]. $\overline{\mathbf{CPM}}$ has infinite indexed sums of morphisms $\sum_{i \in I} f_i$, defined as $\bigsqcup_{S \subseteq I}(\sum_{s \in S} f_s)$. Then, $\mathbf{CP}^\infty$ is constructed as the infinite biproduct completion of $\overline{\mathbf{CPM}}$, and can be shown to inherit compact closure from $\mathbf{CPM}$ (see [10] section 4).

*The Quantum I/O Monad.* We now define our monad for qubit I/O, $T : \mathbf{CP}^\infty \longrightarrow \mathbf{CP}^\infty$, and obtain an explicit characterisation by unrolling the fixed point and noting that $[\mathbf{qbit}^{\mathsf{in}}, Y] \cong \mathbf{qbit}^{\mathsf{in}} \otimes Y$ by compact closure and self-duality.

$$T(X) \triangleq \mu Y.[\mathbf{qbit}^{\mathsf{in}}, Y] \oplus (\mathbf{qbit}^{\mathsf{out}} \otimes Y) \oplus X = \bigoplus_{\sigma \in \{\mathsf{in}, \mathsf{out}\}^*} \mathsf{Aux}(\sigma) \otimes X \tag{5}$$

where $\mathsf{Aux}(\epsilon) \triangleq \mathbb{C}$ and for $s \in \{\mathsf{in}, \mathsf{out}\}$, $\mathsf{Aux}(s + \sigma) = \mathbf{qbit}^s \otimes \mathsf{Aux}(\sigma)$. Note that the distinction between $\mathbf{qbit}^{\mathsf{in}}$ and $\mathbf{qbit}^{\mathsf{out}}$ is purely for our convenience and has no mathematical meaning. This definition makes intuitive sense: the trace of I/O operations performed in a computation determine exactly how many input qubits it depends on, and how many output qubits it generates, and the order in which they occur. Rather interestingly, however, the inputs seem to appear 'on the wrong side'. If we consider a state over $T(X)$ (i.e. a morphism $\rho : \mathbb{C} \longrightarrow T(X)$) and only look at its component for a particular trace, say $\sigma = (\mathsf{in}, \mathsf{out}, \mathsf{in})$, we get a state over $\mathbf{qbit}^{\mathsf{in}} \otimes \mathbf{qbit}^{\mathsf{out}} \otimes \mathbf{qbit}^{\mathsf{in}} \otimes X$, wherein the input qubits appear on the output side, instead of a map of the form $\mathbf{qbit}^{\mathsf{in}} \otimes \mathbf{qbit}^{\mathsf{in}} \to \mathbf{qbit}^{\mathsf{out}} \otimes X$. However, this is fine because of the Choi-Jamiołkowski isomorphism (equivalently the compact closure of $\mathbf{CP}^{\infty}$):

$$\mathbf{CP}^{\infty}(\mathbb{C}, \mathbf{qbit}^{\mathsf{in}} \otimes \mathbf{qbit}^{\mathsf{out}} \otimes \mathbf{qbit}^{\mathsf{in}} \otimes X) \cong \mathbf{CP}^{\infty}(\mathbf{qbit}^{\mathsf{in}} \otimes \mathbf{qbit}^{\mathsf{in}}, \mathbf{qbit}^{\mathsf{out}} \otimes X). \tag{6}$$

In fact, we can 'bend the input wires' back and forth by using the (unnormalised) maximally entangled state ($|\Phi\rangle\langle\Phi|$) and its adjoint CP map ($M \mapsto \langle\Phi|M|\Phi\rangle$) where $\Phi = \sum_i |ii\rangle$.

*Monadic Denotational Semantics.* Firstly we define $[\![n]\!] \triangleq \mathbf{qbit}^{\otimes n}$. Then, parameter contexts are interpreted as $[\![\Delta]\!] \triangleq [\![|\Delta|]\!]$ and computation contexts $\Gamma = x_1 : n_1 \ldots x_k : n_k$ as $[\![\Gamma]\!] \triangleq \bigotimes_{i=1}^{k} [\![n_i]\!]$. Next, we interpret a term $\Gamma \mid \Delta \vdash c$ as a morphism in the Kleisli category, $[\![c]\!] : [\![\Delta]\!] \to T([\![\Gamma]\!])$, where

$$T([\![\Gamma]\!]) = \bigoplus_{\sigma} \mathsf{Aux}(\sigma) \otimes [\![\Gamma]\!] \cong \bigoplus_{\substack{\sigma \in \{\mathsf{in},\mathsf{out}\}^* \\ (x:n) \in \Gamma}} \mathsf{Aux}(\sigma) \otimes [\![n]\!]. \tag{7}$$

Each operation can also be naturally interpreted according to their type (cf. [14] Figure 5.1). The interesting case is once again the input operator: $[\![\mathsf{in}]\!] : \mathbb{C} \to T(\mathbf{qbit})$ is defined as

$$[\![\mathsf{in}]\!] \triangleq \mathsf{inj}^{\mathsf{in}} \circ \left[ \mathbf{qbit}^{\mathsf{in}} \quad \cup \quad \mathbf{qbit} \right] \tag{8}$$

where the 'cup' corresponds here to the (unnormalised) maximally entangled state.

