Programming with Quantum Communication

Anya Tafliovich[1,2]

Computer Science
University of Toronto
Toronto, Canada

Eric C. R. Hehner[3]

Computer Science
University of Toronto
Toronto, Canada

Abstract

This work develops a formal framework for specifying, implementing, and analysing quantum communication protocols. We provide tools for developing simple proofs and analysing programs which involve communication, both via quantum channels and exhibiting the LOCC (local operations, classical communication) paradigm.

Keywords: Quantum Computing, Quantum Communication Protocol, Formal Verification, Formal Methods of Program Design

1 Introduction

The term quantum communication refers to the process of transferring a quantum state between distinct physical locations. There are two ways of accomplishing this task. The first one is analogous to classical communication and involves sending a quantum bit over a quantum communication channel (just as classical communication is associated with sending classical bits over a classical communication channel). The second one has no classical analogue. In a quantum world it is possible to transfer a quantum bit without utilising a quantum channel, by using a classical communication channel and a pair of entangled states and applying quantum operations locally.

This work develops a formal framework for specifying, implementing, and analysing quantum communication protocols. We provide tools for developing sim-

[1] This work is in part supported by NSERC.

[2] Email: anya@cs.toronto.edu

[3] Email: hehner@cs.toronto.edu

ple proofs and analysing programs which involve communication, both via quantum channels and exhibiting the LOCC (local operations, classical communication) paradigm. We look at quantum communication in the context of formal methods of program development, or programming methodology. This is the field of computer science concerned with applications of mathematics and logic to software engineering tasks. In particular, the formal methods provide tools to formally express specifications, prove correctness of implementations, and reason about various properties of specifications (e.g. implementability) and of implementations (e.g. time and space complexity).

In this work the analysis of quantum communication protocols is based on quantum predicative programming ([23,24,22]), a recent generalisation of the well-established predicative programming ([14,15]). It supports the style of program development in which each programming step is proved correct as it is made. We inherit the advantages of the theory, such as its generality, simple treatment of recursive programs, and of time and space complexity. The theory of quantum programming provides tools to write both classical and quantum specifications, develop quantum programs that implement these specifications, and reason about their comparative time, space, and communication complexity, all in the same framework.

There has been a number of proposals for formal approaches to quantum programming, including the language qGCL [20,28], process algebraic approaches of [4,18,17], tools developed in the field of category theory [1,2,3,10,21], functional languages of [6,7,5,25], as well as work of [12], [11], and [13]. A detailed discussion of the work related to quantum predicative programming is presented in [23].

The contribution of this work is twofold. Firstly, we present a framework for implementing quantum communication protocols, specifying desired properties of the protocols, and formally proving whether these properties hold. The properties are not restricted to reasoning about the data sent or received by the parties involved. We provide tools to prove properties which deal with the complexity of the protocol, such as the number of classical and quantum bits sent during its execution. Secondly, the reasoning about quantum communication fits nicely in the general framework of quantum predicative programming, and thus inherits all of its advantages. The definitions of specification and program are simple: a specification is a boolean (or probabilistic) expression and a program is a specification. The treatment of recursion is simple: there is no need for additional semantics of loops. The treatment of termination simply follows from the introduction of a time variable; if the final value of the time variable is ∞, then the program is a non-terminating one. There is a uniform method for proving correctness and time, space, and communication complexity; moreover, after proving them separately, we naturally obtain the conjunction. The use of Dirac-like notation makes it easy to write down specifications and develop algorithms. Finally, the treatment of computation with mixed states does not require any additional mechanisms.

The rest of this work is organised as follows. Section 2 is a brief introduction to quantum predicative programming. The contribution of this work is Section 3 which introduces a formal framework for specifying, implementing, and analysing quantum communication protocols and presents the analysis of two such protocols: quantum teleportation and quantum dense coding. Section 4 states conclusions and

outlines directions for future research. A short introduction to quantum computing is presented in the Appendix A.

2 Quantum Predicative Programming

This section introduces the programming theory of our choice — quantum predicative programming. We briefly introduce parts of the theory necessary for understanding Section 3 of this work. For a course in predicative programming the reader is referred to [14]. An introduction to probabilistic predicative programming can be found in [15,16]. Quantum predicative programming is developed in [23,24,22].

2.1 Predicative programming

In predicative programming a specification is a boolean expression. The variables in a specification represent the quantities of interest, such as prestate (inputs), poststate (outputs), and computation time and space. We use primed variables to describe outputs and unprimed variables to describe inputs. For example, specification $x' = x + 1$ states that the final value of x is its initial value plus 1. A computation *satisfies* a specification if, given a prestate, it produces a poststate, such that the pair makes the specification true. A specification is *implementable* if for each input state there is at least one output state that satisfies the specification.

We use standard logical notation for writing specifications: $\wedge$ (conjunction), $\vee$ (disjunction), $\Rightarrow$ (logical implication), $=$ (equality, boolean equivalence), $\neq$ (non-equality, non-equivalence), and **if then else**. The larger operators $\mathrel{==}$, $\leq$, and $\Longrightarrow$ are the same as $=$, $\leq$, and $\Rightarrow$, but with lower precedence. We use standard mathematical notation, such as $+ \; - \; \times \, / \, mod \, div$. We use lowercase letters for variables of interest and uppercase letters for specifications.

In addition to the above, we use the following notations: σ (prestate), σ' (poststate), ok ($\sigma' = \sigma$), and $x := e$ ($x' = e \wedge y' = y \wedge \ldots$). The notation ok specifies that the values of all variables are unchanged. In the assignment $x := e$, x is a state variable (unprimed) and e is an expression (in unprimed variables) in the domain of x.

If R and S are specifications in variables $x, y, \ldots$, then the *sequential composition* of R and S is defined by

$$R \,;\, S \mathrel{==} \exists x'', y'', \ldots \cdot R'' \wedge S'' \tag{1}$$

where R'' is obtained from R by substituting all occurrences of primed variables $x', y', \ldots$ with double-primed variables $x'', y'', \ldots$, and S'' is obtained from S by substituting all occurrences of unprimed variables $x, y, \ldots$ with double-primed variables $x'', y'', \ldots$.

Various laws can be proved about sequential composition. One of the most important ones is the substitution law, which states that for any expression e of the prestate, state variable x, and specification P,

$$x := e \,;\, P \mathrel{==} (\text{for } x \text{ substitute } e \text{ in } P) \tag{2}$$

Specification S *is refined by* specification P if and only if S is satisfied whenever P is satisfied, that is $\forall \sigma, \sigma' \cdot S \Leftarrow P$. Given a specification, we are allowed to implement an equivalent specification or a stronger one.

A *program* is an implemented specification. A good basis for classical (non-quantum) programming is provided by: *ok*, assignment, **if then else**, sequential composition, booleans, numbers, bunches, and functions. Given a specification S, we proceed as follows. If S is a program, there is no work to be done. If it is not, we build a program P, such that P refines S, i.e. $S \Leftarrow P$. The refinement can proceed in steps: $S \Leftarrow \ldots \Leftarrow R \Leftarrow Q \Leftarrow P$.

In $S \Leftarrow P$ it is possible for S to appear in P. No additional rules are required to prove the refinement. For example, it is trivial to prove that

$$x \geq 0 \Rightarrow x' = 0 \Longleftarrow \textbf{if } x = 0 \textbf{ then } ok \textbf{ else } (x := x - 1 \ ; \ x \geq 0 \Rightarrow x' = 0)$$

The specification says that if the initial value of x is non-negative, its final value must be 0. The solution is: if the value of x is zero, do nothing, otherwise decrement x and repeat.

