Available online at www.sciencedirect.com

SCIENCE DIRECT®

Electronic Notes in Theoretical Computer Science 158 (2006) 19–39

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Reasoning About Imperative Quantum Programs

R. Chadha[1,2] P. Mateus[1,3] A. Sernadas[1,4]

CLC, Department of Mathematics
Instituto Superior Técnico
Lisbon, Portugal

Abstract

A logic for reasoning about states of basic quantum imperative programs is presented. The models of the logic are ensembles obtained by attaching probabilities to pairs of quantum states and classical states. The state logic is used to provide a sound Hoare-style calculus for quantum imperative programs. The calculus is illustrated by proving the correctness of the Deutsch algorithm.

Keywords: Quantum imperative programs, exogenous semantics, Hoare logic.

1 Introduction

Reasoning about quantum programs has gained prominence due to a big potential in applications such as information processing, security, distributed systems and randomized algorithms. This has attracted research in reasoning about quantum states [34,33,24,7] and quantum programs [20,31,2,11,4,32,5].

We present herein another tool to reason about quantum programs. This work is the first attempt to provide Hoare-style reasoning about quantum

[1] Supported by FCT and FEDER through POCI via QuantLog POCI/MAT/55796/2004 Project. Additional support for Rohit Chadha came from FCT and FEDER grant SFRH/BPD/26137/2005.

[2] Email: rchadha@math.ist.utl.pt

[3] Email: pmat@math.ist.utl.pt

[4] Email: acs@math.ist.utl.pt

doi:10.1016/j.entcs.2006.04.003

imperative programs using a quantitative state logic. Other work in this direction [5] uses the traditional qualitative logic for reasoning about quantum states [8]. A Hoare assertion [17] is a triple of the form $\{\delta_1\}\,P\,\{\delta_2\}$ meaning that if program P starts in a state satisfying δ_1 and P halts then P ends in a state satisfying δ_2. The Hoare logic herein is inspired by the dynamical logics and Hoare logics for probabilistic programs [30,13,22,26,9,6]. The quantitative quantum state logic we use allows us to reason about probabilities and amplitudes contrary to the qualitative quantum logic which is appropriate to reason about orthogonality of observations.

The Hoare logic proposed in this paper is for a basic quantum imperative programming language. The programming language uses both classical and quantum memory. The classical part of the memory is manipulated by the standard imperative programming construct: assignment. The quantum part of the memory is manipulated by a universal set of unitary transformations and measurements in the standard computational basis. For simplicity sake, we only assume a finite memory. We also have other standard imperative programming constructs: sequential composition, alternative composition and bounded iteration. Thus, termination is not an issue since we do not have divergent iterations. This restricted programming language is convenient enough to model several known quantum programs. Actually, like circuits, it is computationally complete for n-ary total functions for each $n \in \mathbb{N}$.

The states of the programming language are (discrete and finitely supported) sub-probability distributions over pairs of classical and quantum memory states, henceforth referred as *ensembles*. The probability distributions are present to model outcomes of measurements. The *sub*-probability distributions are useful to model alternative composition. Furthermore, if we augment our programming language with unbounded iteration, as we plan to do, subprobability measures are a natural solution to deal with partiality. This is also the case with probabilistic programs [21]. The semantics of the programming language is a endo-map in the space of ensembles.

The state logic is designed by taking the exogenous semantics approach [29,12,25] to enriching a given base logic – the models of the enriched logic are sets of models of the given base logic with additional structure. In our case, the pairs of classical and quantum memory states (henceforth called *pure states*) constitute the models of the base logic. The base logic itself is a variant of the quantum logic presented in [24,7]. The base logic has terms that are interpreted by real and complex numbers. Amongst these complex terms, there are terms that represent amplitudes of quantum memory states. The base logic is built from comparison formulas that compare real terms and classical propositional logic that reasons about classical memory using the usual propositional connectives. The formulas of the base logic are called pure state formulas.

The state logic, henceforth referred to as Ensemble Exogenous Quantum Propositional Logic (EEQPL), contains terms to reason about expectations of the real terms of the base logic. We do not have expectations of the complex terms, but we can reason about the real and imaginary parts of complex terms. The terms (called *expected terms*) in EEQPL are built from these terms representing expectations using addition and multiplications.

There are two atomic EEQPL formulas (also called *expected formulas*): $\odot\gamma$ meaning that the pure state formula γ is true with probability 1, and $p_1 \leq p_2$ meaning that the expected term p_1 is less than the expected term p_2. EEQPL formulas are then built from these atoms using the propositional connectives ff and $\supset$, and a *conditional construct* ξ/γ. The formula ξ/γ is satisfied in a model of EEQPL if ξ is true of the model obtained by eliminating the measure of the pure states which do not satisfy γ, and the formula ξ/γ is used to deal with the alternative composition. We also present a sound axiomatization for EEQPL, and discuss completeness briefly. This logic is adapted from the state logic for probabilistic programs in [6].

We give a sound Hoare logic using the EEQPL formulas. The main contributions of the Hoare logic are the axioms for the unitary transformations and measurement operations which are presented in the weakest pre-condition form. The axiom for the measurement operator and the rule for the alternative composition is adapted from probabilistic toss axiom and alternative composition rule in [6] respectively. We envisage to achieve a complete Hoare logic by further extending the state logic, but this is out of the scope of this paper.

The rest of the paper is organized as follows. The syntax and semantics of the programming language is introduced in Section 2. The syntax, semantics and a sound axiomatization of the state logic EEQPL is presented in Section 3. The sound Hoare logic is given in Section 4 and illustrated with an example (Deutsch algorithm) in 5. We summarize the results and future work in Section 6.

2 Imperative quantum programs

We introduce a basic imperative quantum language with a hybrid memory. In the memory we have boolean and natural registers and qubits. We also generalize qubits to quantum natural registers (qunits). Please recall that a qubit is a unit vector in the two-dimensional Hilbert space. A qunit is a unit vector in a Nth-dimensional Hilbert space where N is a power of 2. The classical part of the memory is manipulated through classical imperative constructs. The quantum part of the memory is manipulated through unitary transformation and measurement operators. The measurement is always carried in a fixed (computational) basis and the result is stored in a classical register.

Before we present the full language we introduce the signature. We assume a fixed size memory, and towards this end we define the set $M = \{0, \ldots, M - 1\}$. We have four kinds of registers:

- boolean registers $\{\mathbf{b}_k : k \in M\}$ that take value in $2 = \{0, 1\}$;
- natural registers $\{\mathbf{n}_k : k \in M\}$ that take value in $N = \{0, \ldots, N - 1\}$;
- quantum boolean registers (qubits) $\{\mathbf{qb}_k : k \in M\}$ that take value in the unit sphere of $\mathcal{H}(2) = \mathbb{C}^2$; and
- quantum natural registers (qunits) $\{\mathbf{qn}_k : k \in M\}$ that take value in the unit sphere of $\mathcal{H}(N) = \mathbb{C}^N$.

We proceed to introduce the syntax and the semantics of the programming language.

2.1 Syntax

We have term expressions that are interpreted over the set N. These terms, henceforth called *natural expressions*, are defined in the BNF as follows:

$$\nu := \mathbf{n} \;[\!]\; \mathsf{c} \;[\!]\; (\nu + \nu) \;[\!]\; (\nu\, \nu)$$

where $\mathbf{n}$ is a natural register and $\mathsf{c} \in N$. We also have *boolean expressions*:

$$\beta := \mathbf{b} \;[\!]\; (\nu \leq \nu) \;[\!]\; \mathsf{f} \;[\!]\; \mathsf{t} \;[\!]\; (\beta \sqsupset \beta)$$

where $\mathbf{b}$ is a boolean register and f and t correspond to the value 0 and 1 respectively.

For unitary transformations, we only use a universal set. Any unitary transformation can be approximated as closely as desired using this set of transformations. The *transformation expressions* are:

$$U := \mathsf{I} \;[\!]\; \mathsf{X} : \mathbf{qb} \;[\!]\; \mathsf{X} : \mathbf{qn}(\nu, \nu) \;[\!]\; \mathsf{H} : \mathbf{qb} \;[\!]\; \mathsf{H} : \mathbf{qn} \;[\!]\; \mathsf{S} : \mathbf{qb}(\nu, \beta) \;[\!]\; \mathsf{S} : \mathbf{qn}(\nu, \nu) \;[\!]$$
$$(U\, U) \;[\!]\; (\mathsf{qif}\ \mathbf{qb}\ \mathsf{then}\ U\ \mathsf{else}\ U) \;[\!]\; (\mathsf{qcase}\ \mathbf{qn} \triangleright 0 : U, \ldots, N - 1 : U)$$

where I is the identity, $\mathsf{X} : \mathbf{qb}$ is the Pauli X operator acting on qubit $\mathbf{qb}$, $\mathsf{H} : \mathbf{qb}$ is the Hadamard operator acting on $\mathbf{qb}$ and $\mathsf{S} : \mathsf{qb}(\nu, \beta)$ is the shift operator acting on $\mathbf{qb}$ with phase shift parametrized by ν and β as will be explained in Section 2.2. The $(\mathsf{qif}\ \mathbf{qb}\ \mathsf{then}\ U_1\ \mathsf{else}\ U_2)$ is $\mathbf{qb}$-controlled unitary transformation, which is the usual generalization of a controlled-NOT transformation. The other unitary transformations are extensions to qunits. In quantum alternate and quantum case expressions the quantum register guard should not be in the target of the body expressions. The target of a transformation expression is a set of qubits and qunits defined as follows:

- $\mathsf{target}(\mathsf{I}) = \emptyset$;
- $\mathsf{target}(\mathsf{X} : \mathbf{qb}) = \{\mathbf{qb}\}$;
- $\mathsf{target}(\mathsf{X} : \mathbf{qn}(\nu_1, \nu_2)) = \{\mathbf{qn}\}$;
- $\mathsf{target}(\mathsf{H} : \mathbf{qb}) = \{\mathbf{qb}\}$;
- $\mathsf{target}(\mathsf{H} : \mathbf{qn}) = \{\mathbf{qn}\}$;
- $\mathsf{target}(\mathsf{S} : \mathbf{qb}(\nu, \beta)) = \{\mathbf{qb}\}$;
- $\mathsf{target}(\mathsf{S} : \mathbf{qn}(\nu_1, \nu_2)) = \{\mathbf{qn}\}$;
- $\mathsf{target}(U_2 U_1) = \mathsf{target}(U_1) \cup \mathsf{target}(U_2)$;
- $\mathsf{target}((\mathsf{qif}\ \mathbf{qb}\ \mathsf{then}\ U_1\ \mathsf{else}\ U_0)) = \mathsf{target}(U_0) \cup \mathsf{target}(U_1)$ where it is assumed that $\mathbf{qb} \notin \mathsf{target}(U_0) \cup \mathsf{target}(U_1)$;
- $\mathsf{target}((\mathsf{qcase}\ \mathbf{qb}\ \triangleright\ 0 : U_0, \ldots, N-1 : U_{N-1})) = \mathsf{target}(U_0) \cup \ldots \cup \mathsf{target}(U_{N-1})$ where it is assumed that $\mathbf{qb} \notin \mathsf{target}(U_0) \cup \ldots \cup \mathsf{target}(U_{N-1})$.

Finally, the statements of the programming language are:

$$\begin{aligned} s := {} & \mathsf{skip} \mathbin{[\!]} \mathbf{b} \leftarrow \beta \mathbin{[\!]} \mathbf{n} \leftarrow \nu \mathbin{[\!]} U \mathbin{[\!]} \mathbf{b} \leftleftarrows \mathbf{qb} \mathbin{[\!]} \mathbf{n} \leftleftarrows \mathbf{qn} \mathbin{[\!]} (s; \ldots; s) \mathbin{[\!]} \\ & (\mathsf{if}\ \mathbf{b}\ \mathsf{then}\ s\ \mathsf{else}\ s) \mathbin{[\!]} (\mathsf{case}\ \mathbf{n}\ \triangleright\ 0 : s, \ldots, N-1 : s) \mathbin{[\!]} (\mathbf{n}\ \mathsf{repeat}\ s) \end{aligned}$$

where $\mathbf{b} \leftarrow \beta$ and $\mathbf{n} \leftarrow \gamma$ are assignments to classical registers, U is a unitary transformation expression, $\mathbf{b} \leftleftarrows \mathbf{qb}$ and $\mathbf{n} \leftleftarrows \mathbf{qn}$ are measurements of the quantum registers in the computational basis whose value is stored in the indicated classical registers, and the remaining statements are the usual constructs of imperative languages. In alternative and iterative programs the guard must not be the *target of a change* in the body, *i.e*, should not be modified in the body. The target of a change can be defined straightforwardly. This is done to ensure that the iterative programs always terminate and to be consistent with the quantum alternative unitary transformations.

2.2 Semantics

The programs are interpreted as maps between *ensembles*. Intuitively, an ensemble is a (sub-)probability measure of *pure states*. A pure state is a pair one part of which assigns values to the classical memory and the other one assigns a quantum state to the quantum memory. We now formalize these definitions.

A *classical valuation* v is a map that provides values to the classical memory registers. That is, $v \in 2^M \times N^M$. A *quantum valuation* $|\psi\rangle$ is a unit vector of $\mathcal{H} = \mathcal{H}(2^M \times N^M) = \mathbb{C}^{(2^M \times N^M)}$. A *pure state* is a pair $\sigma = (v, |\psi\rangle)$ where v is a classical valuation and $|\psi\rangle$ is a quantum valuation. Let Σ be the set of all pure states. An *ensemble* ρ is a discrete sub-probability measure on $\wp\Sigma$ with finite support. An ensemble is said to be *normal* if $\rho(\Sigma) = 1$.

We first define the semantics for the natural and boolean expressions. Given a classical valuation v, the denotation is inductively defined as follows:

- $[\![\mathbf{n}]\!]_v = v(\mathbf{n})$;
- $[\![\mathsf{c}]\!]_v = \mathsf{c} \bmod N$;
- $[\![\nu_1 + \nu_2]\!]_v = ([\![\nu_1]\!]_v + [\![\nu_2]\!]_v) \bmod N$;
- $[\![\nu_1\, \nu_2]\!]_v = ([\![\nu_1]\!]_v\, [\![\nu_2]\!]_v) \bmod N$;
- $[\![\mathbf{b}]\!]_v = v(\mathbf{b})$;
- $[\![\nu_1 \leq \nu_2]\!]_v = ([\![\nu_1]\!]_v \leq [\![\nu_2]\!]_v)$;
- $[\![\mathsf{f}]\!]_v = 0$;
- $[\![\mathsf{t}]\!]_v = 1$;
- $[\![\beta_1 \sqsupset \beta_2]\!]_v = ([\![\beta_1]\!]_v \leq [\![\beta_2]\!]_v)$.

The transformation expressions are interpreted as unitary operators on $\mathcal{H} = \mathcal{H}(2^M \times N^M)$. Given $\mathbf{qb}$, $\mathcal{H}$ can be viewed as a tensor product $\mathcal{H} = \mathcal{H}_{\mathbf{qb}} \otimes \mathcal{H}_{]\mathbf{qb}[}$. The space $\mathcal{H}_{\mathbf{qb}}$ is the tensor factor of $\mathcal{H}$ corresponding to $\mathbf{qb}$ and $\mathcal{H}_{]\mathbf{qb}[}$ is the tensor factor of $\mathcal{H}$ corresponding to the rest of the quantum registers. Similarly, we can define $\mathcal{H}_{\mathbf{qn}}$ and $\mathcal{H}_{]\mathbf{qn}[}$. A unitary transformation U acting on the tensor factor $\mathcal{H}'$ of $\mathcal{H}$ will usually be denoted by $U_{\mathcal{H}'}$.

The transformation $\mathsf{X} : \mathbf{qb}$ is interpreted as a tensor product of $\mathsf{X}_{\mathcal{H}_{\mathbf{qb}}}$ and $\mathsf{I}_{\mathcal{H}_{]\mathbf{qb}[}}$ where X is the Pauli X operator and I is the identity. Recall that the Pauli X operator acting on a two dimensional Hilbert space exchanges the amplitudes of the input vector. The generalized Pauli X operator $\mathsf{X}_H^{n_1 n_2}$ acts on $\mathcal{H}(N)$ and exchanges the amplitudes of $|n_1\rangle$ and $|n_2\rangle$ of the input vector.

The transformation $\mathsf{H} : \mathbf{qb}$ is interpreted as a tensor product of $\mathsf{H}_{\mathcal{H}_{\mathbf{qb}}}$ and $\mathsf{I}_{\mathcal{H}_{]\mathbf{qb}[}}$ where H is Hadamard operator. Similarly, the transformation $\mathsf{H} : \mathbf{qn}$ is interpreted as a tensor product of $\mathsf{H}_{\mathcal{H}_{\mathbf{qn}}}$ and $\mathsf{I}_{\mathcal{H}_{]\mathbf{qn}[}}$ where H is the Walsh-Hadamard operator. Please note that the Walsh-Hadamard operator is defined on $\mathcal{H}(N)$ only if N is a power of 2.

The phase shift transformation $\mathsf{S}_{\mathcal{H}}^{\theta d}(2)$ rotates the amplitude of the basis element $|d\rangle$ by $e^{2\pi/\theta}$. The transformation $\mathsf{S} : \mathbf{qb}(\nu, \beta)$ is interpreted as a tensor product of $\mathsf{S}_{\mathcal{H}_{\mathbf{qb}}}^{\theta d}$ and $\mathsf{I}_{\mathcal{H}_{]\mathbf{qb}[}}$ where θ is the interpretation of ν and d is the interpretation of β. Its generalization to a qunit is obvious. Furthermore, as ρ is a discrete measure, we will often confuse $\rho(\{\sigma\})$ by $\rho(\sigma)$ as ρ is uniquely determined by its point mass.

Finally, recall the notion of quantum alternate [28]. Given the unitary transformations U_0 and U_1 acting on $\mathcal{H}_{]\mathbf{qb}[}$, we denote by $\mathsf{qbcontrolled}(\mathbf{qb}, U_1, U_0)$ the unitary transformation acting on $\mathcal{H}$ such that:

- $\mathsf{qbcontrolled}(\mathbf{qb}, U_1, U_0)(|0\rangle \otimes |\psi\rangle) = |0\rangle \otimes U_0|\psi\rangle$;
- $\mathsf{qbcontrolled}(\mathbf{qb}, U_1, U_0)(|1\rangle \otimes |\psi\rangle) = |1\rangle \otimes U_1|\psi\rangle$.

