# Quantum Hoare logic with classical variables



YUAN FENG, University of Technology Sydney, Australia
MINGSHENG YING, University of Technology Sydney, Australia, Chinese Academy of Sciences, China, and Tsinghua University, China

Hoare logic provides a syntax-oriented method to reason about program correctness, and has been proven effective in verification of classical and probabilistic programs. Existing proposals for quantum Hoare logic either lack of completeness or support only quantum variables, thus limiting their capability in practical use. In this paper, we propose a quantum Hoare logic for a simple while language which involves both classical and quantum variables. Its soundness and relative completeness are proven for both partial and total correctness of quantum programs written in the language. Remarkably, with novel definitions of classical-quantum states and corresponding assertions, the logic system is quite simple and similar to the traditional Hoare logic for classical programs. Furthermore, to simplify reasoning in real applications, auxiliary proof rules are provided which support the introduction of disjunction and quantifiers in the classical part of assertions, and of super-operator application and superposition in the quantum part. Finally, a series of practical quantum algorithms, in particular the whole algorithm of Shor's factorisation, are formally verified to show the effectiveness of the logic.

Additional Key Words and Phrases: Hoare logic, quantum programming, verification

## 1 INTRODUCTION

Quantum computing and quantum communication provide potential speedup and enhanced security compared with their classical counterparts [Bennett 1992; Bennett and Brassard 1984; Grover 1996; Harrow et al. 2009; Shor 1994]. However, the quantum features which are responsible for these benefits, such as entanglement between different systems and non-commutativity of quantum operations, also make analysis of quantum algorithms and protocols notoriously difficult [Mayers 2001]. Furthermore, due to the lack of reliable and scalable hardware on which practical quantum algorithms can be executed, traditional techniques such as testing and debugging in classical software engineering will not be readily available in the near future, and formal methods based static analysis of quantum programs seems indispensable.

Among other techniques, Hoare logic provides a syntax-oriented proof system to reason about program correctness [Hoare 1969]. For classical (non-probabilistic) programs, the correctness is expressed in the Hoare triple form $\{P\}S\{Q\}$ where $S$ is a program, and $P$ and $Q$ are first-order logic formulas called *assertions* that describe the pre- and postconditions of $S$, respectively. Intuitively, the triple claims that if $S$ is executed at a state (valuation of program variables) satisfying $P$ and it terminates, then $Q$ must hold in the final state. This is called *partial correctness*. If termination is further guaranteed in all states that satisfy $P$, then partial correctness becomes a *total* one. After decades of development, Hoare logic has been successfully applied in analysis of programs with non-determinism, recursion, parallel execution, etc. For a detailed survey, we refer to [Apt et al. 2010; Apt and Olderog 2019].

Hoare logic was also extended to programming languages with probabilistic features. As the program states for probabilistic languages are (sub)distributions over valuations of program variables,

Authors' addresses: Yuan Feng, Centre for Quantum Software and Information, University of Technology Sydney, NSW, Australia, Yuan.Feng@uts.edu.au; Mingsheng Ying, Centre for Quantum Software and Information, University of Technology Sydney, NSW, Australia, Institute of Software, Chinese Academy of Sciences, Beijing, China, Department of Computer Science, Tsinghua University, Beijing, China, Mingsheng.Ying@uts.edu.au.

. 2643-6817//1-ART1 $15.00
https://doi.org/

the extension naturally follows two different approaches, depending on how assertions of probabilistic states are defined. The first one takes subsets of distributions as (qualitative) assertions, similar to the non-probabilistic case, and the satisfaction relation between distributions and assertions is then just the ordinary membership [Barthe et al. 2018; Chadha et al. 2007; Den Hartog and de Vink 2002; Ramshaw 1979]. In contrast, the other approach takes nonnegative functions on valuations as (quantitative) assertions. Consequently, one is concerned with the *expectation* of a distribution satisfying an assertion [Kaminski et al. 2018; McIver et al. 2005; Morgan et al. 1996]. In addition to the standard Hoare logic, the relational one, which relates two programs, was also proposed for the purpose of analysing program robustness and security [Barthe et al. 2009, 2012; Benton 2004].

In recent years, Hoare logic and relational Hoare logic for quantum programs have been developed, also following two different approaches similar to the probabilistic setting. Note that quantum (mixed) states are described mathematically by density operators in a finite dimensional Hilbert space. Assertions in the satisfaction-based logics proposed in [Chadha et al. 2006a; Kakutani 2009] extend the probabilistic counterparts in [Chadha et al. 2006b; Den Hartog and de Vink 2002] with the ability to reason about probabilities (or even the complex amplitudes) and expected values of measuring a quantum state. The satisfaction-based logics proposed in [Unruh 2019a,b; Zhou et al. 2019] regard subspaces of the Hilbert space as assertions, and a quantum state $\rho$ satisfies an assertion $P$ iff the support (the image space of linear operators) of $\rho$ is included in $P$. In contrast, the expectation-based approaches [Barthe et al. 2019; Li and Unruh 2019; Ying 2012, 2016, 2019; Ying et al. 2018] take positive operators as assertions for quantum states, following the observation of [D'Hondt and Panangaden 2006a], and the expectation of a quantum state $\rho$ satisfying an assertion $M$ is then defined to be $\mathrm{tr}(M\rho)$. A comparison of the quantum Hoare logics in [Chadha et al. 2006a; Kakutani 2009; Ying 2012] was provided in [Rand 2019].

**Limit of the existing quantum Hoare logics**: Existing quantum Hoare logics in the literature suffer from the following problems:

(1) *Completeness.* The logics proposed in [Chadha et al. 2006a; Kakutani 2009] support classical variables in the language. However, whether or not they are complete is still unknown. Completeness of the logic for a purely quantum language in [Unruh 2019a] has not been established either.
(2) *Expressiveness.* The quantum Hoare logics in [Ying 2012, 2016; Ying et al. 2018; Zhou et al. 2019] are complete, but the programming languages they consider do not support classical variables. Although infinite dimensional quantum variables are provided to encode classical variables, in practice it is inconvenient (if possible) to specify and reasoning about properties in infinite dimensional Hilbert spaces. Furthermore, the subspace assertions in [Unruh 2019a; Zhou et al. 2019] cannot describe probabilistic satisfiability. Consequently, quantum algorithms which succeed with probability less than 1 cannot be verified in their logic.

**Contribution of the current paper**: Our main contribution is a sound and relatively complete Hoare logic for a simple while-language where both classical and quantum variables are involved. The expressiveness and effectiveness of our logic are demonstrated by formally specifying and verifying Shor's factorisation algorithm [Shor 1994] and its related subroutines such as quantum Fourier transform, phase estimation, and order finding algorithms. To the best of our knowledge, this is the first time quantum Hoare logic is applied on verification of the whole algorithm of Shor's factorisation.

Our work distinguishes itself from the works on quantum Hoare logic mentioned above in the following aspects:

(1) *Programming language.* The language considered in this paper supports both infinite type classical variables and quantum variables. In contrast, the programming language in [Chadha et al.

2006a; Kakutani 2009] allows only a finite variant of type **Integer** (and bounded iteration for [Chadha et al. 2006a]), while only quantum variables are considered in [Barthe et al. 2019; Li and Unruh 2019; Unruh 2019a,b; Ying 2012, 2016, 2019; Ying et al. 2018; Zhou et al. 2019].

(2) *Classical-quantum states.* We define program states of our quantum language to be mappings from classical valuations to partial density operators. This notion of *positive-operator valued distribution* is a direct extension of probability distribution in the probabilistic setting, and often simplifies both description and verification of program correctness, compared with the way adopted in [Chadha et al. 2006a] of regarding probability distributions over pairs of classical valuation and quantum pure state as classical-quantum states. Note also that if only finite type classical variables and qubit type quantum variables are considered, our definition coincides with the one in [Selinger 2004].

(3) *Classical-quantum assertions.* Accordingly, assertions for the classical-quantum program states are defined to be mappings from classical valuations to positive operators, analogous to discrete random variables in the probabilistic case. This follows the expectation-based approach in [Barthe et al. 2019; Li and Unruh 2019; Ying 2012, 2016, 2019; Ying et al. 2018]. However, we also require that the preimage of each positive operator under the mapping be characterised by a classical first-order logic formula. Thus our definition of assertions is essentially in a *hybrid* style, combining the satisfaction-based approach for the classical part and the expectation-based one for the quantum part. Interestingly, this makes the resultant quantum Hoare logic much simpler and similar to the traditional Hoare logic, compared with those in [Chadha et al. 2006a; Kakutani 2009] for classical-quantum languages.

(4) *Auxiliary rules.* In addition to the sound and complete proof system, various auxiliary proof rules are provided to simplify reasoning in real applications. These include the standard disjunction, invariance, and quantifier introduction rules for the classical part of the assertions, and super-operator application and superposition for the quantum part. In particular, the (ProbComp) rule plays an essential role in verification of quantum algorithms which succeed with a certain probability. These rules turn out to be useful, as illustrated by a series of examples including Grover's search algorithm and Shor's factorisation algorithm.

The paper is organised as follows. In the remainder of this section, related work on quantum Hoare logic is further discussed in detail. We review in Sec 2 some basic notions from linear algebra and quantum mechanics that will be used in this paper. Classical-quantum states and assertions, which serve as the basis for the semantics and correctness of quantum programs, are defined in Sec 3. The quantum programming language that we are concerned with is introduced in Sec 4. A structural operational semantics, a denotational semantics, and a weakest (liberal) precondition semantics are also defined there. Sec 5 is devoted to a Hoare logic for quantum programs written in our language, where proof rules for both partial and total correctness are proposed. These proof systems are shown to be both sound and relatively complete with respective to their corresponding correctness semantics. Auxiliary proof rules are presented in Sec 6 to help reasoning in real applications, and Sec 7 concludes the paper and points out some directions for future study. In addition to the running example of Grover's algorithm, verification of quantum Fourier transform, phase estimation, order finding, and Shor's algorithm are provided in the Appendix to illustrate the expressiveness of our language as well as the effectiveness of the proposed Hoare logic. Due to the lack of space, all proofs are put in the Appendix.

### 1.1 Related work

Although the first quantum programming languages traced back to [Bettelli et al. 2003; Ömer 1998; Sanders and Zuliani 2000], Selinger's seminal paper [Selinger 2004] proposed for the first time a

rigorous semantics for a simple quantum language QPL. The syntax of our language is heavily influenced by Selinger's work. We also borrow from him the idea of using partial density operators (i.e., not normalising them at each computational step) to describe quantum states. This convention simplifies both notationally and conceptually the semantics of quantum languages, especially the description of non-termination. Our language excludes general recursion and procedure call from QPL, but includes **Integer** as a classical data type. Consequently, the semantic model in [Selinger 2004], which takes finite tuples (indexed by valuations of **Boolean** variables in the program) of partial density operators as program states, does not apply directly to our language considered in this paper.

An Ensemble Exogenous Quantum Propositional Logic (EEQPL) was proposed in [Chadha et al. 2006a] for a simple quantum language with bounded **Integer** type and bounded iteration. In contrast with Selinger's approach, program states of the language are probability sub-distributions over pairs of classical valuation and quantum pure state. EEQPL has the ability of reasoning about amplitudes of quantum states. This makes it very strong in expressiveness, but also hinders its use in applications such as debugging, as amplitudes of quantum states are not physically accessible through measurements. The soundness and (weak) completeness of EEQPL is proven in a special case where all real and complex values involved range over a finite set. General completeness result has not been reported. A qualitative Hoare logic called QHL for Selinger's QPL (again, without general recursion and procedure call) was proposed in [Kakutani 2009]. The assertion language of QHL is an extended first-order logic with the primitives of applying a matrix on a set of qubits and computing the probability that a classical predicate is satisfied by the outcome of a quantum measurement. The proof system of QHL is sound, but no completeness result was established.