2.2 Probabilistic predicative programming

A *probability* is a real number between 0 and 1, inclusive. A *distribution* is an expression whose value is a probability and whose sum over all values of variables is 1. Given a distribution of several variables, we can sum out some of the variables to obtain a distribution of the rest of the variables.

To generalise boolean specifications to probabilistic specifications, we use 1 and 0 both as numbers and as boolean *true* and *false*, respectively. If R and S are specifications in variables $x, y, \ldots$, then the definition (1) of *sequential composition* of R and S is generalised to

$$R \ ; \ S == \sum x'', y'', \ldots \cdot R'' \times S''$$

where R'' and S'' are defined as before.

If p is a probability and R and S are distributions, then

$$\textbf{if } p \textbf{ then } R \textbf{ else } S == p \times R + (1 - p) \times S$$

If S is an implementable deterministic specification and p is a distribution of the initial state $x, y, ...$, then the distribution of the final state is

$$p' \ ; \ S$$

Various laws can be proved about sequential composition. One of the most important ones, the substitution law, introduced earlier, applies to probabilistic specifications as well.

To implement a boolean specification S, we need to provide a program P which is either equivalent to or is stronger than S, that is $S \Leftarrow P$. In developing probabilistic programs, the analog of $\Rightarrow$ is $\leq$. However, one must be careful when using it.

Consider, for example, the following probabilistic expression:

$$(x' = 0)/2 + (x' \neq 0)/2$$

which is intended to capture the specification that says that the final value of x is 0 half of the time and non-zero half of the time. We can refine this expression with the following program (in one variable x):

$$\begin{array}{rl} & \textbf{if } 1/2 \textbf{ then } x{:=}0 \textbf{ else } x{:=}1 \\ == & (x' = 0)/2 + (x' = 1)/2 \\ \leq & (x' = 0)/2 + (x' \neq 0)/2 \end{array}$$

However, it is also the case that

$$\begin{array}{rl} & \textbf{if } 1/2 \textbf{ then } x{:=}1 \textbf{ else } x{:=}2 \\ == & (x' = 1)/2 + (x' = 2)/2 \\ \leq & 1/2 \\ == & (x' = 0)/2 + (x' \neq 0)/2 \end{array}$$

which does not correspond to the intended meaning of the specification. There is an intrinsic problem with formulating probabilistic specifications as super-distributions, known as a convex closure problem. We avoid this problem in this work by only allowing the following uses of super-distributions in probabilistic refinement: if P_0 is a probabilistic specification (a distribution) in variables σ_0' and P_1 is a distribution in variables σ_1' then we allow a probabilistic analog of weakening :

$$\begin{array}{l} P_0 \times P_1 \leq P_0 \text{ and} \\ P_0 \times P_1 \leq P_1 \end{array}$$

2.3 Quantum Predicative Programming

Let $\mathbb{C}$ be the set of all complex numbers with the absolute value operator $|\cdot|$ and the complex conjugate operator *. Then a state of an n-qubit system is a function $\psi : 0, ..2^n \to \mathbb{C}$, such that $\sum x : 0, ..2^n \cdot |\psi x|^2 == 1$. Here notation $i, ..j$ means from (and including) i to (and excluding) j.

If ψ and ϕ are two states of an n-qubit system, then their *inner product*, denoted by $\langle \psi | \phi \rangle$, is defined by:

$$\langle \psi | \phi \rangle = \sum x : 0, ..2^n \cdot (\psi x)^* \times (\phi x)$$

A *basis* of an n-qubit system is a collection of 2^n quantum states $b_{0,..2^n}$, such that $\forall i, j : 0, ..2^n \cdot \langle b_i | b_j \rangle = (i = j)$. We adopt the following Dirac-like notation for the computational basis: if x is from the domain $0, ..2^n$, then $\mathbf{x}$ denotes the corresponding n-bit binary encoding of x and $|\mathbf{x}\rangle : 0, ..2^n \to \mathbb{C}$ is the following quantum state:

$$|\mathbf{x}\rangle = \lambda i : 0, ..2^n \cdot (i = x)$$

where $\lambda x : D \cdot b$ is a function of a variable x with domain D and body b. If ψ is a state of an m-qubit system and ϕ is a state of an n-qubit system, then $\psi \otimes \phi$, the tensor product of ψ and ϕ, is the following state of a composite $m+n$-qubit system:

$$\psi \otimes \phi = \lambda i : 0, ..2^{m+n} \cdot \psi(i \; div \; 2^n) \times \phi(i \; mod \; 2^n)$$

We write $\phi^{\otimes n}$ to mean "ϕ tensored with itself n times". An operation defined on an n-qubit quantum system is a higher-order function, whose domain and range are maps from $0, ..2^n$ to the complex numbers. An *identity* operation on a state of an n-qubit system is defined by

$$I^n = \lambda \psi : 0, ..2^n \to \mathbb{C} \cdot \psi$$

For a linear operation A, the *adjoint* of A, written $A^\dagger$, is the (unique) operation, such that for any two states ψ and ϕ, $\langle \psi | A\phi \rangle = \langle A^\dagger \psi | \phi \rangle$.

The *unitary transformations* that describe the evolution of an n-qubit quantum system are operations U defined on the system, such that $U^\dagger U = I^n$.

In this setting, the *tensor product* of operators is defined in the usual way. If ψ is a state of an m-qubit system, ϕ is a state of an n-qubit system, and U and V are operations defined on m and n-qubit systems, respectively, then the tensor product of U and V is defined on an $m+n$ qubit system by

$$(U \otimes V)(\psi \otimes \phi) = (U\psi) \otimes (V\phi)$$

To apply an operation U defined on a 1-qubit system to qubit i in a composite n-qubit system, we apply the operation U_i^n to the entire system, where U_i^n is defined by:

$$U_i^n = \underbrace{I \otimes ... \otimes I}_{i} \otimes U \otimes \underbrace{I \otimes ... \otimes I}_{n-i-1}$$

Suppose we have a system of n qubits in state ψ and we measure (observe) it. Suppose also that we have a variable r from the domain $0, ..2^n$, which we use to record the result of the measurement, and variables $x, y, \ldots$, which are not affected by the measurement. Then the measurement corresponds to a probabilistic specification that gives the probability distribution of ψ' and r' (these depend on ψ and on the type of measurement) and states that the variables $x, y, \ldots$ are unchanged.

For a general quantum measurement described by a collection $M = M_{0,..2^n}$ of measurement operators, which satisfy the completeness equation (see Appendix A), the specification is $\mathbf{measure}_M \, \psi \, r$, where

$$\mathbf{measure}_M \, \psi \, r == \langle \psi | M_{r'}^\dagger M_{r'} \psi \rangle \times \left(\psi' = \frac{M_{r'} \psi}{\sqrt{\langle \psi | M_{r'}^\dagger M_{r'} \psi \rangle}} \right) \times (\sigma' = \sigma)$$

where $\sigma' = \sigma$ is an abbreviation of $(x' = x) \times (y' = y) \times \ldots$ and means "all other variables are unchanged".

Given an arbitrary orthonormal basis $B = b_{0,..2^n}$, measurement of ψ in basis B is:

$$\mathbf{measure}_B \, \psi \, r == |\langle b_{r'} | \psi \rangle|^2 \times (\psi' = b_{r'}) \times (\sigma' = \sigma)$$

The simplest and the most commonly used measurement in the computational basis is:

$$\textbf{measure}\ \psi\, r == |\psi r'|^2 \times (\psi' = |\mathbf{r}'\rangle) \times (\sigma' = \sigma)$$

In this case the distribution of r' is $|\psi r'|^2$ and the distribution of the quantum state is:

$$\sum r' \cdot |\psi r'|^2 \times (\psi' = |\mathbf{r}'\rangle)$$

which is precisely the mixed quantum state that results from the measurement.