Similarly, we introduce $\mathsf{qncontrolled}(\mathbf{qn}, U_0, \dots, U_{N-1})$:

- $\mathsf{qncontrolled}(\mathbf{qn}, U_0, \dots, U_{N-1})(|0\rangle \otimes |\psi\rangle) = |0\rangle \otimes U_0|\psi\rangle$;
- $\dots$
- $\mathsf{qncontrolled}(\mathbf{qn}, U_0, \dots, U_{N-1})(|N-1\rangle \otimes |\psi\rangle) = |N-1\rangle \otimes U_{N-1}|\psi\rangle$.

Formally, given a classical valuation v, the denotation of transformation expressions is as follows:

- $[\![\mathsf{I}]\!]_v = \mathsf{I}_{\mathcal{H}}$;
- $[\![\mathsf{X} : \mathbf{qb}]\!]_v = \mathsf{X}_{\mathcal{H}_{[\![\mathbf{qb}]\!]_v}} \otimes \mathsf{I}_{\mathcal{H}_{][\![\mathbf{qb}]\!]_v[}}$;
- $[\![\mathsf{X} : \mathbf{qn}(\nu_1, \nu_2)]\!]_v = \mathsf{X}^{[\![\nu_1]\!]_v [\![\nu_2]\!]_v}_{\mathcal{H}_{[\![\mathbf{qn}]\!]_v}} \otimes \mathsf{I}_{\mathcal{H}_{][\![\mathbf{qn}]\!]_v[}}$;
- $[\![\mathsf{H} : \mathbf{qb}]\!]_v = \mathsf{H}_{\mathcal{H}_{[\![\mathbf{qb}]\!]_v}} \otimes \mathsf{I}_{\mathcal{H}_{][\![\mathbf{qb}]\!]_v[}}$;
- $[\![\mathsf{H} : \mathbf{qn}]\!]_v = \mathsf{H}_{\mathcal{H}_{[\![\mathbf{qn}]\!]_v}} \otimes \mathsf{I}_{\mathcal{H}_{][\![\mathbf{qn}]\!]_v[}}$;
- $[\![\mathsf{S} : \mathbf{qb}(\nu, \beta)]\!]_v = \mathsf{S}^{[\![\nu]\!]_v [\![\beta]\!]_v}_{\mathcal{H}_{[\![\mathbf{qb}]\!]_v}} \otimes \mathsf{I}_{\mathcal{H}_{][\![\mathbf{qb}]\!]_v[}}$;
- $[\![\mathsf{S} : \mathbf{qn}(\nu_1, \nu_2)]\!]_v = \mathsf{S}^{[\![\nu_1]\!]_v [\![\nu_2]\!]_v}_{\mathcal{H}_{[\![\mathbf{qn}]\!]_v}} \otimes \mathsf{I}_{\mathcal{H}_{][\![\mathbf{qn}]\!]_v[}}$;
- $[\![U_2\, U_1]\!]_v = [\![U_2]\!]_v\, [\![U_1]\!]_v$;
- $[\![(\mathsf{qif}\ \mathbf{qb}\ \mathsf{then}\ U_1\ \mathsf{else}\ U_2)]\!]_v =$
  $\mathsf{qbcontrolled}(\mathbf{qb}, ([\![U_1]\!]_v)_{\mathcal{H}_{]\mathbf{qb}[}}, ([\![U_2]\!]_v)_{\mathcal{H}_{]\mathbf{qb}[}})$;
- $[\![(\mathsf{qcase}\ \mathbf{qn} \vartriangleright 0 : U_0, \dots, N-1 : U_{N-1})]\!]_v =$
  $\mathsf{qncontrolled}(\mathbf{qn}, ([\![U_0]\!]_v)_{\mathcal{H}_{]\mathbf{qn}[}}, \dots, ([\![U_{N-1}]\!]_v)_{\mathcal{H}_{]\mathbf{qn}[}})$.

Observe that for any transformation expression U, $[\![U]\!]_v = \mathsf{I}_{\mathcal{H}_{]\mathsf{target}(U)[}} \otimes U'$ where U' acts on $\mathcal{H}_{\mathsf{target}(U)}$.

Before providing the semantics of the program statements, we need some auxiliary notation. Given a classical valuation v, we denote by $v^{\mathbf{b}_k}_d$ the valuation that coincides with v except for $\mathbf{b}_k$ where it returns d. Similarly, we introduce $v^{\mathbf{n}_k}_d$. Furthermore, we need the following endo-maps in Σ:

- $\eta^{\mathbf{b}_k \leftarrow \beta}(v, |\psi\rangle) = (v^{\mathbf{b}_k}_{[\![\beta]\!]_v}, |\psi\rangle)$;
- $\eta^{\mathbf{n}_k \leftarrow \nu}(v, |\psi\rangle) = (v^{\mathbf{n}_k}_{[\![\nu]\!]_v}, |\psi\rangle)$;
- $\eta^{U}(v, |\psi\rangle) = (v, [\![U]\!]_v|\psi\rangle)$.

For $d \in 2$, we denote by $P^{\mathbf{qb}_k}_{|d\rangle}$ the projector from $\mathcal{H}$ to its subspace defined by the tensor product of the space generated by $|d\rangle$ with $\mathcal{H}_{]\mathbf{qb}_k[}$. Similarly, we introduce $P^{\mathbf{qn}_k}_{|d\rangle}$ for $d \in N$. With these projectors we are able to define the following maps that given a pure state return an ensemble:

- $$(\delta^{\mathbf{b}_{k_1} \Leftarrow \mathbf{qb}_{k_2}}(v, |\psi\rangle))\,(v', |\psi'\rangle) = \begin{cases} \|P^{\mathbf{qb}_{k_2}}_{|0\rangle}|\psi\rangle\| \text{ if } \left[\begin{array}{l} v' = v_0^{\mathbf{b}_{k_1}} \\ |\psi'\rangle = \dfrac{P^{\mathbf{qb}_{k_2}}_{|0\rangle}|\psi\rangle}{\|P^{\mathbf{qb}_{k_2}}_{|0\rangle}|\psi\rangle\|} \end{array}\right. \\ \|P^{\mathbf{qb}_{k_2}}_{|1\rangle}|\psi\rangle\| \text{ if } \left[\begin{array}{l} v' = v_1^{\mathbf{b}_{k_1}} \\ |\psi'\rangle = \dfrac{P^{\mathbf{qb}_{k_2}}_{|1\rangle}|\psi\rangle}{\|P^{\mathbf{qb}_{k_2}}_{|1\rangle}|\psi\rangle\|} \end{array}\right. \\ 0 \qquad\qquad \text{otherwise} \end{cases}$$
- $$(\delta^{\mathbf{n}_{k_1} \Leftarrow \mathbf{qn}_{k_2}}(v, |\psi\rangle))\,(v', |\psi'\rangle) = \begin{cases} \|P^{\mathbf{qb}_{k_2}}_{|0\rangle}|\psi\rangle\| \text{ if } \left[\begin{array}{l} v' = v_0^{\mathbf{b}_{k_1}} \\ |\psi'\rangle = \dfrac{P^{\mathbf{qb}_{k_2}}_{|0\rangle}|\psi\rangle}{\|P^{\mathbf{qb}_{k_2}}_{|0\rangle}|\psi\rangle\|} \end{array}\right. \\ \dots \\ \|P^{\mathbf{qb}_{k_2}}_{|N-1\rangle}|\psi\rangle\| \text{ if } \left[\begin{array}{l} v' = v_{N-1}^{\mathbf{b}_{k_1}} \\ |\psi'\rangle = \dfrac{P^{\mathbf{qb}_{k_2}}_{|N-1\rangle}|\psi\rangle}{\|P^{\mathbf{qb}_{k_2}}_{|N-1\rangle}|\psi\rangle\|} \end{array}\right. \\ 0 \qquad\qquad \text{otherwise} \end{cases}$$

Finally, we introduce some notation to denote restrictions to ensembles:

- for $d \in 2$, $\rho|_{\mathbf{b}:d}$ is the sub-measure of ρ defined as follows: $\rho|_{\mathbf{b}:d}(v, |\psi\rangle) = \rho(v, |\psi\rangle)$ if $v(\mathbf{b}) = d$; and $\rho|_{\mathbf{b}:d}(v, |\psi\rangle) = 0$ otherwise;
- similarly, for $d \in N$, $\rho|_{\mathbf{n}:d}$ is as follows: $\rho|_{\mathbf{n}:d}(v, |\psi\rangle) = \rho(v, |\psi\rangle)$ if $v(\mathbf{n}) = d$; and $\rho|_{\mathbf{n}:d}(v, |\psi\rangle) = 0$ otherwise.