The idea of taking hermitian operators as quantum assertions was first proposed in [D'Hondt and Panangad 2006a], which paves the way for expectation-based reasoning about quantum programs. The notion of quantum weakest precondition was also proposed in the same paper in a language-independent manner. Based on these notions, a sound and relatively complete Hoare logic was proposed in [Ying 2012] for a quantum while language where only quantum variables are involved. However, infinite types such as **Integer** can also be encoded, as the quantum variables can be (countably) infinite dimensional. In contrast, our language explicitly includes classical data types, but only allows **Qudit** (associated with a $d$-dimensional Hilbert space, where $d$ is an arbitrary but finite integer) for quantum variables. The operational semantics of our language, as well as the way the denotational one is derived from it, are inspired by [Ying 2012]. Some auxiliary proof rules presented in Sec 6 are motivated by [Ying 2019].

The quantum Hoare logic in [Ying 2012] has been implemented on Isabelle/HOL [Liu et al. 2019], and a restricted version of it, called applied quantum Hoare logic (aQHL), was proposed in [Zhou et al. 2019] where quantum predicates are restricted to be projections, instead of hermitian operators, with the purpose of simplifying its use in debugging and testing of real quantum programs. It was also generalised in [Hung et al. 2019] for reasoning about robustness of quantum programs against noise during execution, and in [Ying et al. 2018] for analysis of parallel quantum programs. A quantum Hoare logic with ghost variables is introduced in [Unruh 2019a]. Interestingly, by introducing the ghost variables, one can express properties such as a quantum variable is unentangled with others. The logic is shown to be sound, but again, no completeness result is provided.

## 2 PRELIMINARIES

This section is devoted to fixing some notations from linear algebra and quantum mechanics that will be used in this paper. For a thorough introduction of relevant backgrounds, we refer to [Nielsen and Chuang 2002, Chapter 2].

### 2.1 Basic linear algebra

Let $\mathcal{H}$ be a Hilbert space. In the finite-dimensional case which we are concerned with here, it is merely a complex linear space equipped with an inner product. Following the tradition in quantum computing, vectors in $\mathcal{H}$ are denoted in the Dirac form $|\psi\rangle$. The inner product of $|\psi\rangle$ and $|\phi\rangle$ is written $\langle\psi|\phi\rangle$, and they are *orthogonal* if $\langle\psi|\phi\rangle = 0$. The *outer product* of them, denoted $|\psi\rangle\langle\phi|$, is a rank-one linear operator which maps any $|\psi'\rangle$ in $\mathcal{H}$ to $\langle\phi|\psi'\rangle|\psi\rangle$. The *length* of $|\psi\rangle$ is defined to be $\||\psi\rangle\| \triangleq \sqrt{\langle\psi|\psi\rangle}$ and it is called *normalised* if $\||\psi\rangle\| = 1$. A set of vectors $B \triangleq \{|i\rangle : i \in I\}$ in $\mathcal{H}$ is *orthonormal* if each $|i\rangle$ is normalised and every two of them are orthogonal. Furthermore, if they span the whole space $\mathcal{H}$; that is, any vector in $\mathcal{H}$ can be written as a linear combination of vectors in $B$, then $B$ is called an *orthonormal basis* of $\mathcal{H}$.

Let $\mathcal{L}(\mathcal{H})$ be the set of linear operators on $\mathcal{H}$, and $\mathbf{0}_{\mathcal{H}}$ and $I_{\mathcal{H}}$ the zero and identity operators respectively. Let $A \in \mathcal{L}(\mathcal{H})$. The *trace* of $A$ is defined to be $\mathrm{tr}(A) \triangleq \sum_{i\in I}\langle i|A|i\rangle$ for some (or, equivalently, any) orthonormal basis $\{|i\rangle : i \in I\}$ of $\mathcal{H}$. The *adjoint* of $A$, denoted $A^\dagger$ is the unique linear operator in $\mathcal{L}(\mathcal{H})$ such that $\langle\psi|A|\phi\rangle = \langle\phi|A^\dagger|\psi\rangle^*$ for all $|\psi\rangle, |\phi\rangle \in \mathcal{H}$. Here for a complex number $z$, $z^*$ denotes its conjugate. Operator $A$ is said to be *normal* if $A^\dagger A = AA^\dagger$, *hermitian* if $A^\dagger = A$, *unitary* if $A^\dagger A = I_{\mathcal{H}}$, and *positive* if for all $|\psi\rangle \in \mathcal{H}$, $\langle\psi|A|\psi\rangle \geq 0$. Obviously, hermitian operators are normal, and both unitary operators and positive ones are hermitian. Any normal operator $A$ can be written into a *spectral decomposition* form $A = \sum_{i\in I}\lambda_i|i\rangle\langle i|$ where $\{|i\rangle : i \in I\}$ constitute some orthonormal basis of $\mathcal{H}$. Furthermore, if $A$ is hermitian, then all $\lambda_i$ are real; if $A$ is unitary, then all $\lambda_i$ has unit length; if $A$ is positive, then all $\lambda_i$ are nonnegative. The Löwner (partial) order $\sqsubseteq_{\mathcal{H}}$ on the set of hermitian operators on $\mathcal{H}$ is defined by letting $A \sqsubseteq_{\mathcal{H}} B$ iff $B - A$ is positive.

Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be two finite dimensional Hilbert spaces, and $\mathcal{H}_1 \otimes \mathcal{H}_2$ their tensor product. Let $A_i \in \mathcal{L}(\mathcal{H}_i)$. The tensor product of $A_1$ and $A_2$, denoted $A_1 \otimes A_2$ is a linear operator in $\mathcal{L}(\mathcal{H}_1 \otimes \mathcal{H}_2)$ such that $(A_1 \otimes A_2)|(\psi_1\rangle \otimes |\psi_2\rangle) = (A_1|\psi_1\rangle) \otimes (A_2|\psi_2\rangle)$ for all $|\psi_i\rangle \in \mathcal{H}_i$. To simplify notations, we often write $|\psi_1\rangle|\psi_2\rangle$ for $|\psi_1\rangle \otimes |\psi_2\rangle$. Given $\mathcal{H}_1$ and $\mathcal{H}_2$, the *partial trace* with respect to $\mathcal{H}_2$, denoted by $\mathrm{tr}_{\mathcal{H}_2}$ is a linear mapping from $\mathcal{L}(\mathcal{H}_1 \otimes \mathcal{H}_2)$ to $\mathcal{L}(\mathcal{H}_1)$ such that for any $|\psi_i\rangle, |\phi_i\rangle \in \mathcal{H}_i$, $i = 1, 2$,

$$\mathrm{tr}_{\mathcal{H}_2}(|\psi_1\rangle\langle\phi_1| \otimes |\phi_1\rangle\langle\phi_2|) = \langle\phi_2|\phi_1\rangle|\psi_1\rangle\langle\phi_1|.$$

A linear operator $\mathcal{E}$ from $\mathcal{L}(\mathcal{H}_1)$ to $\mathcal{L}(\mathcal{H}_2)$ is called a *super-operator* (from $\mathcal{H}_1$ to $\mathcal{H}_2$). It is said to be (1) *positive* if it maps positive operators to positive operators; (2) *completely positive* if all the cylinder extension $\mathcal{I}_{\mathcal{H}} \otimes \mathcal{E}$ is positive for all finite dimensional Hilbert space $\mathcal{H}$, where $\mathcal{I}_{\mathcal{H}}$ is the identity super-operator on $\mathcal{L}(\mathcal{H})$; (3) *trace-preserving* (resp. *trace-nonincreasing*) if $\mathrm{tr}(\mathcal{E}(A)) = \mathrm{tr}(A)$ (resp. $\mathrm{tr}(\mathcal{E}(A)) \leq \mathrm{tr}(A)$ for any positive operator $A \in \mathcal{L}(\mathcal{H}_1)$; (4) *unital* (resp. *sub-unital*) if $\mathcal{E}(I_{\mathcal{H}_1}) = I_{\mathcal{H}_2}$ (resp. $\mathcal{E}(I_{\mathcal{H}_1}) \sqsubseteq_{\mathcal{H}_2} I_{\mathcal{H}_2}$. From *Kraus representation theorem* [Kraus et al. 1983], a super-operator $\mathcal{E}$ from $\mathcal{L}(\mathcal{H}_1)$ to $\mathcal{L}(\mathcal{H}_2)$ is completely positive iff there is some set of linear operators, called *Kraus operators*, $\{E_i : i \in I\}$ from $\mathcal{H}_1$ to $\mathcal{H}_2$ such that $\mathcal{E}(A) = \sum_{i\in I} E_i A E_i^\dagger$ for all $A \in \mathcal{L}(\mathcal{H}_1)$. It is easy to check that the trace and partial trace operations defined above are both completely positive and trace-preserving super-operators. Given a completely positive super-operator $\mathcal{E}$ from $\mathcal{H}_1$ to $\mathcal{H}_2$ with Kraus operators $\{E_i : i \in I\}$, the adjoint of $\mathcal{E}$, denoted $\mathcal{E}^\dagger$, is a completely positive super-operator from $\mathcal{H}_2$ back to $\mathcal{H}_1$ with Kraus operators $\{E_i^\dagger : i \in I\}$. Then we have $(\mathcal{E}^\dagger)^\dagger = \mathcal{E}$, and $\mathcal{E}$ is trace-preserving (resp. trace-nonincreasing) iff $\mathcal{E}^\dagger$ is unital (resp. sub-unital). Furthermore, for any $A \in \mathcal{L}(\mathcal{H}_1)$ and $B \in \mathcal{L}(\mathcal{H}_2)$, $\mathrm{tr}(\mathcal{E}(A) \cdot B) = \mathrm{tr}(A \cdot \mathcal{E}^\dagger(B))$.

### 2.2 Basic quantum mechanics

According to von Neumann's formalism of quantum mechanics [Von Neumann 1955], any quantum system with finite degrees of freedom is associated with a finite-dimensional Hilbert space

$\mathcal{H}$ called its *state space*. When $\dim(\mathcal{H}) = 2$, we call such a system a *qubit*, the analogy of bit in classical computing. A *pure state* of the system is described by a normalised vector in $\mathcal{H}$. When the system is in one of an ensemble of states $\{|\psi_i\rangle : i \in I\}$ with respective probabilities $p_i$, we say it is in a *mixed* state, represented by the *density operator* $\sum_{i\in I} p_i|\psi_i\rangle\langle\psi_i|$ on $\mathcal{H}$. Obviously, a density operator is positive and has trace 1. Conversely, by spectral decomposition, any such operator corresponds to some (not necessarily unique) mixed state.

The state space of a composite system (for example, a quantum system consisting of multiple qubits) is the tensor product of the state spaces of its components. For a mixed state $\rho$ on $\mathcal{H}_1 \otimes \mathcal{H}_2$, partial traces of $\rho$ have explicit physical meanings: the density operators $\mathrm{tr}_{\mathcal{H}_1}(\rho)$ and $\mathrm{tr}_{\mathcal{H}_2}(\rho)$ are exactly the reduced quantum states of $\rho$ on the second and the first component system, respectively. Note that in general, the state of a composite system cannot be decomposed into tensor product of the reduced states on its component systems. A well-known example is the 2-qubit state $|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. This kind of state is called *entangled state*, and usually is the key to many quantum information processing tasks such as teleportation [Bennett et al. 1993] and superdense coding [Bennett and Wiesner 1992].

The *evolution* of a closed quantum system is described by a unitary operator on its state space: if the states of the system at times $t_1$ and $t_2$ are $\rho_1$ and $\rho_2$, respectively, then $\rho_2 = U\rho_1 U^\dagger$ for some unitary operator $U$ which depends only on $t_1$ and $t_2$. In contrast, the general dynamics which can occur in a physical system is described by a completely positive and trace-preserving super-operator on its state space. Note that the unitary transformation $U(\rho) = U\rho U^\dagger$ is such a super-operator.