In order to develop quantum programs we need to add to our list of implemented things. We add variables of type quantum state as above and we allow the following three kinds of operations on these variables. If ψ is a state of an n-qubit quantum system, r is a natural variable, and M is a collection of measurement operators that satisfy the completeness equation, then:

(i) $\psi := |0\rangle^{\otimes n}$ is a program

(ii) $\psi := U\psi$, where U is a unitary transformation on an n-qubit system, is a program

(iii) $\textbf{measure}_M\, \psi\, r$ is a program

where the superscript $^{\otimes n}$ means "tensored with itself n times". The special cases of measurements are therefore also allowed.

Some unitary operations that we will use in the later sections are (here $x, c : 0, 1$):

$I\|x\rangle = \|x\rangle$	identity
$X\|x\rangle = \|1 - x\rangle$	X - Pauli matrix
$Y\|x\rangle = (-1)^x \times i \times \|1 - x\rangle$	Y - Pauli matrix
$Z\|x\rangle = (-1)^x \times \|x\rangle$	Z - Pauli matrix
$H\|x\rangle = (\|0\rangle + (-1)^x \times \|1\rangle)/\sqrt{2}$	Hadamard
$CNOT\|cx\rangle = (I \otimes X^c)\|cx\rangle$	controlled-not

3 Distributed Quantum Systems and Communication

In predicative programming, to reason about distributed computation we (disjointly) partition the variables between the processes involved in a computation. Parallel composition is then simply boolean conjunction. For example, consider two processes P and Q. P owns integer variables x and y and Q owns an integer variable z. Suppose $P == x := x + 1\ ;\ y := x$ and $Q == z := -z$. Parallel composition of P with Q is then simply

$$P||Q == P \wedge Q == x' = x + 1 \wedge y' = x + 1 \wedge z' = -z$$

In quantum predicative programming, one needs to reason about distributed quantum systems. Recall that if ψ is a state of an m-qubit system and ϕ is a state of an n-qubit system, then $\psi \otimes \phi$, the tensor product of ψ and ϕ, is the state of a composite $m + n$-qubit system. On the other hand, given a composite $m + n$-qubit system, it is not always possible to describe it in terms of the tensor product of the component m- and n-qubit systems. Such a composed system is

entangled. Entanglement is one of the most non-classical, most poorly understood, and most interesting quantum phenomena. An entangled system is in some sense both distributed and shared. It is distributed in the sense that each party can apply operations and measurements to only its qubits. It is shared in the sense that the actions of one party affect the outcome of the actions of another party. Simple partitioning of qubits is therefore insufficient to reason about distributed quantum computation.

The formalism we introduce fully reflects the physical properties of a distributed quantum system. We start by partitioning the qubits between the parties involved. For example, consider two parties P and Q. P owns the first qubit of the composite entangled quantum system $\psi = |00\rangle/\sqrt{2} + |11\rangle/\sqrt{2}$ and Q owns the second qubit. A specification is a program only if each party computes with its own qubits. In our example,

$$P :: \psi_0 := H\psi_0 \,;\, \textbf{measure}\ \psi_0\ p \quad \text{and} \quad Q :: \textbf{measure}\ \psi_1\ q$$

are programs, if p and q are integer variables owned by P and Q, respectively.

Sometimes we want to explicitly include partitioning of variables as part of a specification. For this purpose, we introduce notation $\textbf{var}_P$ to mean the bunch of variables that belong to process P. In the above example we can make the partitioning of variables explicit with the specification

$$\psi_0, p : \textbf{var}_P \wedge \psi_1, q : \textbf{var}_Q$$

We define parallel composition of P and Q which share an $n{+}m$ quantum system in state ψ with the first n qubits belonging to P and the other m qubits belonging to Q as follows. If

$$P :: \psi_{0,..n} := U_P\psi_{0,..n} \quad \text{and} \quad Q :: \psi_{n,..n+m} := U_Q\psi_{n,..n+m}$$

where U_P is a unitary operation on an n-qubit system and U_Q is a unitary operation on an m-qubit system, then

$$P \,||_\psi\, Q :: \psi := (U_P \otimes U_Q)\psi$$

Performing *ok* is equivalent to performing the identity unitary operation, and therefore if

$$P :: \psi_{0,..n} := U_P\psi_{0,..n} \quad \text{and} \quad Q :: ok$$

then

$$P \,||_\psi\, Q :: \psi := (U_P \otimes I^{\otimes m})\psi$$

Similarly, if

$$P :: \textbf{measure}_{M_P}\ \psi_{0,..n}\ p \quad \text{and} \quad Q :: \textbf{measure}_{M_Q}\ \psi_{n,..n+m}\ q$$

where M_P and M_Q are a collection of proper measurement operators for n- and m-qubit systems, respectively, then

$$P \,||_\psi\, Q :: \textbf{measure}_{M_P \otimes M_Q}\psi\ p{\times}q$$

In our example,

$$
\begin{array}{lr}
\psi := |00\rangle/\sqrt{2} + |11\rangle/\sqrt{2}\ ;\ P\ ||_\psi Q & \text{expand, substitute} \\
== \psi := |00\rangle/\sqrt{2} + |11\rangle/\sqrt{2}\ ; & \\
\quad \textbf{measure}\ (H\psi_0)\ p\ ||_\psi\ \textbf{measure}\ \psi_1\ q & \text{compose on } \psi \\
== \psi := |00\rangle/\sqrt{2} + |11\rangle/\sqrt{2}\ ;\ \textbf{measure}\ (H \otimes I)\psi\ pq & \text{substitute} \\
== \textbf{measure}\ (H \otimes I)(|00\rangle/\sqrt{2} + |11\rangle/\sqrt{2})\ pq & \text{apply } H \otimes I \\
== \textbf{measure}\ (|00\rangle + |01\rangle + |10\rangle - |11\rangle)/2\ pq & \text{measure} \\
== |(|00\rangle + |01\rangle + |10\rangle - |11\rangle)/2\ pq|^2 \times (\psi' = |\mathbf{p'q'}\rangle) & \text{application} \\
== (\psi' = |\mathbf{p'q'}\rangle)/4 &
\end{array}
$$

When explicitly specifying partitioning of variables in a parallel composition, it is convenient to allow the variables to appear as subscripts on the corresponding processes. For example, the specification $P_{\psi_0,p}\ ||_\psi\ Q_{\psi_1,q}$ denotes a parallel composition of processes P and Q that share an entangled state ψ, such that ψ_0 and p belong to P and ψ_1 and q belong to Q.

To reason about communication between processes we use the framework of Hehner's calculus([14]). A named, one-way communication channel c is described by an infinite message script M_c, an infinite time script T_c, and read and write cursors r_c and w_c. The message and time scripts are the list of all messages that appear on the channel and the list of corresponding times. The read and write cursors specify how many messages have been read from and written to a channel. To specify two-way communication, we use two channels. The input and output on channel c are defined by the following operations (here t is the time variable):

$$
\begin{array}{lll}
c!e & == M_c w_c = e \wedge T_c w_c = t \wedge w_c := w_c + 1 & c \text{ output } e \\
c? & == r_c := r_c + 1 & c \text{ input} \\
c & == M_c(r_c - 1) &
\end{array}
$$

A channel declaration $\textbf{chan}\ c : T \cdot P$ defines a new channel c with communication of type T; the declaration applies to the specification P (here $xnat$ stands for naturals extended with ∞):

$$\textbf{chan}\ c : T \cdot P == \exists M_c : [\infty * T] \cdot \exists T_c : [\infty * real] \cdot \textbf{var}\ r_c, w_c : xnat := 0 \cdot P$$

where $[\infty * T]$ is an infinite sequence of elements of type T. One useful theorem that we use in later examples is the equivalence of communication on a local channel with assignment:

$$\textbf{chan}\ c : T \cdot c!e\ ||\ (c?\ ;\ x := c) == x := e$$

The reader is referred to [14] for a detailed description of formal treatment of classical communication in Hehner's calculus.