The denotation of a statement s is an endo-map $[\![s]\!]$ on the set of ensembles, inductively defined as follows:

- $[\![\mathsf{skip}]\!](\rho) = \rho$;
- $[\![\mathbf{b}_k \leftarrow \beta]\!](\rho) = \rho \circ (\eta^{\mathbf{b}_k \leftarrow \beta})^{-1}$;
- $[\![\mathbf{n}_k \leftarrow \nu]\!](\rho) = \rho \circ (\eta^{\mathbf{n}_k \leftarrow \nu})^{-1}$;
- $[\![U]\!](\rho) = \rho \circ (\eta^{U})^{-1}$;
- $[\![\mathbf{b} \Leftarrow \mathbf{qb}]\!](\rho) = \sum_{\sigma \in \Sigma} \rho(\sigma)\, \delta^{\mathbf{b}_{k_1} \Leftarrow \mathbf{qb}_{k_2}}(\sigma)$;
- $[\![\mathbf{n} \Leftarrow \mathbf{qn}]\!](\rho) = \sum_{\sigma \in \Sigma} \rho(\sigma)\, \delta^{\mathbf{n}_{k_1} \Leftarrow \mathbf{qn}_{k_2}}(\sigma)$;

- $[\![s_1; \ldots; s_m]\!](\rho) = [\![s_m]\!](\ldots([\![s_1]\!](\rho))\ldots)$;
- $[\![(\mathsf{if}\ \mathbf{b}\ \mathsf{then}\ s_1\ \mathsf{else}\ s_0)]\!](\rho) = [\![s_1]\!](\rho|_{\mathbf{b}:1}) + [\![s_0]\!](\rho|_{\mathbf{b}:0})$;
- $[\![(\mathsf{case}\ \mathbf{n}\ \triangleright\ 0 : s_0, \ldots, N-1 : s_{N-1})]\!](\rho) = \sum_{d=0}^{N-1} [\![s_d]\!](\rho|_{\mathbf{n}:d})$;
- $[\![(\mathbf{n}\ \mathsf{repeat}\ s)]\!](\rho) = \sum_{d=0}^{N-1} [\![s]\!]^d(\rho|_{\mathbf{n}:d})$.

3 Logic of ensembles

We now introduce the logic to reason about ensembles, the states of our programming language. The logic is built by enriching the quantum logic [24] adopting the exogenous semantics approach. The models the quantum logic in [24] are quantum valuations. First, we enrich the quantum logic to reason about classical registers. Therefore, we have a logic to reason about pure states. Then, the semantic models of our logic are obtained exogenously by taking (sub-)probability measures over pure states. We shall call our logic Ensemble Exogenous Quantum Propositional Logic (EEQPL).

3.1 Syntax

The syntax of the language has two levels, one to reason about pure states and the other to reason about ensembles.

We first give the syntax of the logic to reason about pure states. The syntax uses two kind of terms, real and complex terms, which take values in real and complex numbers. The complex terms will also include terms representing amplitudes of the quantum valuation. Towards this we start by defining *valuation term*, which specifies a basis element of the computational basis of $\mathcal{H} = \mathcal{H}(2^M \times N^M)$. A valuation is a list of *valuation constraints* which specify the values of all registers in the basis elements. The syntax of valuation constraints in BNF form is:

$$\omega := \mathbf{qb} : 2 \mathrel{[\!]} \mathbf{qn} : N \mathrel{[\!]} \omega, \ldots, \omega.$$

A valuation term is a valuation constraint specifying the values of all quantum registers (M qubits and M qunits). The amplitude of the basis element ω is represented by the *amplitude term* $\langle\omega|\mathsf{t}\rangle$.

The *real terms* are ranged over by $\xi_1, \ldots \xi_n, \ldots$ while the *complex terms* are ranged over by $\zeta_1, \ldots \zeta_n, \ldots$. The syntax of real and complex terms is given by the following BNF :

$$\xi := \mathsf{r} \mathrel{[\!]} \nu \mathrel{[\!]} (\xi + \xi) \mathrel{[\!]} (\xi\,\xi) \mathrel{[\!]} \mathsf{Re}\zeta \mathrel{[\!]} \mathsf{Im}\zeta \mathrel{[\!]} \mathsf{Arg}\zeta \mathrel{[\!]} |\zeta|$$

$$\zeta := (\xi + \mathsf{i}\xi) \mathrel{[\!]} \xi\mathsf{e}^{\mathsf{i}\xi} \mathrel{[\!]} \langle\omega|\mathsf{t}\rangle \mathrel{[\!]} \zeta/\xi \mathrel{[\!]} (\zeta + \zeta) \mathrel{[\!]} (\zeta\,\zeta) \mathrel{[\!]} (\beta \triangleright \zeta; \zeta)$$

with the proviso that ω is a valuation term and r is a computational real constant. Most of these terms are self-explanatory. The only term that needs explanation is the alternative term $(\beta \rhd \zeta_1; \zeta_2)$. This term evaluates to ζ_1 if β is true and to ζ_2 otherwise.

The set of *pure state formulas* is built from boolean expression β and comparison formulas $(\xi \leq \xi)$ using the propositional connectives (f and $\sqsupset$):

$$\gamma := \beta \,[\!]\, (\xi \leq \xi) \,[\!]\, \mathsf{f} \,[\!]\, (\gamma \sqsupset \gamma).$$

The boolean expression β is true if evaluates to 1 on the classical valuation of the pure state. The interpretation of the other formulas are as expected. As usual, other classical connectives $(\boxminus, \sqcup, \sqcap, \equiv)$ are introduced as abbreviations. For instance, $(\boxminus\, \gamma)$ stands for $(\gamma \sqsupset \mathsf{f})$.

For the syntax of the *ensemble formulas* we use terms that reason about expected values. In particular we have the term $(\oint \xi \, \mathrm{d}\gamma)$ which denotes the expected value of ξ over the sub-probability measure induced by γ (*i.e.*, by eliminating the measure on the part where γ is not satisfied). The set of *expected terms* is given by:

$$p := \mathsf{r} \,[\!]\, (\oint \xi \, \mathrm{d}\gamma) \,[\!]\, (p + p) \,[\!]\, (p\, p).$$

The ensemble formulas are built from necessity formula $(\odot\, \gamma)$, conditional formulas (δ/γ) and comparison formulas $(p \leq p)$ using the propositional connectives (ff and $\supset$):

$$\delta := (\odot\, \gamma) \,[\!]\, (\delta/\gamma) \,[\!]\, (p \leq p) \,[\!]\, \mathsf{ff} \,[\!]\, (\delta \supset \delta).$$

The necessity formula $(\odot\, \gamma)$ is valid if γ is true with probability 1, the condition a formula (δ/γ) is valid if it is valid on the sub-probability measure induced by γ and the interpretation of the other formulas is as expected. Please note that $(\odot\, \gamma)$ is not a modality (for example, we do not have the formula $(\odot(\odot\, \gamma))$). We chose to confuse probability with possibility for the sake of simplicity. It is possible to maintain the distinction as was done in [25,6]. As usual, other classical connectives $(\ominus, \cup, \cap, \approx)$ are introduced as abbreviations. For instance, $(\ominus\, \gamma)$ stands for $(\gamma \supset \mathsf{ff})$.

3.2 Semantics

We are ready to give precise semantics of the language. We start by defining the semantics of real and complex terms, and pure state formulas. Given a pure state $\sigma = (v, |\psi\rangle)$, the denotation of complex and real terms and pure state formulas is as follows:

- $[\![\mathsf{r}]\!]_\sigma = \mathsf{r}$;
- $[\![\nu]\!]_{(v,|\psi\rangle)} = [\![\nu]\!]_v$;
- $[\![\xi_1 + \xi_2]\!]_\sigma = [\![\xi_1]\!]_\sigma + [\![\xi_2]\!]_\sigma$;
- $[\![\xi_1\, \xi_2]\!]_\sigma = [\![\xi_1]\!]_\sigma\, [\![\xi_2]\!]_\sigma$;
- $[\![\mathsf{Re}\zeta]\!]_\sigma = \mathsf{Re}[\![\zeta]\!]_\sigma$;
- $[\![\mathsf{Im}\zeta]\!]_\sigma = \mathsf{Im}[\![\zeta]\!]_\sigma$;
- $[\![\mathsf{Arg}\zeta]\!]_\sigma = \mathsf{Arg}[\![\zeta]\!]_\sigma$;
- $[\![|\zeta|]\!]_\sigma = |[\![\zeta]\!]_\sigma|$;
- $[\![\xi_1 + \mathsf{i}\xi_2]\!]_\sigma = [\![\xi_1]\!]_\sigma + \mathsf{i}[\![\xi_2]\!]_\sigma$;
- $[\![\xi_1 \mathsf{e}^{\mathsf{i}\xi_2}]\!]_\sigma = [\![\xi_1]\!]_\sigma \mathsf{e}^{\mathsf{i}[\![\xi_2]\!]_\sigma}$;
- $[\![\langle\omega|\mathsf{t}\rangle]\!]_\sigma = \langle[\![\omega]\!]|\psi\rangle$;
- $[\![\zeta_1 + \zeta_2]\!]_\sigma = [\![\zeta_1]\!]_\sigma + [\![\zeta_2]\!]_\sigma$;
- $[\![\zeta_1\, \zeta_2]\!]_\sigma = [\![\zeta_1]\!]_\sigma\, [\![\zeta_2]\!]_\sigma$;
- $[\![(\beta \rhd\ \zeta_1;\ \zeta_2)]\!]_{(v,|\psi\rangle)} = \begin{cases} [\![\zeta_1]\!]_\sigma \text{ if } [\![\beta]\!]_v = 1 \\ [\![\zeta_2]\!]_\sigma \text{ otherwise} \end{cases}$;
- $[\![\beta]\!]_{(v,|\psi\rangle)} = [\![\beta]\!]_v$;
- $[\![\xi_1 \leq \xi_2]\!]_\sigma = ([\![\xi_1]\!]_\sigma \leq [\![\xi_2]\!]_\sigma)$;
- $[\![\mathsf{f}]\!]_\sigma = 0$;
- $[\![\gamma_1 \sqsupset \gamma_2]\!]_\sigma = ([\![\gamma_1]\!]_\sigma \leq [\![\gamma_2]\!]_\sigma)$.