A quantum *measurement* $\mathcal{M}$ is described by a collection $\{M_i : i \in J\}$ of linear operators on $\mathcal{H}$, where $J$ is the set of measurement outcomes. It is required that the measurement operators satisfy the completeness equation $\sum_{i\in J} M_i^\dagger M_i = I_{\mathcal{H}}$. If the system is in state $\rho$, then the probability that measurement result $i$ occurs is given by $p_i = \mathrm{tr}(M_i^\dagger M_i \rho)$, and the state of the post-measurement system is $\rho_i = M_i \rho M_i^\dagger / p_i$ whenever $p_i > 0$. Note that the super-operator

$$\mathcal{E}_{\mathcal{M}} : \rho \mapsto \sum_{i\in J} p_i \rho_i = \sum_{i\in J} M_i \rho M_i^\dagger$$

which maps the initial state to the final (mixed) one when the measurement outcome is ignored is completely positive and trace-preserving. A particular case of measurement is *projective measurement* which is usually represented by a hermitian operator $M$ in $\mathcal{L}(\mathcal{H})$ called *observable*. Let

$$M = \sum_{m\in spec(M)} mP_m$$

where $spec(M)$ is the set of eigenvalues of $M$, and $P_m$ the projection onto the eigenspace associated with $m$. Obviously, the projectors $\{P_m : m \in spec(M)\}$ form a quantum measurement.

In this paper, we are especially concerned with the set

$$\mathcal{P}(\mathcal{H}) \triangleq \{M \in \mathcal{L}(\mathcal{H}) : \mathbf{0}_{\mathcal{H}} \sqsubseteq M \sqsubseteq I_{\mathcal{H}}\}$$

of observables whose eigenvalues lie between 0 and 1, where $\sqsubseteq$ is the Löwner order on $\mathcal{L}(\mathcal{H})$. Furthermore, following Selinger's convention [Selinger 2004], we regard the set of *partial density operators*

$$\mathcal{D}(\mathcal{H}) \triangleq \{\rho \in \mathcal{L}(\mathcal{H}) : \mathbf{0}_{\mathcal{H}} \sqsubseteq \rho, \mathrm{tr}(\rho) \leq 1\}$$

as (unnormalised) quantum states. Intuitively, the partial density operator $\rho$ means that the legitimate quantum state $\rho/\mathrm{tr}(\rho)$ is reached with probability $\mathrm{tr}(\rho)$. As a matter of fact, we note that $\mathcal{D}(\mathcal{H}) \subseteq \mathcal{P}(\mathcal{H})$.

| Classical | Probabilistic | Quantum | Classical-quantum |
|---|---|---|---|
| state | probability (sub)distribution | (partial) density operator | cq-state |
| $\sigma \in \Sigma$ | $\mu \in \Sigma \to [0,1]$ | $\rho \in \mathcal{D}(\mathcal{H})$ | $\Delta \in \Sigma \to \mathcal{D}(\mathcal{H})$ |
| | countable support | | countable support |
| assertion | (discrete) random variable | observable | cq-assertion |
| $p \in \Sigma \to \{0,1\}$ | $f \in \Sigma \to [0,1]$ | $M \in \mathcal{P}(\mathcal{H})$ | $\Theta \in \Sigma \to \mathcal{P}(\mathcal{H})$ |
| | countable image | | countable image |
| satisfaction | expectation | expectation | expectation |
| $\sigma \models p$ | $\sum_{\sigma \in \lceil \mu \rceil} \mu(\sigma) f(\sigma)$ | $\mathrm{tr}(M\rho)$ | $\sum_{\sigma \in \lceil \Delta \rceil} \mathrm{tr}\,[\Delta(\sigma)\Theta(\sigma)]$ |

Table 1. Comparison of the basic notions in different language paradigms.

## 3 CLASSICAL-QUANTUM STATES AND ASSERTIONS

In this section, the notions of program states and assertions are introduced for our quantum language where classical variables are involved. To motivate the definition, we first review the corresponding ones in classical (non-probabilistic), probabilistic, and purely quantum programs. A brief summary of the comparison, which extends the one presented in [D'Hondt and Panangaden 2006a], is depicted in Table 1.

Let $\Sigma$ be a non-empty set which serves as the state space of classical programs. An assertion $p$ for classical states is (semantically) a mapping from $\Sigma$ to $\{0,1\}$ such that a state $\sigma$ satisfies $p$, written $\sigma \models p$, iff $p(\sigma) = 1$. In contrast, a state for probabilistic programs is a probability sub-distribution $\mu$ on $\Sigma$ which has countable support[1]; that is, $\mu(\sigma) > 0$ for at most countably infinite many $\sigma \in \Sigma$. Accordingly, an assertion for probabilistic states is a discrete random variable $f$ on $\Sigma$ with countable image; that is, there are at most countably infinite many values for $f$. Finally, the expectation of a state satisfying an assertion corresponds naturally to the expected value of a random variable with respect to a probability distribution. In particular, when the assertion is a traditional one, meaning that its image set is $\{0,1\}$, this expectation reduces to the probability of satisfaction.

To motivate the corresponding notions proposed in [D'Hondt and Panangaden 2006b] for purely quantum programs where classical variables are excluded, note that for any partial density operator $\rho$ in $\mathcal{D}(\mathcal{H})$ and any orthonormal basis $\{|i\rangle : i \in J\}$ of $\mathcal{H}$, the function $\mu$ with $\mu(i) = \langle i|\rho|i\rangle$ defines a probability sub-distribution over $J$. Thus the set $\mathcal{D}(\mathcal{H})$ can naturally be taken as the state space for purely quantum programs. Similarly, for any observable $M \in \mathcal{P}(\mathcal{H})$, $\langle i|M|i\rangle \in [0,1]$ for all $i \in J$. Thus $\mathcal{P}(\mathcal{H})$ can be regarded as the quantum extension of probabilistic assertions. Finally, the expectation of a state $\rho$ satisfying an assertion $M$ is the expected value $\sum_i \mu(i)\langle i|M|i\rangle$, which, when $|i\rangle$'s are eigenstates of $M$, is exactly $\mathrm{tr}(M\rho)$. Most remarkably, as $\mathrm{tr}(M\rho)$ is the expected value of outcomes when the projective measurement represented by $M$ is applied on state $\rho$, it can be physically estimated (instead of mathematically calculated) when multi-copies of $\rho$ is available. This physical implementability is especially important in black box testing of quantum programs, where programs can be executed multiple times, but the implementation detail is not available.

[1]For simplicity, we only consider here probabilistic programs in which all random variables are taken discrete. The probabilities are not required to sum up to 1, for the sake of describing non-termination.

For programs where both quantum and classical variables are involved, we have to find a way to combine the notions for probabilistic programs and purely quantum ones. The following three subsections are devoted to this goal.

### 3.1 Classical-quantum states

We assume two basic types for classical variables: **Boolean** with the corresponding domain $D_{\mathbf{Boolean}} \triangleq \{\mathbf{true}, \mathbf{false}\}$ and **Integer** with $D_{\mathbf{Integer}} \triangleq \mathbb{Z}$. For each integer $d \geq 2$, we assume a basic quantum type **Qudit** with domain $\mathcal{H}_{\mathbf{Qudit}}$, which is a $d$-dimensional Hilbert space with an orthonormal basis $\{|0\rangle, \ldots, |d-1\rangle\}$. In particular, we denote the quantum type for $d = 2$ as **Qubit**. Let *Var*, ranged over by $x, y, \cdots$, and *qVar*, ranged over by $q, r, \cdots$, be countably infinite sets of classical and quantum variables, respectively. Let $\Sigma \triangleq Var \to D$ be the (uncountably infinite) set of classical states, where $D \triangleq D_{\mathbf{Boolean}} \cup D_{\mathbf{Integer}}$. We further require that states in $\Sigma$ respect the types of classical variables; that is, $\sigma(x) \in D_{type(x)}$ for all $\sigma \in \Sigma$ and $x \in Var$. For any finite subset $V$ of *qVar*, let

$$\mathcal{H}_V \triangleq \bigotimes_{q \in V} \mathcal{H}_q,$$

where $\mathcal{H}_q \triangleq \mathcal{H}_{type(q)}$ is the Hilbert space associated with $q$. As we use subscripts to distinguish Hilbert spaces with different (sets of) quantum variables, their order in the tensor product is not essential. In this paper, when we refer to a subset of *qVar*, it is always assumed to be finite.

*Definition 3.1.* Given $V \subseteq qVar$, a classical-quantum state (cq-state for short) $\Delta$ over $V$ is a function in $\Sigma \to \mathcal{D}(\mathcal{H}_V)$ such that

(1) the support of $\Delta$, denoted $\lceil \Delta \rceil$, is countable. That is, $\Delta(\sigma) \neq \mathbf{0}_{\mathcal{H}_V}$ for at most countably infinite many $\sigma \in \Sigma$;
(2) $\mathrm{tr}(\Delta) \triangleq \sum_{\sigma \in \lceil \Delta \rceil} \mathrm{tr}[\Delta(\sigma)] \leq 1$.

One may notice the similarity of the above definition with probability sub-distributions. Actually, a probability sub-distribution is obtained by taking the trace of each value of a cq-state. Note also that this definition, when restricted to finite type classical variables, coincides with the 'tuple of matrices' representation introduced in [Selinger 2004].

*Example 3.2.* To better understand the notion of cq-states, let us consider the output of quantum Teleportation algorithm [Bennett et al. 1993], where Alice would like to teleport an arbitrary state $\rho_q$ to Bob, using a pre-shared Bell state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)_{q_1, q_2}$ between them. Here $q$, $q_1$, and $q_2$ are all **Qubit**-type variables, and we use subscripts to indicate the quantum variables on which the states and operators are acting. Suppose $\sigma_i$ and $\rho^i$, $0 \leq i \leq 3$, are the classical and final quantum states (of $q_2$), respectively, when the measurement outcome of Alice is $i$. Then the cq-state output by the algorithm is a function which maps $\sigma_i$ to $\frac{1}{4}|i\rangle_{q,q_1}\langle i| \otimes \rho^i_{q_2}$, $0 \leq i \leq 3$, and other classical states to $\mathbf{0}$.

Note that another, more intuitive at the first glance, way to describe the same cq-state is to use a probability distribution of classical-quantum state pairs: $\{\frac{1}{4}(\sigma_i, |i\rangle_{q,q_1}\langle i| \otimes \rho^i_{q_2}) : 0 \leq i \leq 3\}$. We will explain why we do not choose to do so in more detail at the end of this subsection after more notations are introduced.

Sometimes it is convenient to denote a cq-state $\Delta$ by the explicit form $\bigoplus_{i \in I} \langle \sigma_i, \rho_i \rangle$ where $\lceil \Delta \rceil = \{\sigma_i : i \in I\}$ and $\Delta(\sigma_i) = \rho_i$ for each $i \in I$. When $\Delta$ is a simple function such that $\lceil \Delta \rceil = \{\sigma\}$ for some $\sigma$ and $\Delta(\sigma) = \rho$, we denote $\Delta$ simply by $\langle \sigma, \rho \rangle$. Let $\mathcal{S}_V$ be the set of all cq-states over $V$, and $\mathcal{S}$ the set of all cq-states; that is,

$$\mathcal{S} \triangleq \bigcup_{V \subseteq qVar} \mathcal{S}_V.$$

When $\Delta \in \mathcal{S}_V$, let $qv(\Delta) \triangleq V$ denote the set of quantum variables in $\Delta$. We extend the Löwner order $\sqsubseteq_V$ for $\mathcal{H}_V$ point-wisely to $\mathcal{S}$ by letting $\Delta \sqsubseteq \Delta'$ iff $qv(\Delta) = qv(\Delta')$ and for all $\sigma \in \Sigma$, $\Delta(\sigma) \sqsubseteq_{qv(\Delta)} \Delta'(\sigma)$. The following lemma shows that $\mathcal{S}$ is an $\omega$-complete partial order set (CPO) under $\sqsubseteq$.