When defining a quantum communication channel one must be careful not to introduce any unwanted behaviour, such as violation of the no-cloning principle (i.e. creation of identical copies of an unknown arbitrary quantum state). For this

purpose we make the change of ownership of the transported qubit explicit in the definition:

$$\begin{aligned} c!\psi \quad &== M_c w_c = \psi \wedge T_c w_c = t \wedge w_c' = w_c + 1 \wedge \mathbf{var}_P' = \mathbf{var}_P \backslash \psi \wedge \sigma' = \sigma \\ c?\psi \quad &== r_c' = r_c + 1 \wedge \psi' = M_c r_c \wedge \mathbf{var}_Q' = \mathbf{var}_Q, \psi \wedge \sigma' = \sigma \end{aligned}$$

where c is a quantum communication channel from process P to process Q and $\sigma' = \sigma$ is shorthand for "the rest of the variables are unchanged".

Now that we allow changing of ownership of the variables, the specification $\sigma_P' = \sigma_P$, "the rest of the variables of process P are unchanged" is defined by $\forall v' : \mathbf{var}_P' \cdot v' = v$.

The declaration of a quantum channel **qchan** $c : qbit \cdot P$ is similar to the declaration of a local classical channel:

$$\mathbf{qchan}\ q : T \cdot P == \exists M_q : [\infty * T] \cdot \exists T_q : [\infty * real] \cdot \mathbf{var}\ r_q, w_q : xnat := 0 \cdot P$$

Similarly to the above-mentioned theorem, we can prove the equivalence of communication on a local quantum channel with the change of ownership. If $P == c!\psi$ and $Q == c?\psi$, then (leaving out time)

$$\begin{aligned} & \mathbf{qchan}\ c : qbit \cdot P \mid\mid Q && \text{def. qchan} \\ == {} & \exists M : [\infty * qbit] \cdot \mathbf{var}\ r, w : xnat := 0 \cdot P \mid\mid Q && \text{expand} \\ == {} & \exists M : [\infty * qbit] \cdot \mathbf{var}\ r, w : xnat := 0 \cdot \\ & \quad Mw = \psi \wedge w' = w + 1 \wedge \mathbf{var}_P' = \mathbf{var}_P \backslash \psi \\ & \quad \wedge \mathbf{var}_Q' = \mathbf{var}_Q, \psi \wedge r' = r + 1 \wedge \psi' = Mr && \text{initialisation} \\ == {} & \exists M : [\infty * qbit] \cdot \mathbf{var}\ r, w : xnat \cdot \\ & \quad M0 = \psi \wedge w' = 1 \wedge \mathbf{var}_P' = \mathbf{var}_P \backslash \psi \\ & \quad \wedge \mathbf{var}_Q' = \mathbf{var}_Q, \psi \wedge r' = 1 \wedge \psi' = M0 && \text{simplify} \\ == {} & \mathbf{var}_P' = \mathbf{var}_P \backslash \psi \wedge \mathbf{var}_Q' = \mathbf{var}_Q, \psi \wedge \sigma' = \sigma \end{aligned}$$

3.1 Quantum teleportation

Quantum teleportation is the most famous quantum communication protocol. Its description first appeared in a seminal article by Bennett *et al* in 1993 ([9]), it has since been extensively used as part of more complex quantum communication protocols, and has received much attention in experimental research. The protocol achieves transmission of quantum information by utilising only a classical communication channel and an entangled pair of qubits: no qubits are sent in the process.

The protocol: Alice and Bob share an entangled pair of qubits in the state $(|00\rangle + |11\rangle)/\sqrt{2}$. Alice has some qubit ψ in her possession (she may not know the state of the qubit) that she wishes to transfer to Bob. Alice starts by interacting the qubit she wishes to teleport with her half of the entangled pair (she applies a controlled-not followed by a Hadamard transform) and measuring her two qubits. She then sends the results of her measurements to Bob (two classical bits). Bob receives the two classical bits and, depending of their values, applies one of the three

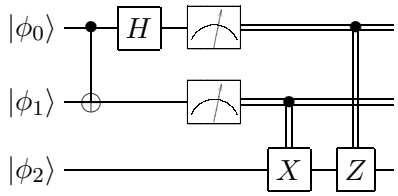

Fig. 1. Quantum teleportation protocol

Pauli operators or the identity to his qubit. Surprisingly, he has recovered the state Alice wished to teleport.

The protocol is usually described informally, by using a diagram as in Figure 1 [4]. Such a description is insufficient, in part since it only describes the evolution of the quantum system and does not specify the distribution of the system nor the communication. Alternatively, the description of the protocol is given informally, in English. Our goal is to formally define and prove correctness of the quantum teleportation protocol. Some approaches proposed in the literature (e.g. [27]) define teleportation as a program that implements a specification of the form $\phi' = \psi$. We point out that this specification may as well be implemented by a program that involves sending a qubit on a quantum channel, which is not teleportation. Furthermore, the specification does not mention that two classical bits are sent on a classical channel, which is an important part of the specification of teleportation. Similarly, it is important to specify that a pair of maximally entangled qubits is required.

To formalise the quantum teleportation protocol we let c be the number of classical bits sent on a communication channel and q be the number of quantum bits sent. The formal specification of quantum teleportation is:

$$\begin{aligned} S ==\quad & \phi_{01} : \mathbf{var}_{Alice} \wedge \phi_2 : \mathbf{var}_{Bob} \wedge \\ & \phi_{0,..3} = (\alpha \times |0\rangle + \beta \times |1\rangle) \otimes (|00\rangle + |11\rangle)/\sqrt{2} \\ \Rightarrow\ & \phi'_2 = \alpha \times |0\rangle + \beta \times |1\rangle \wedge c' = c + 2 \wedge q' = q \end{aligned}$$

The specification says that if the computation starts with a qubit (specified in the most general form) in Alice's possession and if Alice and Bob share a maximally entangled state $(|00\rangle + |11\rangle)/\sqrt{2}$, then at the end of the computation the qubit is teleported to Bob at a cost of 2 classical bits of communication and 0 qubits of communication. The specification does not restrict the quantum system to three qubits, so that teleportation can be a part of a bigger computation.

The well-known solution is the following parallel program:

$$\begin{aligned} P ==\ & \mathbf{chan}\ ch : bit \cdot Alice_{a_0,a_1,\phi_{01}} \ ||_\phi\ Bob_{b_0,b_1,\phi_2} \\ \text{where } Alice ==\ & \phi_{01} := CNOT\phi_{01}\ ;\ \phi_0 := H\phi_0\ ;\ \mathbf{measure}\ \phi_{01}\ a_0a_1\ ; \\ & c := c+1\ ;\ ch!a_0\ ;\ c := c+1\ ;\ ch!a_1 \\ \text{and } Bob ==\ & ch?\ ;\ b_0 := ch\ ;\ ch?\ ;\ b_1 := ch\ ;\ \phi_2 := Z^{b_0}X^{b_1}\phi_2 \end{aligned}$$

That is, two processes, *Alice* and *Bob*, partition a 3-qubit quantum system ϕ, so

[4] The figure is generated with *qasm2pdf*

that *Alice* owns the first qubit (the one she wants to teleport) and the second qubit and *Bob* owns the third qubit. *Alice* can write to a local classical communication channel *ch* and *Bob* can read from it. Finally, a_0 and a_1 are two bits that belong to *Alice*, and b_0 and b_1 are two bits that belong to *Bob*. The process *Alice* performs local operations and measurements and sends two classical bits on the channel. The process *Bob* reads from the channel and performs local operations.