For the semantics of ensemble formulas we need the definition of the measure induced by a pure state formula γ. Given an ensemble ρ and a pure state formula γ, the measure induced by γ, ρ_γ, is the sub-measure of ρ such that

$$\rho_\gamma(\sigma) = \begin{cases} \rho(\sigma) \text{ if } [\![\gamma]\!]_\sigma = 1 \\ 0 \quad\ \ \text{otherwise} \end{cases}.$$

Finally, given an ensemble ρ, the denotation of expected terms and ensemble formulas is as follows:

- $[\![\mathsf{r}]\!]_\rho = \mathsf{r}$;
- $[\![(\oint \xi\, \mathrm{d}\gamma)]\!]_\rho = \sum_{\sigma\in\Sigma} [\![\xi]\!]_\sigma \rho_\gamma(\sigma)$;
- $[\![p_1 + p_2]\!]_\rho = [\![p_1]\!]_\rho + [\![p_2]\!]_\rho$;
- $[\![p_1 p_2]\!]_\rho = [\![p_1]\!]_\rho [\![p_2]\!]_\rho$;
- $[\![(\odot\, \gamma)]\!]_\rho = ([\![(\oint 1\, \mathrm{d}\gamma)]\!]_\rho = \rho(\Sigma))$;

- $[\![(\delta/\gamma)]\!]_\rho = [\![\delta]\!]_{\rho_\gamma}$;
- $[\![p_1 \leq p_2]\!]_\rho = ([\![p_1]\!]_\rho \leq [\![p_2]\!]_\rho)$;
- $[\![\mathsf{ff}]\!]_\rho = 0$;
- $[\![\delta_1 \supset \delta_2]\!]_\rho = ([\![\delta_1]\!]_\rho \leq [\![\delta_2]\!]_\rho)$.

3.3 Axiomatization

We need three new concepts for the axiomatization, one of *expected tautology*, a second of *valid analytical formulas* and *ground substitutions* and third of *valid pure state formulas.*

Consider propositional formulas built from a countable set of propositional symbols Q using the classical connectives $\perp$ and $\rightarrow$. An expected formula δ is said to be a *expected tautology* if there is a propositional tautology ε over Q and a map χ from Q to the set of expected state formulas such that δ coincides with ε_χ where ε_χ is the expected formula obtained from ε by replacing all occurrences of $\perp$ by ff, $\rightarrow$ by $\supset$ and $q \in$ Q by $\chi(q)$. For instance, the expected formula $((y_1 \leq y_2) \supset (y_1 \leq y_2))$ is tautological (obtained, for example, from the propositional tautology $q \rightarrow q$).

Now, assume a countable set of variables $X = \{x_k : k \in \mathbb{N}\}$, and consider the following set of formulas built from X:

$$\kappa := (a \leq a) \;[\!]\; \mathsf{ff} \;[\!]\; (\kappa \supset \kappa)$$
$$a := x \;[\!]\; \mathsf{r} \;[\!]\; (a + a) \;[\!]\; (aa).$$

We shall call the formulas of this language *analytical formulas.* Analytical formulas are interpreted over real numbers, and for that we need to give values to the logical variables x_k. We say that a map $u : X \rightarrow \mathbb{R}$ is an *assignment.* The interpretation of an analytical formula κ given an assignment u is straightforward. We say that an analytical formula κ is *valid* if it is valid for every assignment u. A *ground substitution* σ is map from the set of variable X to the set of expected terms. The substitution σ can then be extended (inductively) from the set of analytical formulas to the set of expected formulas.

A pure state formula γ is said to be valid if it is true of all pure state $\rho \in \Sigma$. We will not go into the axiomatization of the valid state formulas, although a complete recursive axiomatization can be obtained by adapting the axiomatization in [24,7].

The axioms and inference rules of EEQPL are listed in Table 1 and better understood in the following groups.

The axiom **ETaut** says that an expected tautology is an axiom. Since the

Axioms

[**ETaut**]	$\vdash$	δ for each expected tautology δ
[**Oracle**]	$\vdash$	$\sigma(\kappa)$ where κ is a valid analytical formula, and σ is a ground substitution.
[**Meas∅**]	$\vdash$	$((\oint \xi \, \mathrm{d}\mathsf{f}) = 0)$
[**FAdd**]	$\vdash$	$(((\oint \xi \, \mathrm{d}(\gamma_1 \sqcap \gamma_2)) = 0) \supset ((\oint \xi \, \mathrm{d}(\gamma_1 \sqcup \gamma_2)) = (\oint \xi \, \mathrm{d}\gamma_1) + (\oint \xi \, \mathrm{d}\gamma_2)))$
[**Mon1**]	$\vdash$	$((\odot(\gamma_1 \sqsupset \gamma_2)) \supset ((\oint 1 \, \mathrm{d}\gamma_1) \leq (\oint 1 \, \mathrm{d}\gamma_2)))$
[**Mon2**]	$\vdash$	$((\odot(\gamma \sqsupset (\xi_1 \leq \xi_2))) \supset ((\oint \xi_1 \, \mathrm{d}\gamma) \leq (\oint \xi_2 \, \mathrm{d}\gamma)))$
[**Lin**]	$\vdash$	$((\oint (\mathsf{r}_1\xi_1 + \mathsf{r}_2\xi_2) \, \mathrm{d}\gamma) = (\mathsf{r}_1(\oint \xi_1 \, \mathrm{d}\gamma) + \mathsf{r}_2(\oint \xi_2 \, \mathrm{d}\gamma)))$
[**QTaut**]	$\vdash$	$\odot \gamma$ for every valid pure state formula
[**ElimNec**]	$\vdash$	$\odot \gamma \approx ((\oint 1 \, \mathrm{d}\gamma) = (\oint 1 \, \mathrm{d}\mathsf{t}))$
[**Dist⊃**]	$\vdash$	$(((\delta_1 \supset \delta_2)/\gamma) \approx ((\delta_1/\gamma) \supset (\delta_2/\gamma)))$
[**ElimCond**]	$\vdash$	$(((p_1 \leq p_2)/\gamma) \approx ((p_1 \leq p_2)\vert^{(\oint \xi \, \mathrm{d}\gamma_1)}_{(\oint \xi \, \mathrm{d}(\gamma_1 \sqcap \gamma))}))$

Inference rules

[**PMP**]	$\delta_1, (\delta_1 \supset \delta_2) \vdash \delta_2$
[**Cond**]	$\vdash \delta/\gamma$ whenever $\vdash \delta$

Table 1

set of probabilistic tautologies is recursive, there is no need to spell out the details of tautological reasoning.

The axiom **Oracle** says that if κ is a valid analytical formula and σ is a ground substitution then $\sigma(\kappa)$ is a valid EEQPL formula. The axiom **Oracle** is controversial as the set of valid analytical formulas is not recursively enumerable[5]. A recursive axiomatization is possible if we work in arbitrary real closed fields instead of real numbers. However, this is out of scope of this paper.

The axiom **Meas∅** says that the probability of empty set is 0. The axiom **FAdd** is the consequence of finite additivity of probabilities. The axiom **Mon1** is a consequence of monotonicity of probabilities. The axiom **Mon2** is the monotonicity of expectations and the axiom **Lin** is linearity of expectations.

[5] For example, equality of two computational real numbers is undecidable

The axiom **QTaut** relates the pure state formulas with the expected formulas. Please note that since the set of valid pure formulas is also not recursively enumerable, this is also like the axiom **Oracle**. However, we can achieve a recursive axiomatization if we work with algebraic closed fields [7]. The axiom **ElimNec** allows us to rewrite a necessity formula as a comparison formula, and hence it is like an elimination rule.

The axiom **Dist**$\supset$ says that the connective $\supset$ distributes over the conditional construct. The axiom **ElimCond** eliminates the conditional construct. The expected term

$$(p_1 \leq p_2)|^{(\oint \xi \, \mathrm{d}\gamma_1)}_{(\oint \xi \, \mathrm{d}(\gamma_1 \sqcap \gamma))}$$

in **ElimCond** is the term obtained by replacing <u>all</u> occurrences of $(\oint \xi \, \mathrm{d}\gamma_1)$ by $(\oint \xi \, \mathrm{d}(\gamma_1 \sqcap \gamma))$ for <u>each</u> classical state formula γ_1.

The inference rule **PMP** ist the *modus ponens* for expected implication. The inference rule **Cond** says that if ξ is an theorem, then so is ξ/γ. This inference rule is useful for showing the meta-theorem of equivalence and is similar to the generalization rule in modal logics.

As usual we say that a (possibly infinite) set of formulas Γ *derives* γ, written $\Gamma \vdash \delta$, if we can build a derivation of δ from axioms and the inference rules using formulas in Γ as hypothesis. Please note that while applying the rule **Cond**, we are allowed to use only theorems of the logic (and not any hypothesis or any intermediate step in the derivation). The above set of axioms and rules is sound:

Theorem 3.1 EEQPL is sound.