Lemma 3.3. *For any $V \subseteq qVar$, $\mathcal{S}_V$ is a pointed $\omega$-CPO under $\sqsubseteq$, with the least element being the constant $\mathbf{0}_{\mathcal{H}_V}$ function, denoted $\bot_V$. Furthermore, $\mathcal{S}$ as a whole is an $\omega$-CPO under $\sqsubseteq$.*

When $\Delta \sqsubseteq \Delta'$, there exists a unique $\Delta'' \in \mathcal{S}_{qv(\Delta)}$, denoted $\Delta' - \Delta$, such that $\Delta'' + \Delta = \Delta'$. Let $\mathcal{E}$ be a completely positive and trace-nonincreasing linear map from $\mathcal{H}_V$ to $\mathcal{H}_W$. We extend it to $\mathcal{S}_V$ in a point-wise way: $\mathcal{E}(\Delta)(\sigma) = \mathcal{E}(\Delta(\sigma))$ for all $\sigma$. Note that $\mathcal{E}(\mathbf{0}_{\mathcal{H}_V}) = \mathbf{0}_{\mathcal{H}_W}$ and $\mathrm{tr}(\mathcal{E}(\rho)) \leq \mathrm{tr}(\rho)$ for all $\rho \in \mathcal{D}(\mathcal{H}_V)$. Thus $\mathcal{E}(\Delta)$ is a valid cq-state provided that $\Delta$ is. In particular, for any $V \subseteq qv(\Delta)$, the partial trace $\mathrm{tr}_{\mathcal{H}_V}(\Delta)$ is a cq-state which maps any $\sigma \in \Sigma$ to $\mathrm{tr}_{\mathcal{H}_V}(\Delta(\sigma))$. Furthermore, for any $\rho \in \mathcal{D}(\mathcal{H}_W)$ with $W \cap qv(\Delta) = \emptyset$ and $\mathrm{tr}(\rho) \leq 1$, $\Delta \otimes \rho$ is a cq-state in $\mathcal{S}_{qv(\Delta) \cup W}$ which maps $\sigma$ to $\Delta(\sigma) \otimes \rho$. In the special case that $W = \emptyset$, $\rho$ becomes a real number in $[0, 1]$, and we write $\rho\Delta$ for $\Delta \otimes \rho$.

Let $\{\Delta_i : i \in I\}$ be a countable set of cq-states. The summation of them, denoted $\sum_{i \in I} \Delta_i$, is a function $\Delta$ such that for any $\sigma \in \Sigma$, $\Delta(\sigma) = \sum_{i \in I} \Delta_i(\sigma)$, whenever $\Delta$ is a valid cq-state. Obviously, $\lceil \Delta \rceil = \bigcup_{i \in I} \lceil \Delta_i \rceil$. It is worth noting the difference between $\sum_{i \in I} \langle \sigma_i, \rho_i \rangle$, the summation of some (simple) cq-states, and $\bigoplus_{i \in I} \langle \sigma_i, \rho_i \rangle$, the explicit form of a single one: in the latter $\sigma_i$'s must be distinct while in the former they may not. Furthermore, for any real numbers $\lambda_i$, $i \in I$, if both $\Delta_+ \triangleq \sum_{\lambda_i > 0} \lambda_i \Delta_i$ and $\Delta_- \triangleq \sum_{\lambda_i < 0} (-\lambda_i) \Delta_i$ are well-defined, then the linear-sum $\sum_{i \in I} \lambda_i \Delta_i$ is defined to be $\Delta_+ - \Delta_-$ as long as it is a valid cq-state as well. In the rest of this paper, whenever we write $\sum_{i \in I} \lambda_i \Delta_i$ we always assume that it is well-defined.

*Example 3.4.* Let $\sigma_1 \neq \sigma_2 \in \Sigma$ and $q \in qVar$ with $type(q) = \mathbf{Qubit}$. Then

$$\Delta \triangleq \langle \sigma_1, 0.5|0\rangle_q\langle 0| \rangle \oplus \langle \sigma_2, 0.25 I_q \rangle$$

is a cq-state over $\{q\}$ which lies in $\langle \sigma_1, |0\rangle_q\langle 0| \rangle$ and $\langle \sigma_2, I_q/2 \rangle$, both legitimate classical-quantum state pairs, with probability $1/2$ respectively. Now let $\Delta' \triangleq \langle \sigma_2, |+\rangle_q\langle +| \rangle$. Then the linear-sum

$$\Delta - 0.25\Delta' = \langle \sigma_1, 0.5|0\rangle_q\langle 0| \rangle \oplus \langle \sigma_2, 0.25|-\rangle_q\langle -| \rangle.$$

As we can see from the above example, a partial density operator $\rho$ encodes both the (normalised) quantum state $\rho/\mathrm{tr}(\rho)$ and the probability $\mathrm{tr}(\rho)$ of reaching it. Thus the meaning of a cq-state $\Delta = \bigoplus_{i \in I} \langle \sigma_i, \rho_i \rangle$ is that with probability $\mathrm{tr}(\rho_i)$, the classical and quantum systems are in states $\sigma_i$ and $\rho_i/\mathrm{tr}(\rho_i)$, respectively. This also explains why we have the requirement in Definition 3.1(2): the probabilities of all possible state pairs sum up to at most 1. One may ask why we do not directly define cq-states as sub-distributions over such classical-quantum state pairs, just as in [Chadha et al. 2006a]? To see the reason, note that $\mathcal{D}(\mathcal{H}_V)$ is a convex set, and the quantum state $\sum_i \lambda_i \rho_i$ is indistinguishable from the assemble that lies in $\rho_i$ with probability $\lambda_i \geq 0$, $\sum_i \lambda_i = 1$. Thus we would have to introduce some auxiliary rules to equate the probability distribution $\sum_i \lambda_i \langle \sigma, \rho_i \rangle$ with the single state $\langle \sigma, \sum_i \lambda_i \rho_i \rangle$, if cq-states had been defined as sub-distributions on classical-quantum state pairs. In contrast, in our framework these two cq-states are *equal by definition*, from the liner-sum form introduced above. Finally, note that this difficulty does not appear in probabilistic programs, as the classical state space $\Sigma$ is discrete, and there does not exist any algebraic structure in it.

### 3.2 Classical-quantum assertions

Recall that assertions for classical program states are usually represented as first order logic formulas over *Var*. For any classical assertion $p$, denote by $[\![p]\!] \triangleq \{\sigma \in \Sigma : \sigma \models p\}$ the set of classical states that satisfy $p$. Two assertions $p$ and $p'$ are equivalent, written $p \equiv p'$, iff $[\![p]\!] = [\![p']\!]$.

*Definition 3.5.* Given $V \subseteq qVar$, a classical-quantum assertion (cq-assertion for short) $\Theta$ over $V$ is a function in $\Sigma \to \mathcal{P}(\mathcal{H}_V)$ such that

(1) the image set $\Theta(\Sigma)$ of $\Theta$ is countable;
(2) for each $M \in \Theta(\Sigma)$, the preimage $\Theta^{-1}(M)$ is definable by a classical assertion $p$ in the sense that $[\![p]\!] = \Theta^{-1}(M)$.

Obviously, the above definition is a natural extension of discrete random variables when $\mathcal{P}(\mathcal{H}_V)$, the set of operators between $\mathbf{0}_{\mathcal{H}_V}$ and $I_{\mathcal{H}_V}$ with respect to the Löwner order, is regarded as the quantum generalisation of $[0, 1]$. The second clause is introduced to guarantee a compact representation of cq-assertions. It is easy to see that $p_i \wedge p_j \equiv \mathbf{false}$ whenever $i \neq j$.

For convenience, we do not distinguish $p$ and $[\![p]\!]$ when denoting a cq-assertion. Consequently, we write $\bigoplus_{i\in I}\langle p_i, M_i\rangle$ instead of $\bigoplus_{i\in I}\langle [\![p_i]\!], M_i\rangle$ for a cq-assertion $\Theta$ whenever $\Theta(\Sigma) = \{M_i : i \in I\}$ and $\Theta^{-1}(M_i) = [\![p_i]\!]$ for each $i \in I$. Note that this representation is not unique: the representative assertion $p_i$ can be replaced by $p_i'$ whenever $p_i \equiv p_i'$. Furthermore, the summand with zero operator $\mathbf{0}_{\mathcal{H}_V}$ is always omitted. In particular, when $\Theta(\Sigma) = \{\mathbf{0}_{\mathcal{H}}, M\}$ or $\{M\}$ for some $M \neq \mathbf{0}_{\mathcal{H}_V}$, we simply denote $\Theta$ by $\langle p, M\rangle$ for some $p$ with $\Theta^{-1}(M) = [\![p]\!]$.

Note that any observable $M$ in $\mathcal{P}(\mathcal{H})$ corresponds to some *quantitative property* of quantum states. Thus intuitively, a cq-assertion $\bigoplus_{i\in I}\langle p_i, M_i\rangle$ specifies that whenever the classical state satisfies $p_i$, the property $M_i$ is checked on the corresponding quantum state. The average value of the satisfiability will be defined in the next subsection.

*Example 3.6.* Back to the Teleportation algorithm in Example 3.2. The cq-assertion

$$\Theta_1 \triangleq \bigoplus_{0\le i\le 3} \langle x = i, |i\rangle_{q,q_1}\langle i|\rangle$$

where $x$ is the classical variable used by Alice to store (and send to Bob) the measurement outcome, claims that the states of $q$ and $q_1$ are both in the computational basis, and they together correspond to the measurement outcome of Alice. To be specific, it states that whenever $x = i$, the corresponding quantum state of $q$ and $q_1$ should be $|x_0\rangle_q|x_1\rangle_{q_1}$ where $x_0x_1$ is the binary representation of $x$. We will make it more rigorous in Example 3.8.

Suppose the teleported state $\rho = |\psi\rangle\langle\psi|$ is a pure one. Then the cq-assertion

$$\Theta_2 \triangleq \langle \mathbf{true}, |\psi\rangle_{q_2}\langle\psi|\rangle,$$

when applied on the output cq-state, actually computes the (average) precision of the Teleportation algorithm when $|\psi\rangle$ is taken as the input. Again, we refer to Example 3.8 for more details.

Let $\mathcal{A}_V$ be the set of all cq-assertions over $V$, and $\mathcal{A}$ the set of all cq-assertions; that is,

$$\mathcal{A} \triangleq \bigcup_{V\subseteq qVar} \mathcal{A}_V.$$

When $\Theta \in \mathcal{A}_V$, let $qv(\Theta) \triangleq V$ denote the set of quantum variables in $\Theta$. Again, we extend the Löwner order $\sqsubseteq$ point-wisely to $\mathcal{A}$ and it is easy to see that $\mathcal{A}_V$ is also a pointed $\omega$-CPO under $\sqsubseteq$, with the least element being $\bot_V$. Furthermore, it has the largest element $\top_V \triangleq \langle \mathbf{true}, I_{\mathcal{H}_V}\rangle$. When $\Theta \sqsubseteq \Theta'$, we denote by $\Theta' - \Theta$ the unique $\Theta'' \in \mathcal{A}_{qv(\Theta)}$ such that $\Theta'' + \Theta = \Theta'$. With these notions, summation and linear-sum of cq-assertions can be defined similarly as for cq-states.