Our goal is to prove that the program P implements the specification S. We first note the following equivalence:

$$
\begin{array}{lll}
 & P \Rightarrow S & \text{def. } S \\
= & P \wedge \phi_{01} : \mathbf{var}_{Alice} \wedge \phi_2 : \mathbf{var}_{Bob} \wedge & \\
 & \phi_{0,..3} = (\alpha \times |0\rangle + \beta \times |1\rangle) \otimes (|00\rangle + |11\rangle)/\sqrt{2} & \\
 & \Rightarrow \phi_2' = \alpha \times |0\rangle + \beta \times |1\rangle \wedge c' = c + 2 \wedge q' = q & \text{simplification} \\
= & P \wedge \phi = (\alpha \times |0\rangle + \beta \times |1\rangle) \otimes (|00\rangle + |11\rangle)/\sqrt{2} & \\
 & \Rightarrow \phi_2' = \alpha \times |0\rangle + \beta \times |1\rangle \wedge c' = c + 2 \wedge q' = q &
\end{array}
$$

Next, we simplify P to prove the above implication. With implicit partitioning of variables (as it does not change):

$$
\begin{array}{ll}
\textbf{chan } ch : bit \cdot & ((\phi_{01} := CNOT\phi_{01} \ ; \ \phi_0 := H\phi_0 \ ; \ \textbf{measure } \phi_{01} \ a_0a_1 \ ; \\
 & \quad c := c + 1 \ ; \ ch!a_0 \ ; \ c := c + 1 \ ; \ ch!a_1) \\
 & ||_\phi \ (ch? \ ; \ b_0 := ch \ ; \ ch? \ ; \ b_1 := ch \ ; \ \phi_2 := Z^{b_0}X^{b_1}\phi_2)) \\
= & \text{substitute, H on first qubit} \\
\textbf{chan } ch : bit \cdot & ((\textbf{measure } H \otimes I(CNOT\phi_{01}) \ a_0a_1 \ ; \\
 & \quad c := c + 1 \ ; \ ch!a_0 \ ; \ c := c + 1 \ ; \ ch!a_1) \\
 & ||_\phi \ (ch? \ ; \ b_0 := ch \ ; \ ch? \ ; \ b_1 := ch \ ; \ \phi_2 := Z^{b_0}X^{b_1}\phi_2)) \\
= & \text{parallel composition, simplification} \\
\textbf{chan } ch : bit \cdot & \\
\quad \textbf{measure}_{01} \ H \otimes I \otimes I(CNOT \otimes I\phi) \ a_0a_1 \ ; & \\
\quad ((c := c + 1 \ ; \ ch!a_0 \ ; \ c := c + 1 \ ; \ ch!a_1) \ || \ (ch? \ ; \ b_0 := ch \ ; \ ch? \ ; \ b_1 := ch)) \ ; & \\
\quad \phi := I \otimes I \otimes Z^{b_0}(I \otimes I \otimes X^{b_1}\phi) & \\
= & \text{classical channel} \\
\quad \textbf{measure}_{01} \ H \otimes I \otimes I(CNOT \otimes I\phi) \ a_0a_1 \ ; & \\
\quad c' = c + 2 \wedge b_0' = a_0 \wedge b_1' = a_1 \wedge \sigma' = \sigma \ ; & \\
\quad \phi := I \otimes I \otimes Z^{b_0}(I \otimes I \otimes X^{b_1}\phi) &
\end{array}
$$

Next we notice that the first line in the above specification (which is, in fact, the effect of Alice's actions) conjoined with the specification of the initial state of the quantum system, result in the following distribution over the states of the computation:

$$\phi' = |a_0'a_1'\rangle \otimes (\alpha \times |a_1'\rangle + (-1)^{a_0'} \times \beta \times |1 - a_1'\rangle)/4$$

That is, with probability 1/4 the quantum system is in state $|00\rangle \otimes (\alpha \times |0\rangle +$

$\beta \times |1\rangle)$ and the values of Alice's bits are $a_0 = 0$ and $a_1 = 0$; with probability 1/4 the quantum system is in state $|01\rangle \otimes (\alpha \times |1\rangle + \beta \times |0\rangle)$ and the values of Alice's bits are $a_0 = 0$ and $a_1 = 1$; etc.

To prove this formally, we first note that:

$$
\begin{array}{ll}
 & H \otimes I \otimes I(CNOT \otimes I((\alpha \times |0\rangle + \beta \times |1\rangle) \otimes (|00\rangle + |11\rangle)/\sqrt{2})) \\
== & \text{apply CNOT} \\
 & H \otimes I \otimes I(\alpha \times |000\rangle + \beta \times |110\rangle + \alpha \times |011\rangle + \beta \times |101\rangle)/\sqrt{2} \\
== & \text{apply H} \\
 & \alpha \times (|0\rangle + |1\rangle) \otimes |00\rangle/2 + \beta \times (|0\rangle - |1\rangle) \otimes |10\rangle/2 + \\
 & \alpha \times (|0\rangle + |1\rangle) \otimes |11\rangle/2 + \beta \times (|0\rangle - |1\rangle) \otimes |01\rangle/2 \\
== & \text{rearrange terms} \\
 & |00\rangle \otimes (\alpha \times |0\rangle + \beta \times |1\rangle)/2 + |01\rangle \otimes (\alpha \times |1\rangle + \beta \times |0\rangle)/2 + \\
 & |10\rangle \otimes (\alpha \times |0\rangle - \beta \times |1\rangle)/2 + |11\rangle \otimes (\alpha \times |1\rangle - \beta \times |0\rangle)/2
\end{array}
$$

Therefore, measurement of the first two qubits of the above state in the computational basis gives:

$$
\begin{array}{ll}
\textbf{measure}_{01} & (|00\rangle \otimes (\alpha \times |0\rangle + \beta \times |1\rangle)/2 + |01\rangle \otimes (\alpha \times |1\rangle + \beta \times |0\rangle)/2 + \\
 & \quad |10\rangle \otimes (\alpha \times |0\rangle - \beta \times |1\rangle)/2 + |11\rangle \otimes (\alpha \times |1\rangle - \beta \times |0\rangle)/2) \\
 & \quad a_0 a_1 \\
== & \phi' = |a'_0 a'_1\rangle \otimes (\alpha \times |a'_1\rangle + (-1)^{a'_0} \times \beta \times |1 - a'_1\rangle)/4
\end{array}
$$

Let Q be the specification of the initial state of the quantum system:

$$Q == \phi = (\alpha \times |0\rangle + \beta \times |1\rangle) \otimes (|00\rangle + |11\rangle)/\sqrt{2}$$