Please note that we can also show as a result of the elimination rules (**ElimNec** and **ElimCond**) that each expected formula δ is equivalent to a formula η without any necessity and conditional sub-formulas.

The above set of axioms and rules will be weakly complete if we assume that the set of real and complex values range over a finite set. This restriction is enough to reason about a large number of quantum programs and protocols. The proof of the weak completeness of this restricted EEQPL follows the style of [24,7,6], but is out of the scope of this paper. The completeness of this axiomatization without this restriction is an open problem. However, previous work in the context of probabilistic logics [1] hints that a recursive axiomatization may not be possible.

4 Quantum Hoare logic

We are ready to define the Hoare logic. As usual, the Hoare assertions are:

$$\theta := \delta \mathrel{[\!]} \{\delta\}\, s\, \{\delta\}.$$

The satisfaction of Hoare assertions is defined as:

- $\rho u \Vdash \delta$ iff $[\![\delta]\!]_{u\rho} = 1$;
- $\rho u \Vdash \{\delta_1\}\, s\, \{\delta_2\}$ iff $([\![s]\!]_u(\rho))u \Vdash \delta_2$ whenever $\rho u \Vdash \delta_1$.

The semantic entailment is defined as expected.

4.1 Axiomatization

A sound Hoare calculus for our quantum programming language is defined below. We discuss the axioms and inference rules below.

Axioms.

The rules for skip and assignments are standard. The interesting axioms are for unitary transformations (**UNIT**) and measurement (**MEASB**).

Unitary Transformation. The axiom for unitary transformation is like a simultaneous assignment where the amplitude terms are updated to new amplitude terms according to the unitary transformation. For the sake of simplicity, we just give the axiom for the unitary operators on qubits, and the axiom for unitary operators on qunits can be similarly obtained.

Please note that in the rule for unitary transformation (**UNIT**) the formula $\delta|^{\langle\omega|\mathsf{t}\rangle}_{\langle U\omega|\mathsf{t}\rangle}$ is the formula obtained by replacing <u>every</u> occurrence of <u>each</u> amplitude term $\langle\omega|\mathsf{t}\rangle$ by $\langle U\omega|\mathsf{t}\rangle$ where the amplitude term $\langle U\omega|\mathsf{t}\rangle$ is defined by induction on U. For the sake of brevity, we consider here the base cases where U is the Hadamard and phase shift operator.

For the Hadamard operator, $(\mathsf{H} : \mathbf{qb})$, the amplitude term $\langle(\mathsf{H} : \mathbf{qb})\omega|\mathsf{t}\rangle$ is defined as:

- $\langle(\mathsf{H} : \mathbf{qb})(\omega_1(\mathbf{qb} : 0)\omega_2)|\mathsf{t}\rangle$ is $\frac{1}{\sqrt{2}}\langle\omega_1(\mathbf{qb} : 0)\omega_2|\mathsf{t}\rangle + \frac{1}{\sqrt{2}}\langle\omega_1(\mathbf{qb} : 1)\omega_2|\mathsf{t}\rangle$
- $\langle(\mathsf{H} : \mathbf{qb})(\omega_1(\mathbf{qb} : 1)\omega_2)|\mathsf{t}\rangle$ is $\frac{1}{\sqrt{2}}\langle\omega_1(\mathbf{qb} : 0)\omega_2|\mathsf{t}\rangle - \frac{1}{\sqrt{2}}\langle\omega_1(\mathbf{qb} : 1)\omega_2|\mathsf{t}\rangle$

For the phase shift operator $(\mathsf{S} : \mathbf{qb}(\nu, \beta))$ and the case where ω contains $(\mathbf{qb} : 0)$, the amplitude term $\langle(\mathsf{S} : \mathbf{qb}(\nu, \beta))\omega|\mathsf{t}\rangle$ is defined as:

- $\langle(\mathsf{S} : \mathbf{qb}(\nu, \beta))(\omega_1(\mathbf{qb} : 0)\omega_2)|\mathsf{t}\rangle$ is

$$((\boxminus \beta) \sqcap (\nu = 0)) \rhd e^{i2\pi/1}\langle(\omega_1(\mathbf{qb} : 0)\omega_2)|\mathsf{t}\rangle$$
$$(((\boxminus \beta) \sqcap (\nu = 1)) \rhd e^{i2\pi/2}\langle(\omega_1(\mathbf{qb} : 0)\omega_2)|\mathsf{t}\rangle$$
$$\ldots$$
$$(((\boxminus \beta) \sqcap (\nu = (N-1))) \rhd e^{i2\pi/N}\langle(\omega_1(\mathbf{qb} : 0)\omega_2)|\mathsf{t}\rangle$$
$$(\omega_1(\mathbf{qb} : 0)\omega_2))\ldots))$$

The case where ω contains $(\mathbf{qb} : 1)$ is similar.

Measurement. The axiom for measurement (**MEASB**) is inspired by the axiom for the probabilistic toss [6]. Measurement is like a probabilistic assignment: it sets the the qubit being measured to 1 with certain probability. The formula δ^b_β is the formula obtained by replacing every occurrence of b by β, and the formula δ^n_ν is the formula obtained by replacing every occurrence of n by ν. For the sake of simplicity, we just give the axiom for the measurement on qubits, and the axiom for unitary operators on qunits can be similarly obtained.

The first thing to note in **MEASB** is that we just consider formulas η without necessity and conditional sub-formulas. This is not a limitation as every EEQPL formula is equivalent to one such formula (see Section 3.3). Also in the rule, the formula

$$\eta|^{(\oint \xi \, \mathrm{d}\gamma)}_{m_1^{\mathbf{b},\mathbf{qb}}((\oint \xi \, \mathrm{d}\gamma))+m_0^{\mathbf{b},\mathbf{qb}}((\oint \xi \, \mathrm{d}\gamma))}$$

is the formula in which $\underline{\text{every}}$ occurrence of $\underline{\text{each}}$ expected term $(\oint \xi \, \mathrm{d}\gamma)$ is replaced by the expected term $m_1^{\mathbf{b},\mathbf{qb}}((\oint \xi \, \mathrm{d}\gamma)) + m_0^{\mathbf{b},\mathbf{qb}}((\oint \xi \, \mathrm{d}\gamma)$ where the expected terms $m_1^{\mathbf{b},\mathbf{qb}}((\oint \xi \, \mathrm{d}\gamma))$ and $m_0^{\mathbf{b},\mathbf{qb}}((\oint \xi \, \mathrm{d}\gamma))$ are defined below.

The term $m_1^{\mathbf{b},\mathbf{qb}}((\oint \xi \, \mathrm{d}\gamma))$ is defined to deal with the (probabilistic) case in which the measurement of the qubit $\mathbf{qb}$ yields 1. The definition uses three auxiliary definitions: $p_1^{\mathbf{qb}}$, $m_1^{\mathbf{b},\mathbf{qb}}(\xi)$ and $m_1^{\mathbf{b},\mathbf{qb}}(\gamma)$. Intuitively, $p_1^{\mathbf{qb}}$ is the probability of the measurement yielding 1. If the result of the measurement is 1, then the bit $\mathbf{b}$ gets set to 0 and the amplitude terms get re-normalized. The terms $m_1^{\mathbf{b},\mathbf{qb}}(\xi)$ and $m_1^{\mathbf{b},\mathbf{qb}}(\gamma)$ are defined mutually recusively to deal with this "assignment". Formally,

- $m_1^{\mathbf{b},\mathbf{qb}}((\oint \xi \, \mathrm{d}\gamma)) = (\oint p_1^{\mathbf{qb}} m_1^{\mathbf{b},\mathbf{qb}}(\xi) \, \mathrm{d}(m_1^{\mathbf{b},\mathbf{qb}}(\gamma)))$
- $m_1^{\mathbf{b},\mathbf{qb}}(\mathbf{b}) = \mathsf{t}$;
- $m_1^{\mathbf{b},\mathbf{qb}}(\langle\omega_1(\mathbf{qb} : 0)\omega_2|\mathsf{t}\rangle) = 0$;
- $m_1^{\mathbf{b},\mathbf{qb}}(\langle\omega_1(\mathbf{qb} : 1)\omega_2|\mathsf{t}\rangle) = \frac{m_1^{\mathbf{b},\mathbf{qb}}(\langle\omega_1(\mathbf{qb}:1)\omega_2|\mathsf{t}\rangle)}{\sqrt{p_1^{\mathbf{qb}}}}$;

Axioms

[**TAUT**]	$\vdash$	δ if δ is a EEQPL tautology;
[**SKIP**]	$\vdash$	$\{\delta\}\,\mathsf{skip}\,\{\delta\}$;
[**ASGB**]	$\vdash$	$\{\delta^{\mathbf{b}}_{\beta}\}\,\mathbf{b} \leftarrow \beta\,\{\delta\}$;
[**ASGR**]	$\vdash$	$\{\delta^{\mathbf{n}}_{\nu}\}\,\mathbf{n} \leftarrow \nu\,\{\delta\}$;
[**UNIT**]	$\vdash$	$\{\delta\vert^{\langle\omega\vert\mathsf{t}\rangle}_{\langle U\omega\vert\mathsf{t}\rangle}\}\,U\,\{\delta\}$
[**MEASB**]	$\vdash$	$\{\eta\vert^{(\oint \xi\,\mathrm{d}\gamma)}_{m_1^{\mathbf{b},\mathbf{qb}}((\oint \xi\,\mathrm{d}\gamma))+m_0^{\mathbf{b},\mathbf{qb}}((\oint \xi\,\mathrm{d}\gamma))}\}\,\mathbf{b} \Leftarrow \mathbf{qb}\,\{\eta\}$