Given a classical assertion $p$, we denote by $p \bowtie \sum_i \langle p_i, M_i\rangle$ the cq-assertion $\sum_i \langle p \bowtie p_i, M_i\rangle$ (if it is valid) where $\bowtie$ can be any logic connective such as $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, etc. As $[\![p \bowtie p_1]\!] = [\![p \bowtie p_2]\!]$ provided that $[\![p_1]\!] = [\![p_2]\!]$, these notations are well-defined. Let $\mathcal{F}$ be a completely positive and sub-unital linear map from $\mathcal{P}(\mathcal{H}_V)$ to $\mathcal{P}(\mathcal{H}_W)$. We extend it to $\mathcal{A}_V$ in a point-wise way. Note that $\mathcal{F}(\mathbf{0}_{\mathcal{H}_V}) = \mathbf{0}_{\mathcal{H}_W}$ and $\mathcal{F}(M) \sqsubseteq \mathcal{F}(I_{\mathcal{H}_V}) \sqsubseteq I_{\mathcal{H}_W}$ for all $M \in \mathcal{P}(\mathcal{H}_V)$. Thus $\mathcal{F}(\Theta)$ is a valid cq-assertion provided that $\Theta$ is. In particular, when $qv(\Theta) \cap W = \emptyset$, $\Theta \otimes I_{\mathcal{H}_W}$ is a cq-assertion which maps any $\sigma \in \Sigma$ to $\Theta(\sigma) \otimes I_{\mathcal{H}_W}$. Note that 1 is the identity operator on $I_{\mathcal{H}_\emptyset}$. Sometimes we also abuse the notation a bit to write $p$ for $\langle p, 1\rangle$ where $p$ is a classical assertion, and $\lambda$ for $\langle \mathbf{true}, \lambda\rangle$ where $\lambda \in [0, 1]$.

The (lifted) Löwner order provides a natural way to compare cq-assertions over the same set of quantum variables. However, in latter discussion of this paper, we sometimes need to compare cq-assertions acting on different quantum variables. To deal with this situation, we introduce a pre-order $\lesssim$ on the whole set $\mathcal{A}$ of cq-assertions. To be specific, let $V_1, V_2$ be two subsets of $qVar$, and $\Theta_i \in \mathcal{A}_{V_i}$, $i = 1, 2$. We say $\Theta_1 \lesssim \Theta_2$ whenever $\Theta_1 \otimes I_{\mathcal{H}_{V_2 \setminus V_1}} \sqsubseteq I_{\mathcal{H}_{V_1 \setminus V_2}} \otimes \Theta_2$. Obviously, when restricted on some given set of quantum variables, $\lesssim$ coincides with $\sqsubseteq$. Let $\eqsim$ be the kernel of $\lesssim$. Then $\Theta_1 \eqsim \Theta_2$ iff there exists $\Theta$ such that $\Theta_1 = \Theta \otimes I_V$ and $\Theta_2 = \Theta \otimes I_W$ for some $V$ and $W$.

### 3.3 Expectation of satisfaction

With the above notions, we are now ready to define the expectation of a cq-state satisfying a cq-assertion.

*Definition 3.7.* Given a cq-state $\Delta$ and a cq-assertion $\Theta$ with $qv(\Delta) \supseteq qv(\Theta)$, the expectation of $\Delta$ satisfying $\Theta$ is defined to be

$$\mathrm{Exp}(\Delta \models \Theta) \triangleq \sum_{\sigma \in \lceil \Delta \rceil} \mathrm{tr}\left[\Theta(\sigma) \otimes I_{\mathcal{H}_V} \cdot \Delta(\sigma)\right] = \sum_{\sigma \in \lceil \Delta \rceil} \mathrm{tr}\left[\Theta(\sigma) \cdot \mathrm{tr}_{\mathcal{H}_V}(\Delta(\sigma))\right]$$

where $V = qv(\Delta) \backslash qv(\Theta)$.

*Example 3.8.* Consider again the Teleportation algorithm in Example 3.2. Let

$$\Delta \triangleq \bigoplus_{0 \le i \le 3} \langle \sigma_i, \frac{1}{4} |i\rangle_{q,q_1}\langle i| \otimes \rho^i_{q_2}\rangle$$

be the output cq-state of the algorithm when $\rho$ is taken as input. Let $\Theta_1$ and $\Theta_2$ be as defined in Example 3.6. Note that $\sigma_i \models (x = j)$ iff $i = j$. Thus

$$\mathrm{Exp}(\Delta \models \Theta_1) = \sum_{i=0}^{3} \frac{1}{4} \mathrm{tr}(|i\rangle_{q,q_1}\langle i| \otimes I_{q_2} \cdot |i\rangle_{q,q_1}\langle i| \otimes \rho^i_{q_2}) = 1,$$

meaning that with probability 1, the value of $x$ equals the value encoded by states of $q$ and $q_1$.

For $\Theta_2$, we compute

$$\mathrm{Exp}(\Delta \models \Theta_2) = \sum_{i=0}^{3} \frac{1}{4} \mathrm{tr}(I_{q,q_1} \otimes |\psi\rangle_{q_2}\langle\psi| \cdot |i\rangle_{q,q_1}\langle i| \otimes \rho^i_{q_2}) = \frac{1}{4} \sum_{i=0}^{3} \langle\psi|\rho^i|\psi\rangle,$$

which denotes the expected value of the distance between the output states $\rho^i$ and the ideal one $|\psi\rangle$.

We collect some properties of the Exp function in the following lemmas.

Lemma 3.9. *For any cq-state $\Delta \in \mathcal{S}_V$, cq-assertion $\Theta \in \mathcal{A}_W$, $W \subseteq V$, and classical assertion $p$,*

(1) $\mathrm{Exp}(\Delta \models \Theta) \in [0, 1]$;
(2) $\mathrm{Exp}(\bot_V \models \Theta) = \mathrm{Exp}(\Delta \models \bot_W) = 0$, $\mathrm{Exp}(\Delta \models \top_W) = \mathrm{tr}(\Delta)$;

(3) $\mathrm{Exp}(\Delta \models \Theta) = \sum_i \lambda_i \mathrm{Exp}(\Delta \models \Theta_i)$ *if* $\Theta = \sum_i \lambda_i \Theta_i$;
(4) $\mathrm{Exp}(\Delta \models \Theta) = \sum_i \lambda_i \mathrm{Exp}(\Delta_i \models \Theta)$ *if* $\Delta = \sum_i \lambda_i \Delta_i$;
(5) $\mathrm{Exp}(\Delta|_p \models \Theta) = \mathrm{Exp}(\Delta \models p \wedge \Theta)$ *where* $\Delta|_p$ *is the cq-state by restricting* $\Delta$ *on the set of classical states* $\sigma$ *with* $\sigma \models p$.

Lemma 3.10. (1) *For any cq-states* $\Delta$ *and* $\Delta'$ *in* $\mathcal{S}_V$,
- *if* $\Delta \sqsubseteq \Delta'$, *then* $\mathrm{Exp}(\Delta \models \Theta) \leq \mathrm{Exp}(\Delta' \models \Theta)$ *for all* $\Theta \in \mathcal{A}_W$ *with* $W \subseteq V$;
- *conversely, if* $\mathrm{Exp}(\Delta \models \Theta) \leq \mathrm{Exp}(\Delta' \models \Theta)$ *for all* $\Theta \in \mathcal{A}_V$, *then* $\Delta \sqsubseteq \Delta'$.

(2) *For any cq-assertions* $\Theta$ *and* $\Theta'$ *with* $W = qv(\Theta) \cup qv(\Theta')$,
- *if* $\Theta \lesssim \Theta'$, *then* $\mathrm{Exp}(\Delta \models \Theta) \leq \mathrm{Exp}(\Delta \models \Theta')$ *for all* $\Delta \in \mathcal{S}_V$ *with* $W \subseteq V$;
- *conversely, if* $\mathrm{Exp}(\Delta \models \Theta) \leq \mathrm{Exp}(\Delta \models \Theta')$ *for all* $\Delta \in \mathcal{S}_V$, *then* $\Theta \lesssim \Theta'$.

Lemma 3.11. *For any cq-states* $\Delta, \Delta_n \in \mathcal{S}_V$ *and cq-assertions* $\Theta, \Theta_n \in \mathcal{A}_W$ *with* $W \subseteq V$, $n = 1, 2, \cdots$,

(1) $\mathrm{Exp}(\bigvee_{n\geq 0} \Delta_n \models \Theta) = \sup_{n\geq 0} \mathrm{Exp}(\Delta_n \models \Theta)$ *for increasing sequence* $\{\Delta_n\}_n$;
(2) $\mathrm{Exp}(\bigwedge_{n\geq 0} \Delta_n \models \Theta) = \inf_{n\geq 0} \mathrm{Exp}(\Delta_n \models \Theta)$ *for decreasing sequence* $\{\Delta_n\}_n$;
(3) $\mathrm{Exp}(\Delta \models \bigvee_{n\geq 0} \Theta_n) = \sup_{n\geq 0} \mathrm{Exp}(\Delta \models \Theta_n)$ *for increasing sequence* $\{\Theta_n\}_n$;
(4) $\mathrm{Exp}(\Delta \models \bigwedge_{n\geq 0} \Theta_n) = \inf_{n\geq 0} \mathrm{Exp}(\Delta \models \Theta_n)$ *for decreasing sequence* $\{\Theta_n\}_n$.

### 3.4 Substitution and state update

Let $e$ and $e'$ be classical expressions, and $x$ a classical variable with the same type of $e'$. Denote by $e[e'/x]$ the expression obtained by substituting $x$ in $e$ with $e'$. Such substitution can be extended to cq-assertions as follows. Given $\Theta \triangleq \bigoplus_{i\in I} \langle p_i, M_i \rangle$, we define

$$\Theta[e/x] \triangleq \bigoplus_{i\in I} \langle p_i[e/x], M_i \rangle.$$

The well-definedness comes from the following two observations: (1) whenever $[\![p_i]\!] \cap [\![p_j]\!] = \emptyset$, it holds $[\![p_i[e/x]]\!] \cap [\![p_j[e/x]]\!] = \emptyset$; (2) whenever $p \equiv p'$, it holds $p[e/x] \equiv p'[e/x]$. Thus the substitution is independent of the choice of the representative classical assertions $p_i$ in $\Theta$.

For classical state $\sigma$ and $d \in D_{type(x)}$, denote by $\sigma[d/x]$ the updated state which maps $x$ to $d$, and other classical variables $y$ to $\sigma(y)$. Similarly, this updating can be extended to cq-states by defining

$$\Delta[e/x] \triangleq \sum_{i\in I} \langle \sigma_i[\sigma_i(e)/x], \rho_i \rangle$$

whenever $\Delta = \bigoplus_{i\in I} \langle \sigma_i, \rho_i \rangle$. Since substitution does change the trace of the whole state, $\Delta[e/x]$ is still a valid cq-state. Note that unlike for cq-assertions, $\sigma_i[\sigma_i(e)/x]$ and $\sigma_j[\sigma_j(e)/x]$ can be equal even when $\sigma_i \neq \sigma_j$. Thus we have $\sum$ instead of $\bigoplus$ here.

The next lemma shows the relation between substitutions for cq-states and cq-assertions.

Lemma 3.12. *For any cq-state* $\Delta$ *and cq-assertion* $\Theta$ *with* $qv(\Delta) \supseteq qv(\Theta)$, $x \in Var$, *and classical expression* $e$ *with the same type of* $x$,

$$\mathrm{Exp}(\Delta \models \Theta[e/x]) = \mathrm{Exp}(\Delta[e/x] \models \Theta).$$

## 4 A SIMPLE CLASSICAL-QUANTUM LANGUAGE

This section is devoted to the syntax and various semantics of our core programming language which supports deterministic and probabilistic assignments, quantum measurements, quantum operations, conditionals, and while loops.