*Adequacy and Full abstraction* We can understand $[\![c]\!]$ as a collection of maps of the form $[\![c]\!]_{\sigma,(x:n)} : [\![\Delta]\!] \to \mathsf{Aux}(\sigma) \otimes [\![n]\!]$, indexed by the I/O trace $\sigma$ and the continuation $x$ that the program ends up calling. This is particularly neat as it corresponds exactly to the program equivalence previously defined: two program are equal iff they are equal on the sums of all reduction paths resulting in the same I/O trace $\sigma$ and continuation call $x$. This exact insight allows us to prove the following theorem:

**Theorem 3.2** (Adequacy and Full Abstraction ([14] Theorems 5.3.4 and 5.3.6))**.** *For all* $\Gamma \mid \Delta \vdash c_1, c_2$, $\Gamma \mid \Delta \vdash c_1 \simeq_{qu} c_2$ *if and only if* $[\![c_1]\!] = [\![c_2]\!]$.

As a corollary, it follows that our denotational semantics form a model of $\mathsf{I/O} \otimes \mathsf{QUANTUM}$:

**Theorem 3.3** (Denotational Semantics gives a model ([14] Corollary 5.4.2))**.** *The object* $\mathbb{C}$ *forms a model of* $\mathsf{I/O} \otimes \mathsf{QUANTUM}$ *in the opposite of the Kleisli category of* $T$*.*

## 4 Conclusion and Future Work

We proposed the first algebraic theory for classically controlled qubit quantum communication, $\mathsf{I/O} \otimes \mathsf{QUANTUM}$, and gave two complementary models: a quantum-stream-based operational semantics and an adequate and fully abstract monadic denotational semantics. A next step is to understand how this work relates to quantum process calculi [2, 4, 8, 15] and higher order quantum computing [7, 10].

## References


[1] Marcello Caleffi, Michele Amoretti, Davide Ferrari, Jessica Illiano, Antonio Manzalini, and Angela Sara Cacciapuoti. 2024. Distributed quantum computing: A survey. *Computer Networks* 254 (Dec. 2024), 110672. https://doi.org/10.1016/j.comnet.2024.110672

[2] Lorenzo Ceragioli, Fabio Gadducci, Giuseppe Lomurno, and Gabriele Tedeschi. 2024. Quantum Bisimilarity via Barbs and Contexts: Curbing the Power of Non-deterministic Observers. *Proc. ACM Program. Lang.* 8, POPL, Article 43 (jan 2024), 29 pages. https://doi.org/10.1145/3632885

[3] Bob Coecke and Aleks Kissinger. 2017. *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press.

[4] Simon Gay and Rajagopal Nagarajan. 2004. Communicating Quantum Processes. arXiv:quant-ph/0409052 [quant-ph] https://arxiv.org/abs/quant-ph/0409052

[5] Chris Heunen and Jamie Vicary. 2019. *Categories for Quantum Theory: An Introduction*. Oxford University Press. https://doi.org/10.1093/oso/9780198739623.001.0001 arXiv:https://academic.oup.com/-book/43710/book-pdf/50991591/9780191060069_web.pdf

[6] Martin Hyland, Gordon Plotkin, and John Power. 2006. Combining effects: Sum and tensor. *Theoretical Computer Science* 357, 1 (2006), 70–99. https://doi.org/10.1016/j.tcs.2006.03.013 Clifford Lectures and the Mathematical Foundations of Programming Semantics.

[7] Aleks Kissinger and Sander Uijlen. 2019. A categorical semantics for causal structure. *Logical Methods in Computer Science* 15 (2019).

[8] Marie Lalire and Philippe Jorrand. 2004. A Process Algebraic Approach to Concurrent and Distributed Quantum Computation: Operational Semantics. arXiv:quant-ph/0407005 [quant-ph] https://arxiv.org/abs/quant-ph/0407005

[9] nLab authors. 2024. Eckmann-Hilton argument. https://ncatlab.org/nlab/show/Eckmann-Hilton+argument. Revision 37.

[10] Michele Pagani, Peter Selinger, and Benoît Valiron. 2013. Applying quantitative semantics to higher-order quantum computing. *CoRR* abs/1311.2290 (2013). arXiv:1311.2290 http://arxiv.org/abs/1311.2290

[11] Gordon Plotkin and John Power. 2002. Notions of Computation Determine Monads. In *Foundations of Software Science and Computation Structures*, Mogens Nielsen and Uffe Engberg (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 342–356.

[12] Peter Selinger. 2007. Dagger Compact Closed Categories and Completely Positive Maps: (Extended Abstract). *Electronic Notes in Theoretical Computer Science* 170 (2007), 139–163. https://doi.org/10.1016/j.entcs.2006.12.018 Proceedings of the 3rd International Workshop on Quantum Programming Languages (QPL 2005).

[13] Sam Staton. 2015. Algebraic Effects, Linearity, and Quantum Programming Languages. In *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (Mumbai, India) *(POPL '15)*. Association for Computing Machinery, New York, NY, USA, 395–406. https://doi.org/10.1145/2676726.2676999

[14] Théo Wang. 2024. *Algebraic Theories and Quantum Communication*. Master's thesis. University of Oxford, Oxford, United Kingdom. https://theo.wang/works/2024-09-MSc-thesis

[15] Mingsheng Ying, Yuan Feng, Runyao Duan, and Zhengfeng Ji. 2010. An Algebra of Quantum Processes. arXiv:0707.0330 [quant-ph] https://arxiv.org/abs/0707.0330

[16] Dongsheng Zhao and Taihe Fan. 2010. Dcpo-completion of posets. *Theoretical Computer Science* 411, 22 (2010), 2167–2173. https://doi.org/10.1016/j.tcs.2010.02.020