Inference rules

[**SEQ**]	$\{\delta_0\}\,s_1\,\{\delta_1\}, \{\delta_1\}\,s_2\,\{\delta_2\}$	$\vdash$	$\{\delta_0\}\,s_1; s_2\,\{\delta_2\}$
[**IF**]	$\{\eta_0\}\,s_1; \mathbf{b} \leftarrow \mathsf{t}\,\{\eta_2\}$		
	$\{\eta_1\}\,s_2; \mathbf{b} \leftarrow \mathsf{f}\,\{\eta_3\}$	$\vdash$	$\{\eta_0 \curlyvee_{\mathbf{b}} \eta_1\}\,(\mathsf{if}\ \mathbf{b}\ \mathsf{then}\ s_1\ \mathsf{else}\ s_2)\,\{\eta_2 \curlyvee_{\mathbf{b}} \eta_3\}$;
[**CONS**]	$\delta_0 \supset \delta_1, \{\delta_1\}\,s\,\{\delta_2\}, \delta_2 \supset \delta_3$	$\vdash$	$\{\delta_1\}\,s\,\{\delta_3\}$;
[**OR**]	$\{\delta_0\}\,s\,\{\delta_2\}, \{\delta_1\}\,s\,\{\delta_2\}$	$\vdash$	$\{\delta_0 \cup \delta_1\}\,s\,\{\delta_2\}$;
[**AND**]	$\{\delta_0\}\,s\,\{\delta_1\}, \{\delta_0\}\,s\,\{\delta_2\}$	$\vdash$	$\{\delta_0\}\,s\,\{\delta_1 \cap \delta_2\}$

Table 2

- $p_1^{\mathbf{qb}} = \sum_{\omega:\omega\downarrow_{\mathbf{qb}}=1} |\langle\omega|\mathsf{t}\rangle|^2$ where $\omega \downarrow_{\mathbf{qb}}= 1$ if $\omega = \omega_1(\mathbf{qb} : 1)\omega_2$ and 0 otherwise.

We have just given the base cases for $m_1^{\mathbf{b},\mathbf{qb}}(\xi)$ and $m_1^{\mathbf{b},\mathbf{qb}}(\gamma)$ for the sake of brevity. The term $m_0^{\mathbf{b},\mathbf{qb}}((\oint \xi\,\mathrm{d}\gamma))$ can be defined similarly.

Inference Rules.

The inference rules **Seq**, **CONS**, **OR** and **AND** are standard. The inference rule **IF** for the alternative construct (if **b** then ... else ...) is inspired by the inference rule for the alternative construct in [6]. Please note that in the inference rule **IF**, we just consider formulas without any necessity and conditional sub-formulas. Furthermore, the formula $\eta \curlyvee_{\mathbf{b}} \eta'$ is an abbreviation for the formula $((\eta/\mathbf{b}) \cap (\eta'/\boxminus\mathbf{b}))$.

We have not given the rules for the alternative construct case and the iterative construct repeat . The construct case can be written as an abbreviation using the alternative construct if. The iterative construct repeat in turn can be written as an abbreviation using the construct case and sequential composition.

The proof of soundness of the Hoare-calculus follows closely that in [6] for probabilistic programs. The only interesting cases are the measurement axiom and the if-then-else construct. The soundness of the measurement axiom can be shown by adapting the proof for the probabilistic-toss axiom in [6], and the same holds for the alternative constructs.

Theorem 4.1 The Hoare-calculus is sound.

5 Worked example

We illustrate our calculus with an example, the Deutsch Algorithm.

Deutsch algorithm. The purpose of Deutsch algorithm [10] is to check whether a boolean function f with one input bit $\mathbf{b}$ is constant or not. Classically it involves computing f on the two different inputs and checking if the results are the same or not. In quantum, we just require one evaluation of f along with two auxiliary qubtis and a unitary transformation U_f constructed from f. The unitary transformation U_f acts on two qubits and converts the computational basis element $|x, y\rangle$ to $|x, y \oplus f(x)\rangle$. Assuming that the two qubits $\mathbf{qb}_1$ and $\mathbf{qb}_2$, the Deutsch algorithm D in our programming language can be written as:

$$\begin{aligned} D \overset{\text{df}}{=} & (\mathsf{H} : \mathbf{qb}_2);\ (\mathsf{H} : \mathbf{qb}_1);\\ & \mathsf{U}_f : (\mathbf{qb}_1, \mathbf{qb}_2);\\ & (\mathsf{H} : \mathbf{qb}_1);\ \mathbf{b} \Leftarrow \mathbf{qb}_1 \end{aligned}$$

Initially the qubits are in the state $|0, 1\rangle$. At the end of the Deutsch algorithm, the bit $\mathbf{b}$ tells whether the function is constant of or not. If it is 0, then the function is constant, otherwise is not constant. Therefore if you assume that we have the function symbol f in EEQPL, then we have to show

$$\{\langle(\mathbf{qb}_1 : 0, \mathbf{qb}_2 : 1)|\mathsf{t}\rangle = 1 \cap \odot\,(f(0) = f(1))\}\, D\, \{\odot(\mathbf{b} = 0)\}$$

and

$$\{\langle(\mathbf{qb}_1 : 0, \mathbf{qb}_2 : 1)|\mathsf{t}\rangle = 1 \cap \odot(f(0) \neq f(1))\}\, D\, \{\odot(\mathbf{b} = 1)\}.$$

For the sake of brevity, we are going only to show the first goal for the case that $f(0) = f(1) = 1$. We can add the function symbol f and also add the unitary transformation U_f in our language. However, in order to avoid a tedious approach, we will just avoid $f(0) = f(1) = 1$ in the precondition and extend the definition of $\langle U\omega|\mathsf{t}\rangle$ where

- $\langle U_f(\mathbf{qb}_1 : 0, \mathbf{qb}_2 : 0)|\mathsf{t}\rangle = \langle \mathbf{qb}_1 : 0, \mathbf{qb}_2 : 1)|\mathsf{t}\rangle$;

- $\langle U_f(\mathbf{qb}_1 : 0, \mathbf{qb}_2 : 1)|\mathsf{t}\rangle = \langle \mathbf{qb}_1 : 0, \mathbf{qb}_2 : 0)|\mathsf{t}\rangle$.

The case where $\mathbf{qb}_1 : 1$ appears in the amplitude term can be easily defined.

Therefore we will just illustrate the judgment:

$$\{\odot(\langle \mathbf{qb}_1 : 0, \mathbf{qb}_2 : 1|\mathsf{t}\rangle = 1)\}\, D\, \{(\odot(\mathbf{b} = 0))\}.$$

In the proof, we will abbreviate $\langle \mathbf{qb}_1 : x, \mathbf{qb}_2 : y|\mathsf{t}\rangle$ as $\langle xy|\mathsf{t}\rangle$. Please, also recall the expected terms $p_1^{\mathbf{qb}_1}$ and $p_0^{\mathbf{qb}_1}$ defined in Section 4.