Putting it all together, we get:

$$
\begin{array}{lll}
 & Q \wedge P & \\
== & Q \times (\textbf{measure}_{01}\ H \otimes I \otimes I(CNOT \otimes I\phi)\ a_0 a_1\ ; & \\
 & c' = c + 2 \wedge b'_0 = a_0 \wedge b'_1 = a_1 \wedge \sigma' = \sigma\ ; & \text{one point law,} \\
 & \phi := I \otimes I \otimes Z^{b_0}(I \otimes I \otimes X^{b_1}\phi)) & \text{as above} \\
== & Q \times \phi' = |a'_0 a'_1\rangle \otimes (\alpha \times |a'_1\rangle + (-1)^{a'_0} \times \beta \times |1 - a'_1\rangle)/4\ ; & \text{sequential} \\
 & c' = c + 2 \wedge b'_0 = a_0 \wedge b'_1 = a_1 \wedge \sigma' = \sigma\ ; & \text{composition,} \\
 & \phi := I \otimes I \otimes Z^{b_0}(I \otimes I \otimes X^{b_1}\phi) & \text{one point law} \\
== & Q \times (c' = c + 2) \times (b'_0 = a'_0) \times (b'_1 = a'_1) \times (\sigma' = \sigma) \times & \\
 & \phi' = I \otimes I \otimes Z^{b'_0}(I \otimes I \otimes X^{b'_1} & \\
 & \quad (|b'_0 b'_1\rangle \otimes (\alpha \times |b'_1\rangle + (-1)^{b'_0} \times \beta \times |1 - b'_1\rangle)/4)) & \text{apply } X^{b'_1} \\
== & Q \times (c' = c + 2) \times (b'_0 = a'_0) \times (b'_1 = a'_1) \times (\sigma' = \sigma) \times & \\
 & \phi' = I \otimes I \otimes Z^{b'_0}(|b'_0 b'_1\rangle \otimes (\alpha \times |0\rangle + (-1)^{b'_0} \times \beta \times |1\rangle)/4 & \text{apply } Z^{b'_0} \\
== & Q \times (c' = c + 2) \times (b'_0 = a'_0) \times (b'_1 = a'_1) \times (\sigma' = \sigma) \times & \\
 & (\phi' = |b'_0 b'_1\rangle \otimes (\alpha \times |0\rangle + \beta \times |1\rangle)/4) &
\end{array}
$$

$$\begin{aligned} &\leq (c' = c + 2) \times (q' = q) \times (\phi_2' = \alpha \times |0\rangle + \beta \times |1\rangle) \\ &== S \end{aligned}$$

This example shows formalisation and analysis of an LOCC (local operations, classical communication) quantum communication protocol. We now turn to our attention to a protocol which involves a quantum communication channel.

3.2 Quantum dense coding

The quantum dense coding (sometimes called super-dense coding) protocol is less famous than the quantum teleportation protocol, but it is no less important. It achieves the transfer of 2 bits of classical information by sending 1 bit of quantum information and utilising 1 entangled pair of qubits. That is, its goal is the opposite of that of the quantum teleportation protocol.

Just as with teleportation, the protocol is usually described informally: either with a diagram or in English. We formalise the specification of the protocol by using the same variables as in section 3.1:

$$\begin{aligned} S == \quad & a_0, a_1, \phi_0 : \mathbf{var}_{Alice} \wedge b_0, b_1, \phi_1 : \mathbf{var}_{Bob} \wedge \phi_{01} = (|00\rangle + |11\rangle)/\sqrt{2} \\ & \Rightarrow b_0' = a_0 \wedge b_1' = a_1 \wedge c' = c \wedge q' = q + 1 \end{aligned}$$

The specification says that if the computation starts with Alice and Bob sharing a maximally entangled state, with classical bits a_0 and a_1 in Alice's possession and b_0 and b_1 in Bob's possession, then at the end of the computation Bob has the values of Alice's classical bits, at a cost of sending no bits on a classical channel and one qubit on a quantum channel. The program for the protocol is:

$$\begin{aligned} P == \ & \mathbf{qchan}\ qch : qbit \cdot Alice_{a_0,a_1,\phi_0} \ ||_\phi \ Bob_{b_0,b_1,\phi_1} \\ \text{where } Alice == \ & \mathbf{if}\ a_0 = a_1 = 0\ \mathbf{then}\ ok \\ & \mathbf{else\ if}\ a_0 = 0 \wedge a_1 = 1\ \mathbf{then}\ \phi_0 := X\phi_0 \\ & \mathbf{else\ if}\ a_0 = 1 \wedge a_1 = 0\ \mathbf{then}\ \phi_0 := Z\phi_0 \\ & \mathbf{else}\ \phi_0 := Y\phi_0\ ; \\ & q := q + 1\ ;\ qch!\phi_0 \\ \text{and } Bob == \ & qch?\phi_0\ ;\ \phi := CNOT\phi\ ;\ \phi_0 := H\phi_0\ ;\ \mathbf{measure}\ \phi\ b_0 b_1 \end{aligned}$$

That is, Alice applies one of the three Pauli operators or an identity to her half of the entangled pair, depending on the values of her classical bits, and sends her qubit to Bob. Bob receives the qubit, applies a controlled-not followed by a Hadamard, and measures the two qubits in his possession. We now show that the program P implements the specification S. First, we simplify the processes *Alice* and *Bob*:

$$\begin{aligned} Alice == \ & \phi_0 := (-i)^{a_0 \times a_1} \times Z^{a_0}(X^{a_1}\phi_0)\ ;\ q := q + 1\ ;\ qch!\phi_0 && \text{(math)} \\ Bob == \ & qch?\phi_0\ ;\ \mathbf{measure}\ H \otimes I(CNOT\phi)\ b_0 b_1 && \text{(substitutions)} \end{aligned}$$

We now look at their parallel composition:

$$P == \mathbf{qchan}\ qch : qbit \cdot Alice_{a_0,a_1,\phi_0} \ ||_\phi \ Bob_{b_0,b_1,\phi_1}$$

$$
\begin{aligned}
&== \textbf{qchan}\ qch : qbit\cdot \\
&\quad ((\phi_0 := (-i)^{a_0 \times a_1} \times Z^{a_0}(X^{a_1}\phi_0)\ ;\ q := q+1\ ;\ qch!\phi_0)_{a_0,a_1,\phi_0} \\
&\quad\ ||_\phi\ (qch?\phi_0; \textbf{measure}\ H \otimes I(CNOT\phi)\ b_0b_1)_{b_0,b_1,\phi_1}) \\
&\qquad \text{quantum channel} \\
&== \phi := (-i)^{a_0 \times a_1} \times Z^{a_0} \otimes I(X^{a_1} \otimes I\phi)\ ; \\
&\quad q' = q+1 \wedge \textbf{var}'_{Alice} = \textbf{var}_{Alice} \backslash \phi_0 \wedge \textbf{var}'_{Bob} = \textbf{var}_{Bob}, \phi_0 \wedge \sigma' = \sigma\ ; \\
&\quad \textbf{measure}\ H \otimes I(CNOT\ \phi)\ b_0b_1 \\
&\qquad \text{sequential composition} \\
&== (\textbf{measure}\ (-i)^{a_0 \times a_1} \times H \otimes I(CNOT\ (Z^{a_0} \otimes I(X^{a_1} \otimes I\phi)))\ b_0b_1) \times \\
&\quad (q' = q+1) \times (\textbf{var}'_{Alice} = a_0, a_1) \times (\textbf{var}'_{Bob} = b_0, b_1, \psi_0, \psi_1) \times (\sigma' = \sigma)
\end{aligned}
$$

Next, we note that the quantum state being measured is:

$$
\begin{aligned}
&(-i)^{a_0 \times a_1} \times H \otimes I(CNOT\ (Z^{a_0} \otimes I(X^{a_1} \otimes I(|00\rangle + |11\rangle)/\sqrt{2}))) \\
== &(a_0 = 0) \times (a_1 = 0) \times H \otimes I(CNOT\ (|00\rangle + |11\rangle)/\sqrt{2}) + \\
&(a_0 = 0) \times (a_1 = 1) \times H \otimes I(CNOT\ (X \otimes I(|00\rangle + |11\rangle)/\sqrt{2})) + \\
&(a_0 = 1) \times (a_1 = 0) \times H \otimes I(CNOT\ (Z \otimes I(|00\rangle + |11\rangle)/\sqrt{2})) + \\
&(a_0 = 1) \times (a_1 = 1) \times (-i) \times H \otimes I(CNOT\ (Z \otimes I(X \otimes I(|00\rangle + |11\rangle)/\sqrt{2}))) \\
&\qquad \text{apply X} \\
== &(a_0 = 0) \times (a_1 = 0) \times H \otimes I(CNOT\ (|00\rangle + |11\rangle)/\sqrt{2}) + \\
&(a_0 = 0) \times (a_1 = 1) \times H \otimes I(CNOT\ (|10\rangle + |01\rangle)/\sqrt{2}) + \\
&(a_0 = 1) \times (a_1 = 0) \times H \otimes I(CNOT\ (Z \otimes I(|00\rangle + |11\rangle)/\sqrt{2})) + \\
&(a_0 = 1) \times (a_1 = 1) \times (-i) \times H \otimes I(CNOT\ (Z \otimes I(|10\rangle + |01\rangle)/\sqrt{2})) \\
&\qquad \text{apply Z} \\
== &(a_0 = 0) \times (a_1 = 0) \times H \otimes I(CNOT\ (|00\rangle + |11\rangle)/\sqrt{2}) + \\
&(a_0 = 0) \times (a_1 = 1) \times H \otimes I(CNOT\ (|10\rangle + |01\rangle)/\sqrt{2}) + \\
&(a_0 = 1) \times (a_1 = 0) \times H \otimes I(CNOT\ (|00\rangle - |11\rangle)/\sqrt{2}) + \\
&(a_0 = 1) \times (a_1 = 1) \times (-i) \times H \otimes I(CNOT\ (-|10\rangle + |01\rangle)/\sqrt{2}) \\
&\qquad \text{apply CNOT} \\
== &(a_0 = 0) \times (a_1 = 0) \times H \otimes I(|00\rangle + |10\rangle)/\sqrt{2} + \\
&(a_0 = 0) \times (a_1 = 1) \times H \otimes I(|11\rangle + |01\rangle)/\sqrt{2} + \\
&(a_0 = 1) \times (a_1 = 0) \times H \otimes I(|00\rangle - |10\rangle)/\sqrt{2} + \\
&(a_0 = 1) \times (a_1 = 1) \times (-i) \times H \otimes I(-|11\rangle + |01\rangle)/\sqrt{2} \\
&\qquad \text{apply H} \\
== &(a_0 = 0) \times (a_1 = 0) \times |00\rangle + \\
&(a_0 = 0) \times (a_1 = 1) \times |01\rangle + \\
&(a_0 = 1) \times (a_1 = 0) \times |10\rangle) + \\
&(a_0 = 1) \times (a_1 = 1) \times (-i) \times |11\rangle \\
== &(-i)^{a_0 \times a_1} \times |a_0a_1\rangle
\end{aligned}
$$

Putting it all together, we get:

$$
\begin{array}{lll}
 & (\phi_{01} = (|00\rangle + |11\rangle)/\sqrt{2}) \wedge P & \text{as above} \\
== & (\textbf{measure}\ (-i)^{a_0 \times a_1} \times |a_0 a_1\rangle\ b_0 b_1) \times (q' = q + 1) \times & \\
 & (\textbf{var}'_{Alice} = a_0, a_1) \times (\textbf{var}'_{Bob} = b_0, b_1, \psi_0, \psi_1) \times (\sigma' = \sigma) & \text{measure} \\
== & (\phi' = |b'_0 b'_1\rangle) \times (b'_0 = a_0) \times (b'_1 = a_1) \times (q' = q + 1) \times & \\
 & (\textbf{var}'_{Alice} = a_0, a_1) \times (\textbf{var}'_{Bob} = b_0, b_1, \psi_0, \psi_1) \times (\sigma' = \sigma) & \\
\leq & S &
\end{array}
$$

This example shows formalisation and analysis of a quantum communication protocol which involves a quantum communication channel.

4 Conclusion and Future Work

We have presented a formal framework for specifying, implementing, and analysing quantum communication protocols. The analysis is not limited to reasoning about the data sent or received during the execution of the protocol. We provide tools to formally prove complexity of the communication protocols, such as the number of classical and quantum bits sent during the execution. We have applied our approach to two important quantum communication protocols: quantum teleportation and quantum dense coding. The resulting formal proofs are short: in fact, the proofs in Sections 3.1 and 3.2 are only slightly longer than the informal reasoning and calculations in [19]. The proofs are easy to read, the use of Dirac-like notation makes the expressions of quantum states look familiar, while providing a formal treatment that fits in the overall framework. Finally, the formal proofs are checkable by a computer (although we currently do not have suitable software implemented), thus providing a measure of confidence in the analysis of correctness and complexity of the protocols.

Current research focuses on formal reasoning about complexity of distributed quantum algorithms (e.g. [26]). Future work involves formalising quantum cryptographic protocols, such as BB84 [8], in our framework and providing formal analysis of these protocols.

References

[1] Abramsky, S., *High-level methods for quantum computation and information*, in: *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science* (2004), pp. 410–414.

[2] Abramsky, S. and B. Coecke, *A categorical semantics of quantum protocols*, in: *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science* (2004), pp. 415–425.

[3] Abramsky, S. and R. Duncan, *A categorical quantum logic*, Mathematical Structures in Computer Science **16** (2006), pp. 469–489.

[4] Adão, P. and P. Mateus, *A process algebra for reasoning about quantum security*, in: *Proceedings of the 3rd International Workshop on Quantum Programming Languages*, 2007, pp. 3–21.

[5] Altenkirch, T. and J. Grattage, *A functional quantum programming language*, in: *Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science* (2005), pp. 249–258.

[6] Arrighi, P. and G. Dowek, *Operational semantics for formal tensorial calculus*, in: *Proceedings of the 2nd International Workshop on Quantum Programming Languages*, 2004, pp. 21–38.

[7] Arrighi, P. and G. Dowek, *Linear-algebraic λ-calculus*, arXiv:quant-ph/0501150 (2005).

[8] Bennett, C. H. and G. Brassard, *Quantum cryptography: Public-key distribution and coin tossing*, in: *Proceedings of IEEE International Conference on Computers, Systems and Signal Processing*.

[9] Bennett, C. H., G. Brassard, C. Crépeau, R. Jozsa, A. Peres and W. K. Wootters, *Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels*, Phys. Rev. Lett. **70** (1993), pp. 1895–1899.

[10] Coecke, B., *The logic of entanglement*, arXiv:quant-ph/0402014 (2004).

[11] Danos, V., E. Kashefi and P. Panangaden, *The measurement calculus*, Journal of the ACM **54**.

[12] D'Hondt, E. and P. Panangaden, *Quantum weakest preconditions*, Mathematical Structures in Computer Science **16** (2006), pp. 429–451.

[13] Gay, S. J. and R. Nagarajan, *Communicating quantum processes*, in: *Proceedings of the 32nd ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages* (2005), pp. 145–157.

[14] Hehner, E. C., "a Practical Theory of Programming," Springer, New York, 1993, first edition, current edn. (2009) Available free at www.cs.utoronto.ca/~hehner/aPToP.

[15] Hehner, E. C., *Probabilistic predicative programming*, in: *Proceedings of the 7th International Conference on Mathematics of Program Construction*, Lecture Notes in Computer Science **3125** (2004), pp. 169–185.

[16] Hehner, E. C., *A probability perspective*, Formal Aspects of Computing (2009), to appear.

[17] Jorrand, P. and M. Lalire, *Toward a quantum process algebra*, in: *Proceedings of the 1st ACM Conference on Computing Frontiers* (2004), pp. 111–119.

[18] Lalire, M. and P. Jorrand, *A process algebraic approach to concurrent and distributed computation: operational semantics*, in: *Proceedings of the 2nd International Workshop on Quantum Programming Languages*, 2004, pp. 109–126.

[19] Nielsen, M. A. and I. L. Chuang, "Quantum Computation and Quantum Information," Cambridge University Press, 2000.

[20] Sanders, J. W. and P. Zuliani, *Quantum programming*, in: *Mathematics of Program Construction*, Lecture Notes in Computer Science **1837** (2000).