### 4.1 Syntax

Our classical-quantum language is based on the one proposed in [Selinger 2004], extended with **Integer** type classical variables and probabilistic assignments, but excluding general recursion and procedure call. The syntax is defined as follows:

$$S ::= \textbf{skip} \mid x := e \mid x :=_{\$} g \mid x := \textbf{meas}\, \mathcal{M}[\bar{q}] \mid q := 0 \mid \bar{q} *= U \mid S_0; S_1 \mid \\ \textbf{if } b \textbf{ then } S_1 \textbf{ else } S_0 \textbf{ end} \mid \textbf{while } b \textbf{ do } S \textbf{ end}$$

where $S, S_0$ and $S_1$ denote classical-quantum programs (cq-programs for short), $x$ a classical variable in $Var$, $e$ a classical expression, $g$ a discrete probability distribution over $D_{type(x)}$, $b$ a **Boolean**-type expression, $q$ is a quantum variable and $\bar{q} = q_1, \ldots, q_n$ a (ordered) tuple of distinct quantum variables in $qVar$, $\mathcal{M}$ a measurement and $U$ a unitary operator on $d_{\bar{q}}$-dimensional Hilbert space where

$$d_{\bar{q}} \triangleq \dim(\mathcal{H}_{\bar{q}}) = \prod_{i=1}^{n} \dim(\mathcal{H}_{q_i}).$$

Sometimes we also use $\bar{q}$ to denote the (unordered) set $\{q_1, q_2, \ldots, q_n\}$. Let $|\bar{q}| \triangleq n$ be the size of $\bar{q}$.

Let $Prog$ be the set of all cq-programs. For any $S \in Prog$, the quantum variables that appear in $S$ is denoted $qv(S)$. The set $change(S)$ (resp. $var(S)$) of classical variables that appear in (resp. can be changed by) $S$ are defined in the standard way. Note that the only way to retrieve information from a quantum system is to measure it, a process which may change its state. Thus the notion of read-only quantum variables does not exist in cq-programs.

In the purely quantum language presented in [Ying 2012], conditional branching and while loop are achieved by program constructs like **meas** $\mathcal{M}[\bar{q}]$ **then** $S_1$ **else** $S_0$ **end** and **while** $\mathcal{M}[\bar{q}]$ **do** $S$ **end** where the outcome set of $\mathcal{M}$ is $\{0, 1\}$. Intuitively, the quantum variables in $\bar{q}$ are first measured according to $\mathcal{M}$, and different subsequent programs will be executed depending on the measurement outcome. These constructs can be written in our language in the following equivalent form:

$$x := \textbf{meas}\, \mathcal{M}[\bar{q}];\ \textbf{if } x = 1 \textbf{ then } S_1 \textbf{ else } S_0 \textbf{ end}$$

and

$$x := \textbf{meas}\, \mathcal{M}[\bar{q}]; \textbf{while } x = 1 \textbf{ do } S; x := \textbf{meas}\, \mathcal{M}[\bar{q}]; \textbf{end} \tag{1}$$

respectively. An advantage of having classical variables explicitly in the language is that we can avoid introducing infinite-dimensional Hilbert spaces to describe quantum systems which encode classical data with infinite types such as **Integer**. This will simplify the verification of real-world quantum programs.

To conclude this subsection, we introduce some syntactic sugars for our language which make it easy to use in describing quantum algorithms. Let $|\bar{q}| = n$.

- *Everywhere abort program.* Let **abort** be a program which nowhere converges. This is achieved by defining, say, **abort** $\triangleq$ **while true do skip end**.
- *Initialisation of multiple quantum variables.* Let $\bar{q} := 0$ stand for $q_1 := 0; \cdots ; q_n := 0$.
- *Measurement according to the computational basis.* We write $x :=$ **meas** $\bar{q}$ for $x :=$ **meas** $\mathcal{M}_{com}[\bar{q}]$ where $\mathcal{M}_{com} \triangleq \{P_k \triangleq |k\rangle\langle k| : 0 \le k < d_{\bar{q}}\}$ is the projective measurement according to the computational basis of $\mathcal{H}_{\bar{q}}$. We always write $|k\rangle$ for the product state $|k_1\rangle \cdots |k_n\rangle$, where $k = \sum_{i=1}^{n} k_i d_{i+1} \ldots d_n$.
- *Application of parametrised unitary operations.* Let $U(i)$, $1 \le i \le K$, be a finite family of unitary operators on $d_{\bar{q}}$-dimensional Hilbert spaces, and $e$ an **Integer**-typed expression. We write $\bar{q} *= U(e)$ for

$$\textbf{if } e < 1 \vee e > K \textbf{ then abort else } S_1; S_2; \cdots ; S_K \textbf{ end}$$

$$\langle \mathbf{skip}, \sigma, \rho\rangle \to \langle E, \sigma, \rho\rangle \qquad \langle x := e, \sigma, \rho\rangle \to \langle E, \sigma[\sigma(e)/x], \rho\rangle$$

$$\langle q := 0, \sigma, \rho\rangle \to \langle E, \sigma, \sum_{i=0}^{d_q-1} |0\rangle_q\langle i|\rho|i\rangle_q\langle 0|\rangle \qquad \langle \bar{q} *= U(e), \sigma, \rho\rangle \to \langle E, \sigma, U_{\bar{q}}\rho U_{\bar{q}}^\dagger\rangle$$

$$\frac{d \in D_{type(x)}}{\langle x :=_\$ g, \sigma, \rho\rangle \to \langle E, \sigma[d/x], g(d)\cdot\rho\rangle} \qquad \frac{\mathcal{M} = \{M_i : i \in J\}}{\langle x := \mathbf{meas}\ \mathcal{M}[\bar{q}], \sigma, \rho\rangle \to \langle E, \sigma[i/x], M_i\rho M_i^\dagger\rangle}$$

$$\frac{\langle S_0, \sigma, \rho\rangle \to \langle S', \sigma', \rho'\rangle}{\langle S_0; S_1, \sigma, \rho\rangle \to \langle S'; S_1, \sigma', \rho'\rangle} \text{ where } E; S_1 \equiv S_1$$

$$\frac{\sigma \models b}{\langle \mathbf{if}\ b\ \mathbf{then}\ S_1\ \mathbf{else}\ S_0\ \mathbf{end}, \sigma, \rho\rangle \to \langle S_1, \sigma, \rho\rangle} \qquad \frac{\sigma \models \neg b}{\langle \mathbf{if}\ b\ \mathbf{then}\ S_1\ \mathbf{else}\ S_0\ \mathbf{end}, \sigma, \rho\rangle \to \langle S_0, \sigma, \rho\rangle}$$

$$\frac{\sigma \models \neg b}{\langle \mathbf{while}\ b\ \mathbf{do}\ S\ \mathbf{end}, \sigma, \rho\rangle \to \langle E, \sigma, \rho\rangle} \qquad \frac{\sigma \models b}{\langle \mathbf{while}\ b\ \mathbf{do}\ S\ \mathbf{end}, \sigma, \rho\rangle \to \langle S; \mathbf{while}\ b\ \mathbf{do}\ S\ \mathbf{end}, \sigma, \rho\rangle}$$

Table 2. Operational semantics for cq-programs.

where for each $1 \le i \le K$,

$$S_i \triangleq \mathbf{if}\ e = i\ \mathbf{then}\ \bar{q} *= U(i)\ \mathbf{else\ skip\ end}.$$

- *Application of parametrised quantum variables.* Let $1 \le k \le n$,

$$R \triangleq \{(i_1, \cdots, i_k) : \forall j. 1 \le i_j \le n \text{ and } i_j\text{'s are distinct}\},$$

and $e_i$'s be **Integer**-type expressions. Then $\bar{q}[e_1, \cdots, e_k] *= U(e)$ stands for

$$\mathbf{if}\ \exists i.(e_i < 1 \vee e_i > n) \vee \exists i, j.(i \neq j \wedge e_i = e_j)\ \mathbf{then\ abort\ else}\ S_1; S_2; \cdots; S_{P(n,k)}\ \mathbf{end}$$

where each $S_\ell$ is of the form

$$\mathbf{if}\ e_1 = i_1 \wedge \cdots \wedge e_k = i_k\ \mathbf{then}\ q_{i_1}, \cdots, q_{i_k} *= U(e)\ \mathbf{else\ skip\ end}$$

and $(i_1, \cdots, i_k)$ ranges over $R$ (there are $P(n,k)$ elements in $R$). Note that the order of $S_\ell$'s is actually irrelevant as there is at most one that will be executed.

### 4.2 Operational and denotational semantics

A *configuration* is a triple $\langle S, \sigma, \rho\rangle$ where $S \in Prog \cup \{E\}$, $E$ is a special symbol to denote termination, $\sigma \in \Sigma$, and $\rho \in \mathcal{D}(\mathcal{H}_V)$ for some $V$ subsuming $qv(S)$. The operational semantics of programs in *Prog* is defined as the smallest transition relation $\to$ on configurations given in Table 2.

The definition is rather standard and intuitive. We would only like to point out that motivated by [Ying 2012], the operational semantics of quantum measurements (and even probabilistic assignments) are described in a non-deterministic way, while the probabilities of different branches are encoded in the quantum part of the configurations. That is why we need to take partial density operators instead of the normalised density operators as the representation of quantum states.

Similar to [Ying 2012], denotational semantics of cq-programs can be derived from the operational one by summing up all the cq-states obtained by terminating computations.

*Definition 4.1.* Let $S \in Prog$, and $\langle \sigma, \rho\rangle \in \mathcal{S}_V$ with $V \supseteq qv(S)$.

- A *computation* of $S$ starting in $\langle \sigma, \rho\rangle$ is a (finite or infinite) maximal sequence of configurations $\langle S_i, \sigma_i, \rho_i\rangle$, $i \ge 1$, such that

$$\langle S, \sigma, \rho\rangle \to \langle S_1, \sigma_1, \rho_1\rangle \to \langle S_2, \sigma_2, \rho_2\rangle \to \cdots$$

and $\rho_i \neq \mathbf{0}_{\mathcal{H}_V}$ for all $i$.

- A computation of $S$ is *terminating* in $\langle \sigma', \rho' \rangle$ if it is finite and the last configuration is $\langle E, \sigma', \rho' \rangle$; otherwise it is *diverging*.

Let $\rightarrow^n$ be the $n$-th composition of $\rightarrow$, and $\rightarrow^* \triangleq \bigcup_{n \geq 0} \rightarrow^n$. Then we have the following lemma.

Lemma 4.2. *Let $S \in Prog$, and $\langle \sigma, \rho \rangle \in \mathcal{S}_V$ with $V \supseteq qv(S)$. Then*

(1) *the multi-set $\{\langle \sigma', \rho' \rangle : \langle S, \sigma, \rho \rangle \rightarrow^n \langle E, \sigma', \rho' \rangle\}$ is countable for all $n \geq 0$.*
(2) *the sequence of cq-states $\{\Delta_n : n \geq 0\}$, where*

$$\Delta_n = \sum \left\{ \langle \sigma', \rho' \rangle : \langle S, \sigma, \rho \rangle \rightarrow^k \langle E, \sigma', \rho' \rangle \text{ for some } k \leq n \right\},$$

*is increasing with respect to $\sqsubseteq$. Thus*

$$\sum \{\langle \sigma', \rho' \rangle : \langle S, \sigma, \rho \rangle \rightarrow^* \langle E, \sigma', \rho' \rangle\} = \bigvee_{n \geq 0} \Delta_n.$$

With this lemma, we are able to define the denotational semantics of cq-programs using the operational one. Let $\mathcal{S}_{\supseteq qv(S)} \triangleq \bigcup_{V \supseteq qv(S)} \mathcal{S}_V$.

*Definition 4.3.* Let $S \in Prog$. The *denotational semantics* of $S$ is a mapping

$$[\![S]\!] : \mathcal{S}_{\supseteq qv(S)} \rightarrow \mathcal{S}_{\supseteq qv(S)}$$

such that for any $\langle \sigma, \rho \rangle \in \mathcal{S}_V$ with $V \supseteq qv(S)$,

$$[\![S]\!](\sigma, \rho) = \sum \{\langle \sigma', \rho' \rangle : \langle S, \sigma, \rho \rangle \rightarrow^* \langle E, \sigma', \rho' \rangle\}.$$

Furthermore, let $[\![S]\!](\Delta) = \sum_{i \in I} [\![S]\!](\sigma_i, \rho_i)$ whenever $\Delta = \bigoplus_{i \in I} \langle \sigma_i, \rho_i \rangle$.