1.	$(\odot(\mathbf{b} = 0)) \approx ((\oint 1\, \mathrm{d}(\mathbf{b} = 0)) = (\oint 1\, \mathrm{d}(\mathsf{t})))$	TAUT
2.	$\{((\oint p_0^{\mathbf{qb}_1}\, \mathrm{d}(0 = 0)) + (\oint p_1^{\mathbf{qb}_1}\, \mathrm{d}(1 = 0))) = (\oint 1\, \mathrm{d}(\mathsf{t}))\}$ $\mathbf{b} \leftleftarrows \mathbf{qb}_1\, \{((\oint 1\, \mathrm{d}(\mathbf{b} = 0)) = (\oint 1\, \mathrm{d}(\mathsf{t})))\}$	MEASB
3.	$\odot(p_0^{\mathbf{qb}_1} = 1) \supset$ $(((\oint p_0^{\mathbf{qb}_1}\, \mathrm{d}(0 = 0)) + (\oint p_1^{\mathbf{qb}_1}\, \mathrm{d}(1 = 0))) = (\oint 1\, \mathrm{d}(\mathsf{t})))$	TAUT
4.	$\{\odot(p_0^{\mathbf{qb}_1} = 1)\}\, \mathbf{b} \leftleftarrows \mathbf{qb}_1\, \{((\oint 1\, \mathrm{d}(\mathbf{b} = 0)) = (\oint 1\, \mathrm{d}(\mathsf{t})))\}$	CONS: 3,2
5.	$\{\odot(\frac{1}{2}(\lvert\langle 00|\mathsf{t}\rangle + \langle 10|\mathsf{t}\rangle\rvert^2 + \lvert\langle 01|\mathsf{t}\rangle + \langle 11|\mathsf{t}\rangle\rvert^2)) = 1\}$ $\mathsf{H} : \mathbf{qb}_1\, \{\odot(p_0^{\mathbf{qb}_1} = 1)\}$	UNIT
6.	$\{\odot(\frac{1}{2}(\lvert\langle 01|\mathsf{t}\rangle + \langle 11|\mathsf{t}\rangle\rvert^2 + \lvert\langle 00|\mathsf{t}\rangle + \langle 10|\mathsf{t}\rangle\rvert^2)) = 1\}$ $\mathsf{U}_f : (\mathbf{qb}_1, \mathbf{qb}_2)\, \{\odot(p_0^{\mathbf{qb}_1} = 1)\}$	UNIT
7.	$\{\odot(\lvert\langle 01|\mathsf{t}\rangle\rvert^2 + \lvert\langle 00|\mathsf{t}\rangle\rvert^2) = 1\}\, \mathsf{H} : \mathbf{qb}_1$ $\{\odot(\frac{1}{2}(\lvert\langle 01|\mathsf{t}\rangle + \langle 11|\mathsf{t}\rangle\rvert^2 + \lvert\langle 00|\mathsf{t}\rangle + \langle 10|\mathsf{t}\rangle\rvert^2)) = 1\}$	UNIT,TAUT
8.	$\{\odot(\lvert\langle 00|\mathsf{t}\rangle\rvert^2 + \lvert\langle 01|\mathsf{t}\rangle\rvert^2) = 1\}\, \mathsf{H} : \mathbf{qb}_2\, \{\odot(\lvert\langle 01|\mathsf{t}\rangle\rvert^2 + \lvert\langle 00|\mathsf{t}\rangle\rvert^2) = 1\}$	UNIT,TAUT
9.	$\odot(\langle 01|\mathsf{t}\rangle = 1) \supset (\odot(\lvert\langle 00|\mathsf{t}\rangle\rvert^2 + \lvert\langle 01|\mathsf{t}\rangle\rvert^2) = 1)$	TAUT
10.	$\{\odot(\langle 01|\mathsf{t}\rangle = 1)\}\, D\, \{(\odot(\mathbf{b} = 0))\}$	SEQ:1-9

6 Outlook

Our main contribution is to provide a practical and sound quantum Hoare calculus including a sound axiomatization for the state assertion language. The state logic is obtained using the exogenous semantics approach to enriching a given logic [23,25,7] by attaching probabilities to the pure states, that is, by adopting ensembles as models. In our case, the given logic is a variation of the exogenous quantum propositional logic presented in [24]. Completeness is also achieved if we assume a finite number of possible quantum and classical valuations.

The novelties of the Hoare calculus are the axioms for the unitary transformations and measurement operators. The axiom for unitary transformation is an extension of the classical assignment rule, while the axiom for the measurement is an extension of the rule for probabilistic toss presented in [6]. The inference rule for the alternative if is borrowed from [6].

We believe that a complete axiomatization for the Hoare calculus for the bounded iteration language can be achieved with further extensions of the state logic. Although our programming language is rich enough to encode several well-known quantum algorithms, we plan to investigate the unbounded iteration construct. We also envisage to replace ensembles with density operators as the models of the state logic.

We view the state calculus and the Hoare calculus presented herein as the first steps towards theorem proving and automated verification (namely, model-checking) of quantum programs and protocols. The next logical step is to investigate dynamic logics [13,22,16] and also to develop modal logics for quantum process algebras [19,3,14]. Another research area that we plan to investigate in the long term is quantum temporal logics following the development of probabilistic temporal logics [15,18,27].

References

[1] M. Abadi and J. Y. Halpern. Decidability and expressiveness for first-order logics of probability. *Information and Computation*, 112(1):1–36, 1994.

[2] S. Abramsky and B. Coecke. A categorical semantics of quantum protocols. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science (LICS 2004)*, pages 415–425. IEEE Computer Science Press, 2004. Extended version at arXiv:quant-ph/0402130).

[3] P. Adão and P. Mateus. A process algebra for reasoning about quantum security. *Electronic Notes in Theoretical Computer Science*, to appear. Preliminary version presented at 3rd International Workshop on Quantum Programming Languages, June 30 - July 1, 2005, Chicago, Affiliated Workshop of LICS 2005.

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

[5] A. Baltag and S. Smets. LQP: The dynamic logic of quantum information. *Mathematical Structures in Computer Science*, 2006. To appear.

[6] R. Chadha, P. Mateus, and A. Sernadas. Reasoning about states of probabilistic sequential programs. Preprint, CLC, Department of Mathematics, Instituto Superior Técnico, 1049-001 Lisboa, Portugal, 2006. Submitted for publication.

[7] R. Chadha, P. Mateus, A. Sernadas, and C. Sernadas. Extending classical logic for reasoning about quantum systems. Preprint, CLC, Department of Mathematics, Instituto Superior Técnico, 1049-001 Lisboa, Portugal, 2005. Invited submission to the Handbook of Quantum Logic.

[8] M. L. D. Chiara, R. Giuntini, and R. Greechie. *Reasoning in Quantum Theory.* Kluwer Academic Publishers, 2004.

[9] J.I. den Hartog and E.P. de Vink. Verifying probabilistic programs using a hoare like logic. *International Journal of Foundations of Computer Science*, 13(3):315–340, 2002.

[10] D. Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London A*, 400:97–117, 1985.

[11] E. D'Hondt and P. Panangaden. Quantum weakest preconditions. In Peter Selinger, editor, *Proceedings of the 2nd International Workshop on Quantum Programming Languages*, number 33 in TUCS General Publications, pages 75–90. Turku Centre for Computer Science, 2004.

[12] R. Fagin, J. Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Information and Computation*, 87(1-2):78–128, 1990.

[13] Y. A. Feldman and David Harel. A probabilistic dynamic logic. *Journal of Computer and System Sciences*, 28:193–215, 1984.

[14] S. J. Gay and R. Nagarajan. Communicating quantum processes. In *POPL '05: Proceedings of the 32nd ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 145–157, 2005.

[15] H. Hanssohn and B. Johnsson. A logic for reasoning about time and reliability. *Formal Aspect of Computing*, 6(5):512–535, 1994.

[16] D. Harel, D. Kozen, and J. Tiuryn. *Dynamic logic*. Foundations of Computing Series. MIT Press, 2000.

[17] C. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12:576–583, 1969.

[18] M. Huth and M. Kwiatkowska. Quantitative analysis and model checking. In *12th Annual IEEE Symposium on Logic in Computer Science (LICS'97)*, pages 111–122, 1997.

[19] P. Jorrand and M. Lalire. Toward a quantum process algebra. In *CF '04: Proceedings of the 1st conference on Computing frontiers*, pages 111–119, 2004.

[20] E. Knill. Conventions for quantum pseudocode. Technical Report LAUR-96-2724, Los Alamos National Laboratory, 1996.

[21] D. Kozen. Semantics of probabilistic programs. *Journal of Computer System Science*, 22:328–350, 1981.

[22] D. Kozen. A probabilistic PDL. *Journal of Computer System Science*, 30:162–178, 1985.

[23] P. Mateus and A. Sernadas. Reasoning about quantum systems. In J. Alferes and J. Leite, editors, *Logics in Artificial Intelligence, Ninth European Conference, JELIA'04*, volume 3229 of *Lecture Notes in Artificial Intelligence*, pages 239–251. Springer-Verlag, 2004.

[24] P. Mateus and A. Sernadas. Weakly complete axiomatization of exogenous quantum propositional logic. *Information and Computation*, to appear.

[25] P. Mateus, A. Sernadas, and C. Sernadas. Exogenous semantics approach to enriching logics. In G. Sica, editor, *Essays on the Foundations of Mathematics and Logic*, volume 1 of *Advanced Studies in Mathematics and Logic*, pages 165–194. Polimetrica, 2005.

[26] C. Morgan, A. McIver, and K. Seidel. Probabilistic predicate transformers. *ACM Transactions on Programming Languages and Systems*, 18(3):325–353, 1996.

[27] M. Narasimha, R. Cleaveland, and P. Iyer. Probabilistic temporal logics via the modal mu-calculus. In *Foundations of Software Science and Computation Structures (FOSSACS 99)*, volume 1578 of *Lecture Notes in Computer Science*, pages 288–305. Springer, 1999.

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

[29] N. J. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28(1):71–87, 1986.

[30] L. H. Ramshaw. *Formalizing the analysis of algorithms.* PhD thesis, Stanford University, 1979.

[31] J. W. Sanders and P. Zuliani. Quantum programming. In *Mathematics of Program Construction*, volume 1837 of *Lecture Notes in Computer Science*, pages 80–99. Springer, 2000.

[32] P. Selinger and B. Valiron. A lambda calculus for quantum computation with classical control. In *Proceedings of the 7th International Conference on Typed Lambda Calculi and Applications (TLCA)*, volume 3461 of *Lecture Notes in Computer Science*, pages 354–368. Springer, 2005.

[33] R. van der Meyden and M. Patra. Knowledge in quantum systems. In M. Tennenholtz, editor, *Theoretical Aspects of Rationality and Knowledge*, pages 104–117. ACM, 2003.

[34] R. van der Meyden and M. Patra. A logic for probability in quantum systems. In M. Baaz and J. A. Makowsky, editors, *Computer Science Logic*, volume 2803 of *Lecture Notes in Computer Science*, pages 427–440. Springer-Verlag, 2003.