[21] Selinger, P., *Towards a quantum programming language*, Mathematical Structures in Computer Science **14** (2004), pp. 527–586.

[22] Tafliovich, A., "Quantum Programming," Master's thesis, University of Toronto (2004).

[23] Tafliovich, A. and E. C. Hehner, *Quantum predicative programming*, in: *Proceedings of the 8th International Conference on Mathematics of Program Construction*, Lecture Notes in Computer Science **4014** (2006), pp. 433–454.

[24] Tafliovich, A. and E. C. Hehner, *Programming telepathy: Implementing quantum non-locality games*, in: *Proceedings of the 10th Brazilian Symposium on Formal Methods* (2007), pp. 70–86.

[25] Valiron, B., *Quantum typing*, in: *Proceedings of the 2nd International Workshop on Quantum Programming Languages*, 2004, pp. 163–178.

[26] Yimsiriwattana, A. and S. J. L. Jr, *Distributed quantum computing: A distributed Shor algorithm*, arXiv:quant-ph/0403146 (2004).

[27] Zuliani, P., "Quantum Programming," DPhil thesis, University of Oxford (2001).

[28] Zuliani, P., *Non-deterministic quantum programming*, in: *Proceedings of the 2nd International Workshop on Quantum Programming Languages*, 2004, pp. 179–195.

A Quantum Computation

In this section we introduce the basic concepts of quantum mechanics, as they pertain to the quantum systems that we consider for quantum computation. The discussion of the underlying physical processes, spin-$\frac{1}{2}$-particles, etc. is not our interest. We are concerned with the model for quantum computation only. A reader not familiar with quantum computing can consult [19] for a comprehensive introduction to the field.

The *Dirac notation*, invented by Paul Dirac, is often used in quantum mechanics. In this notation a vector v (a column vector by convention) is written inside a *ket*: $|v\rangle$. The dual vector of $|v\rangle$ is $\langle v|$, written inside a *bra*. The inner products are *bra-kets* $\langle v|w\rangle$. For n-dimensional vectors $|u\rangle$ and $|v\rangle$ and m-dimensional vector $|w\rangle$, the value of the inner product $\langle u|v\rangle$ is a scalar and the outer product operator $|v\rangle\langle w|$ corresponds to an m by n matrix. The Dirac notation clearly distinguishes vectors from operators and scalars, and makes it possible to write operators directly as combinations of bras and kets.

In quantum mechanics, the vector spaces of interest are the Hilbert spaces of dimension 2^n for some $n \in \mathbb{N}$. A convenient orthonormal basis is what is called a *computational basis*, in which we label 2^n basis vectors using binary strings of length n as follows: if s is an n-bit string which corresponds to the number x_s, then $|s\rangle$ is a 2^n-bit (column) vector with 1 in position x_s and 0 everywhere else. The tensor product $|i\rangle \otimes |j\rangle$ can be written simply as $|ij\rangle$. An arbitrary vector in a Hilbert space can be written as a weighted sum of the computational basis vectors.

Postulate 1 (state space) Associated to any isolated physical system is a Hilbert space, known as the *state space* of the system. The system is completely described by its *state vector*, which is a unit vector in the system's state space.

Postulate 2 (evolution) The evolution of a closed quantum system is described by a *unitary transformation*.

Postulate 3 (measurement) Quantum measurements are described by a collection $\{M_m\}$ of *measurement operators*, which act on the state space of the system being measured. The index m refers to the possible measurement outcomes. If the state of the system immediately prior to the measurement is described by a vector $|\psi\rangle$, then the probability of obtaining result m is $\langle\psi|M_m^\dagger M_m|\psi\rangle$, in which case the state of the system immediately after the measurement is described by the vector $\frac{M_m|\psi\rangle}{\sqrt{\langle\psi|M_m^\dagger M_m|\psi\rangle}}$. The measurement operators satisfy the *completeness equation* $\sum m \cdot M_m^\dagger M_m == I$.

An important special class of measurements is *projective measurements*, which are equivalent to general measurements provided that we also have the ability to perform unitary transformations.

A projective measurement is described by an *observable* M, which is a Hermitian operator on the state space of the system being measured. This observable has a spectral decomposition $M = \sum m \cdot \lambda_m \times P_m$, where P_m is the projector onto the eigenspace of M with eigenvalue λ_m, which corresponds to the outcome of the measurement. The probability of measuring m is $\langle\psi|P_m|\psi\rangle$, in which case

immediately after the measurement the system is found in the state $\frac{P_m|\psi\rangle}{\sqrt{\langle\psi|P_m|\psi\rangle}}$.

Given an orthonormal basis $|v_m\rangle$, $0 \leq m < 2^n$, measurement with respect to this basis is the corresponding projective measurement given by the observable $M = \sum m \cdot \lambda_m \times P_m$, where the projectors are $P_m = |v_m\rangle\langle v_m|$.

Measurement with respect to the computational basis is the simplest and the most commonly used class of measurements. In terms of the basis $|m\rangle$, $0 \leq m < 2^n$, the projectors are $P_m = |m\rangle\langle m|$ and $\langle\psi|P_m|\psi\rangle = |\psi_m|^2$. The state of the system immediately after measuring m is $|m\rangle$.

For example, measuring a single qubit in the state $\alpha \times |0\rangle + \beta \times |1\rangle$ results in the outcome 0 with probability $|\alpha|^2$ and outcome 1 with probability $|\beta|^2$. The state of the system immediately after the measurement is $|0\rangle$ or $|1\rangle$, respectively.

Suppose the result of the measurement is ignored and we continue the computation. In this case the system is said to be in a *mixed state*. A mixed state is not the actual physical state of the system. Rather it describes our knowledge of the state the system is in. In the above example, the mixed state is expressed by the equation $|\psi\rangle = |\alpha|^2 \times \{|0\rangle\} + |\beta|^2 \times \{|1\rangle\}$. The equation is meant to say that $|\psi\rangle$ is $|0\rangle$ with probability $|\alpha|^2$ and it is $|1\rangle$ with probability $|\beta|^2$. An application of operation U to the mixed state results in another mixed state, $U(|\alpha|^2 \times \{|0\rangle\} + |\beta|^2 \times \{|1\rangle\}) = |\alpha|^2 \times \{U|0\rangle\} + |\beta|^2 \times \{U|1\rangle\}$.

Postulate 4 (composite systems) The state space of a composite physical system is the tensor product of the state spaces of the component systems. If we have systems numbered 0 up to and excluding n, and each system i, $0 \leq i < n$, is prepared in the state $|\psi_i\rangle$, then the joint state of the composite system is $|\psi_0\rangle \otimes |\psi_1\rangle \otimes \ldots \otimes |\psi_{n-1}\rangle$.

While we can always describe a composite system given descriptions of the component systems, the reverse is not true. Indeed, given a state vector that describes a composite system, it may not be possible to factor it to obtain the state vectors of the component systems. A well-known example is the state $|\psi\rangle = |00\rangle/\sqrt{2}+|11\rangle/\sqrt{2}$. Such a state is called an *entangled* state.

Just as it may not be possible to represent the state of a multi-qubit system as tensor product of its component systems, it may not be possible to represent an operation on a composite system as a tensor product of single-qubit operations on the component systems. Consider, for example, "controlled-NOT" (CNOT) operation on two qubits defined by

$$CNOT(|0\rangle \otimes |x\rangle) = |0\rangle \otimes |x\rangle$$
$$CNOT(|1\rangle \otimes |x\rangle) = |1\rangle \otimes |1 - x\rangle$$

where $x \in 0, 1$. It can be shown that there are no two single-qubit operations U_0 and U_1, such that $CNOT = U_0 \otimes U_1$.