To simplify notation, we always write $(\sigma, \rho)$ for $(\langle \sigma, \rho \rangle)$ when $\langle \sigma, \rho \rangle$ appears as a parameter of some function. The next lemma guarantees the well-definedness of Definition 4.3.

Lemma 4.4. *For any $S \in Prog$ and $\Delta \in \mathcal{S}_V$ with $V \supseteq qv(S)$,*

(1) $\mathrm{tr}([\![S]\!](\Delta)) \leq \mathrm{tr}(\Delta)$, *and so* $[\![S]\!](\Delta) \in \mathcal{S}_V$;
(2) $[\![S]\!](\Delta) = \sum_i \lambda_i [\![S]\!](\Delta_i)$ *whenever* $\Delta = \sum_i \lambda_i \Delta_i$.

To illustrate the concepts and techniques introduced in this paper, we take Grover's search algorithm [Grover 1996] as a running example. More case studies are presented in the Appendix.

*Example 4.5 (Grover's algorithm).* Suppose we are given an (unstructured) database with $N$ items, $D$ of which are of our concern (called solutions) with $0 < D < N/2$. For simplicity, we assume $N = 2^n$ for some positive integer $n$. Let $\theta \in (0, \pi/2)$ such that

$$\cos \frac{\theta}{2} = \sqrt{\frac{2^n - D}{2^n}},$$

and $K$ be the integer in $(\frac{\pi}{2\theta} - 1, \frac{\pi}{2\theta}]$. Then Grover's search algorithm can be described in our quantum language (with syntactic sugars) as

*Grover* $\triangleq$

$\bar{q} := 0;\ \bar{q} *= H^{\otimes n};\ x := 0;$
**while** $x < K$ **do**
  $\bar{q} *= G;\ x := x + 1;$
**end**
$y :=$ **meas** $\bar{q}$

where $\bar{q} = q_1, \cdots, q_n$ and each $q_i$ has **Qubit**-type, $H = |+\rangle\langle 0| + |-\rangle\langle 1|$ is the Hadamard operator with $|\pm\rangle \triangleq \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$. $G = (2|\psi\rangle\langle\psi| - I)O$ is the Grover rotation where $|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle$ and $O$ is the Grover oracle which maps $|i\rangle$ to $-|i\rangle$ when $i$ is a solution while to $|i\rangle$ otherwise.

The (terminating) computations of *Grover* starting in any $\langle \sigma, \rho\rangle \in \mathcal{S}_{\bar{q}}$ are shown as follows.

$$
\begin{aligned}
& \langle Grover, \sigma, \rho\rangle \\
\to^n \quad & \langle \bar{q} *= H^{\otimes n};\ \cdots, \sigma, |0^{\otimes n}\rangle\langle 0^{\otimes n}|\rangle \\
\to \quad & \langle x := 0;\ \cdots, \sigma, |+^{\otimes n}\rangle\langle +^{\otimes n}|\rangle \\
\to \quad & \langle \textbf{while}; y := \textbf{meas}\ \bar{q}, \sigma[0/x], |+^{\otimes n}\rangle\langle +^{\otimes n}|\rangle \\
\to \quad & \langle \bar{q} *= G;\ x := x+1;\ \textbf{while}; y := \textbf{meas}\ \bar{q}, \sigma[0/x], |+^{\otimes n}\rangle\langle +^{\otimes n}|\rangle \\
\to \quad & \langle x := x+1;\ \textbf{while}; y := \textbf{meas}\ \bar{q}, \sigma[0/x], G|+^{\otimes n}\rangle\langle +^{\otimes n}|G^\dagger\rangle \\
\to \quad & \langle \textbf{while}; y := \textbf{meas}\ \bar{q}, \sigma[1/x], G|+^{\otimes n}\rangle\langle +^{\otimes n}|G^\dagger\rangle \\
\to \quad & \cdots\ \cdots \\
\to \quad & \langle \textbf{while}; y := \textbf{meas}\ \bar{q}, \sigma[K/x], G^K|+^{\otimes n}\rangle\langle +^{\otimes n}|G^{K\dagger}\rangle \\
\to \quad & \langle y := \textbf{meas}\ \bar{q}, \sigma[K/x], G^K|+^{\otimes n}\rangle\langle +^{\otimes n}|G^{K\dagger}\rangle \\
\to^{n+1} \quad & \langle E, \sigma[K/x, i/y], |\langle i|G^K|+^{\otimes n}\rangle|^2 \cdot |i\rangle\langle i|\rangle.
\end{aligned}
$$

where $0 \le i < 2^n$. We write **while** for the while loop in the program. Consequently,

$$
[\![Grover]\!](\sigma, \rho) = \sum_{i=0}^{2^n-1} \langle \sigma[K/x, i/y], |\langle i|G^K|+^{\otimes n}\rangle|^2 \cdot |i\rangle\langle i|\rangle. \tag{2}
$$

Let $Sol \subseteq \{0, \cdots, 2^n - 1\}$ be the set of solutions, $|Sol| = D$, and

$$
|\alpha\rangle = \frac{1}{\sqrt{2^n - D}} \sum_{i \notin Sol} |i\rangle, \quad |\beta\rangle = \frac{1}{\sqrt{D}} \sum_{i \in Sol} |i\rangle.
$$

Then we have $|\psi\rangle = |+^{\otimes n}\rangle = \cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|\beta\rangle$,

$$
G|\alpha\rangle = \cos\theta|\alpha\rangle + \sin\theta|\beta\rangle, \quad G|\beta\rangle = -\sin\theta|\alpha\rangle + \cos\theta|\beta\rangle.
$$

That is, the effect of $G$ in the two-dimensional real space spanned by $|\alpha\rangle$ and $|\beta\rangle$ is a rotation with angle $\theta$ (note that $|\alpha\rangle$ and $|\beta\rangle$ are orthogonal). Thus the success probability of finding a solution by Grover's algorithm, i.e. the probability of $y \in Sol$ after its execution, can be computed as

$$
p_{\text{succ}} = \sum_{i \in Sol} \left|\langle i|G^K|+^{\otimes n}\rangle\right|^2 = \sin^2\left(\frac{2K+1}{2}\theta\right). \tag{3}
$$

Recall that $K \in (\frac{\pi}{2\theta} - 1, \frac{\pi}{2\theta}]$. Thus

$$
1 - p_{\text{succ}} \le |(2K+1)\theta - \pi| \le \theta.
$$

In other words, Grover's algorithm succeeds with a probability at least $1 - O(\sqrt{D/N})$, and runs in time $O(\sqrt{N/D})$, achieving a quadratic speedup over the best classical algorithms which run in $O(N/D)$ time.

The following lemma presents the explicit form for denotational semantics of various program constructs.

Lemma 4.6. *For any cq-state $\langle \sigma, \rho\rangle$ which makes the corresponding formula meaningful,*

(1) $[\![\textbf{skip}]\!](\sigma, \rho) = \langle \sigma, \rho\rangle$;
(2) $[\![x := e]\!](\sigma, \rho) = \langle \sigma[\sigma(e)/x], \rho\rangle$;

(3) $[\![x :=_\$ g]\!](\sigma,\rho) = \sum_{d\in D_{type(x)}} \langle \sigma[d/x], g(d)\cdot\rho\rangle$;
(4) $[\![x := \textbf{meas}\ \mathcal{M}[\bar{q}]]\!](\sigma,\rho) = \sum_{i\in J}\langle \sigma[i/x], M_i\rho M_i^\dagger\rangle$ *where* $\mathcal{M} = \{M_i : i\in J\}$;
(5) $[\![q := 0]\!](\sigma,\rho) = \langle \sigma, \sum_{i=0}^{d_q-1} |0\rangle_q\langle i|\rho|i\rangle_q\langle 0|\rangle\rangle$;
(6) $[\![\bar{q} *= U]\!](\sigma,\rho) = \langle \sigma, U_{\bar{q}}\rho U_{\bar{q}}^\dagger\rangle$;
(7) $[\![S_0; S_1]\!](\sigma,\rho) = [\![S_1]\!]([\![S_0]\!](\sigma,\rho))$;
(8) $[\![(S_0; S_1); S_2]\!] = [\![S_0; (S_1; S_2)]\!]$;
(9) $[\![\textbf{if}\ b\ \textbf{then}\ S_1\ \textbf{else}\ S_0\ \textbf{end}]\!](\sigma,\rho) = [\![S_1]\!](\sigma,\rho)$ *if* $\sigma \models b$, *and* $[\![S_0]\!](\sigma,\rho)$ *otherwise;*
(10) $[\![\textbf{while}]\!](\sigma,\rho) = \bigvee_n([\![(\textbf{while})^n]\!](\sigma,\rho))$, *where* $\textbf{while} \triangleq \textbf{while}\ b\ \textbf{do}\ S\ \textbf{end}$, $(\textbf{while})^0 \triangleq \textbf{abort}$, *and for any* $n \geq 0$,

$$(\textbf{while})^{n+1} \triangleq \textbf{if}\ b\ \textbf{then}\ S; (\textbf{while})^n\ \textbf{else skip end}.$$

The next lemma gives a recursive description of the semantics of while loops, which is easy from Lemma 4.6.

Lemma 4.7. *Let* $\textbf{while} \triangleq \textbf{while}\ b\ \textbf{do}\ S\ \textbf{end}$. *For any* $\Delta \in \mathcal{S}_{qv(S)}$ *and* $n \geq 0$,

$$\Delta|_{\neg b} + [\![(\textbf{while})^n]\!]([\![S]\!](\Delta|_b)) = [\![(\textbf{while})^{n+1}]\!](\Delta).$$

*In particular,*

$$\Delta|_{\neg b} + [\![(\textbf{while})]\!]([\![S]\!](\Delta|_b)) = [\![(\textbf{while})]\!](\Delta).$$

Finally, we can easily compute the operational semantics of the syntactic sugars introduced in Sec 4.1.

Lemma 4.8. *Let* $n = |\bar{q}|$, $1 \leq k \leq n$, *and* $|\bar{U}| = K$. *For any cq-state* $\langle\sigma,\rho\rangle \in \mathcal{S}_V$ *which makes the corresponding formula meaningful,*

(1) $[\![\textbf{abort}]\!](\sigma,\rho) = \perp_V$;
(2) $[\![x := \textbf{meas}\ \bar{q}]\!](\sigma,\rho) = \sum_{i=0}^{d_{\bar{q}}-1}\langle \sigma[i/x], |i\rangle_{\bar{q}}\langle i|\rho|i\rangle_{\bar{q}}\langle i|\rangle$;
(3) $[\![\bar{q} := 0]\!](\sigma,\rho) = \langle \sigma, \sum_{i=0}^{d_{\bar{q}}-1} |0\rangle_{\bar{q}}\langle i|\rho|i\rangle_{\bar{q}}\langle 0|\rangle\rangle$;
(4) $[\![\bar{q}[e_1,\cdots,e_k] *= U(e)]\!](\sigma,\rho) = \perp_V$ *if* $\sigma(e) < 1$, $\sigma(e) > K$, *there exists* $i$ *such that* $\sigma(e_i) < 1$ *or* $\sigma(e_i) > n$, *or* $\sigma(e_i)$*'s are not distinct; otherwise it equals* $\langle \sigma, U(i)_{\bar{r}}\rho U(i)_{\bar{r}}^\dagger\rangle$ *where* $i = \sigma(e)$ *and* $\bar{r} = q_{\sigma(e_1)},\cdots,q_{\sigma(e_k)}$.

### 4.3 Correctness formula

As usual, program correctness is expressed by *correctness formulas* with the form

$$\{\Theta\}\ S\ \{\Psi\}$$

where $S$ is a cq-program, and $\Theta$ and $\Psi$ are both cq-assertions. Note here that we do not put any requirement on the quantum variables which $\Theta$ and $\Psi$ are acting on. In fact, the sets $qv(S)$, $qv(\Theta)$, and $qv(\Psi)$ can be all different.

The following definition is a direct extension of the corresponding one in [Ying 2012], with the new notions of cq-states and assertions.

*Definition 4.9.* Let $S$ be a cq-program, and $\Theta$ and $\Psi$ cq-assertions.

(1) We say the correctness formula $\{\Theta\}\ S\ \{\Psi\}$ is true in the sense of *total correctness*, written $\models_{tot} \{\Theta\}\ S\ \{\Psi\}$, if for any $V \supseteq qv(S,\Theta,\Psi)$ and $\Delta \in \mathcal{S}_V$,

$$\mathrm{Exp}(\Delta \models \Theta) \leq \mathrm{Exp}([\![S]\!](\Delta) \models \Psi).$$

(2) We say the correctness formula $\{\Theta\}\ S\ \{\Psi\}$ is true in the sense of *partial correctness*, written $\models_{par}\ \{\Theta\}\ S\ \{\Psi\}$, if for any $V \supseteq qv(S,\Theta,\Psi)$ and $\Delta \in \mathcal{S}_V$,

$$\mathrm{Exp}(\Delta \models \Theta) \leq \mathrm{Exp}([\![S]\!](\Delta) \models \Psi) + \mathrm{tr}(\Delta) - \mathrm{tr}([\![S]\!](\Delta)).$$

The next lemma shows that the validity of correctness formulas can be checked on simple cq-states.

Lemma 4.10. *Let $S$ be a cq-program, $\Theta$ and $\Psi$ be cq-assertions, and $V \triangleq qv(S,\Theta,\Psi)$. Then*

(1) $\models_{tot}\ \{\Theta\}\ S\ \{\Psi\}$ *iff for any* $\langle\sigma,\rho\rangle \in \mathcal{S}_V$ *with* $\mathrm{tr}(\rho)=1$,

$$\mathrm{Exp}(\langle\sigma,\rho\rangle \models \Theta) \leq \mathrm{Exp}([\![S]\!](\sigma,\rho) \models \Psi).$$

(2) $\models_{par}\ \{\Theta\}\ S\ \{\Psi\}$ *iff for any* $\langle\sigma,\rho\rangle \in \mathcal{S}_V$ *with* $\mathrm{tr}(\rho)=1$,

$$\mathrm{Exp}(\langle\sigma,\rho\rangle \models \Theta) \leq \mathrm{Exp}([\![S]\!](\sigma,\rho) \models \Psi) + \mathrm{tr}(\rho) - \mathrm{tr}([\![S]\!](\sigma,\rho).$$

*Example 4.11.* We have proven in Example 4.5 that no matter what the initial (classical and quantum) state is, the output (value of $y$) of Grover's algorithm lies in *Sol* with probability $p_{succ}$. This correctness can also be stated in the following form

$$\models_{tot}\ \{p_{succ}\}\ Grover\ \{y \in Sol\}, \tag{4}$$

which claims that the postcondition $y \in Sol$ can be established by *Grover* with probability $p_{succ}$. Recall that in Eq.(4), $p_{succ}$ denotes $\langle \mathbf{true}, p_{succ}\rangle$ and $y \in Sol$ denotes $\langle y \in Sol, 1\rangle$. Both the pre- and post-conditions being purely classical means that the initial and final quantum states are irrelevant.

Note that $qv(Grover) = \bar{q}$. For any $\langle\sigma,\rho\rangle \in \mathcal{S}_{\bar{q}}$ with $\mathrm{tr}(\rho)=1$, we have from Eq.(2) that

$$\begin{aligned}\mathrm{Exp}([\![Grover]\!](\langle\sigma,\rho\rangle) \models y \in Sol) &= \sum_{i\in Sol} |\langle i|G^K|+^{\otimes n}\rangle|^2 \\ &= p_{succ} = \mathrm{Exp}(\langle\sigma,\rho\rangle \models p_{succ}).\end{aligned}$$

Then Eq.(4) follows from Lemma 4.10.

Some basic facts about total and partial correctness follow from Lemma 3.9.

Lemma 4.12. *Let $S$ be a cq-program, $\Theta$ and $\Psi$ be cq-assertions, and $V \subseteq qVar$.*

(1) *If* $\models_{tot}\ \{\Theta\}\ S\ \{\Psi\}$ *then* $\models_{par}\ \{\Theta\}\ S\ \{\Psi\}$;
(2) $\models_{tot}\ \{\bot_V\}\ S\ \{\Psi\}$;
(3) $\models_{par}\ \{\Theta\}\ S\ \{\top_V\}$;
(4) *If* $\models_{tot}\ \{\Theta_i\}\ S\ \{\Psi_i\}$ *and* $\lambda_i \geq 0$ *for* $i=1,2$, *then*

$$\models_{tot}\ \{\lambda_1\Theta_1 + \lambda_2\Theta_2\}\ S\ \{\lambda_1\Psi_1 + \lambda_2\Psi_2\}.$$

*The result also holds for partial correctness if* $\lambda_1 + \lambda_2 = 1$.

### 4.4 Weakest (liberal) precondition semantics

Recall that in classical programming theory, the weakest (liberal) precondition of an assertion $p$ with respect to a given program $S$ characterises the largest set of states $\sigma$ which (upon termination) guarantee that the final states $[\![S]\!](\sigma)$ satisfying $p$. Consequently, a program can also be regarded as a predicate transformer which maps any postcondition to its weakest (liberal) precondition. In the following, we extend these semantics to our cq-programs. Let $\mathcal{A}_{\supseteq qv(S)} \triangleq \bigcup_{V\supseteq qv(S)} \mathcal{A}_V$.

*Definition 4.13.* Let $S \in Prog$. The *weakest precondition semantics* $wp.S$ and *weakest liberal precondition semantics* $wlp.S$ of $S$ are both mappings

$$\mathcal{A}_{\supseteq qv(S)} \to \mathcal{A}_{\supseteq qv(S)}$$

$xp.\mathbf{skip}.\Theta = \Theta$ $\qquad$ $xp.(x :=_{\$} g).\Theta = \sum_{d \in D_{type(x)}} g(d) \cdot \Theta[d/x]$

$xp.(x := e).\Theta = \Theta[e/x]$ $\qquad$ $xp.(x := \mathbf{meas}\ \mathcal{M}[\bar{q}]).\Theta = \sum_{i \in J} M_i^\dagger \Theta[i/x] M_i$

$xp.(\bar{q} *= U).\Theta = U_{\bar{q}}^\dagger \Theta U_{\bar{q}}$ $\qquad$ $xp.(q := 0).\Theta = \sum_{i=0}^{d_q - 1} |i\rangle_q\langle 0|\Theta|0\rangle_q\langle i|$

$xp.(S_0; S_1).\Theta = xp.S_0.(xp.S_1.\Theta)$ $\qquad$ $xp.(\mathbf{if}\ b\ \mathbf{then}\ S_1\ \mathbf{else}\ S_0\ \mathbf{end}).\Theta = b \wedge xp.S_1.\Theta + \neg b \wedge xp.S_0.\Theta$

$wlp.(\mathbf{while}\ b\ \mathbf{do}\ S\ \mathbf{end}).\Theta = \bigwedge_{n \geq 0} \Theta_n$, where $\Theta_0 \triangleq \top_V$, and for any $n \geq 0$,

$$\Theta_{n+1} \triangleq \neg b \wedge \Theta + b \wedge wlp.S.\Theta_n.$$

$wp.(\mathbf{while}\ b\ \mathbf{do}\ S\ \mathbf{end}).\Theta = \bigvee_{n \geq 0} \Theta_n$, where $\Theta_0 \triangleq \bot_V$, and for any $n \geq 0$,

$$\Theta_{n+1} \triangleq \neg b \wedge \Theta + b \wedge wp.S.\Theta_n.$$

Table 3. Weakest (liberal) precondition semantics for cq-programs, where $xp \in \{wp, wlp\}$.

defined inductively in Table 3. To simplify notation, we use $xp$ to denote $wp$ or $wlp$ whenever it is applicable for both of them.

We follow the standard notations $wp.S.\Theta$ and $wlp.S.\Theta$ to denote weakest (liberal) preconditions [Dijkstra et al. 1976; Morgan et al. 1996; Ying 2012]. The well-definedness of Definition 4.13 follows from the observation that $xp.S$ is monotonic on $\mathcal{H}_V$ (with respect to $\sqsubseteq_V$; see Lemma 4.16(3) below) for any cq-program $S$ and $V \supseteq qv(S)$. The following lemma shows a duality relation between the denotational and weakest (liberal) precondition semantics of cq-programs.

Lemma 4.14. *Let $S$ be a cq-program, $\Delta$ a cq-state, and $\Theta$ a cq-assertion with $qv(\Delta) \supseteq qv(\Theta) \supseteq qv(S)$. Then*

(1) $\mathrm{Exp}(\Delta \models wp.S.\Theta) = \mathrm{Exp}([\![S]\!](\Delta) \models \Theta)$;
(2) $\mathrm{Exp}(\Delta \models wlp.S.\Theta) = \mathrm{Exp}([\![S]\!](\Delta) \models \Theta) + \mathrm{tr}(\Delta) - \mathrm{tr}([\![S]\!](\Delta))$.

Using Lemmas 4.8 and 4.14, we can also compute the weakest (liberal) precondition semantics of the syntactic sugars introduced in Sec 4.1.

Lemma 4.15. *Let $n = |\bar{q}|$, $1 \leq k \leq n$, $|\bar{U}| = K$, and $xp \in \{wp, wlp\}$. Let $\Theta$ be a cq-assertion in $\mathcal{A}_V$ with $V$ containing the quantum variables of the corresponding cq-program. Then*

(1) $wp.\mathbf{abort}.\Theta = \bot_V$, $wlp.\mathbf{abort}.\Theta = \top_V$;
(2) $xp.(x := \mathbf{meas}\ \bar{q}).\Theta = \sum_{i=0}^{d_{\bar{q}}-1} |i\rangle_{\bar{q}}\langle i|\Theta[i/x]|i\rangle_{\bar{q}}\langle i|$;
(3) $xp.(\bar{q} := 0).\Theta = \sum_{i=0}^{d_{\bar{q}}-1} |i\rangle_{\bar{q}}\langle 0|\Theta|0\rangle_{\bar{q}}\langle i|$;
(4) *Let $S \triangleq \bar{q}[e_1, \cdots, e_k] *= U(e)$. Then*

$$wp.S.\Theta = \sum_{i_1, \cdots, i_k = 1, distinct}^{n} \sum_{i=1}^{K} \bigwedge_{j=1}^{k} (e_j = i_j) \wedge (e = i) \wedge \mathcal{U}^i_{i_1, \cdots, i_k}(\Theta)$$

*where $\mathcal{U}^i_{i_1, \cdots, i_k}(\Theta) \triangleq U(i)_{\bar{r}}^\dagger \Theta U(i)_{\bar{r}}$ and $\bar{r} \triangleq q_{i_1}, \cdots, q_{i_k}$.*

The following collects some properties of the weakest (liberal) precondition semantics, which are directly from Lemmas 4.14, 3.9 and 3.11.

Lemma 4.16. *Let $S$ be a cq-program, $\Delta$ a cq-state, and $\Theta$ a cq-assertion with $qv(\Delta) \supseteq qv(\Theta) \supseteq qv(S)$. Let $xp \in \{wp, wlp\}$. Then*

(1) $wp.S.\Theta + wlp.S.(\top_{qv(\Theta)} - \Theta) = \top_{qv(\Theta)}$;
(2) *if $(V \cup W) \cap qv(S) = \emptyset$, and $\mathcal{F}_{V\to W}$ is completely positive and sub-unital super-operator, then*

$$xp.S.(\mathcal{F}_{V\to W}(\Theta)) = \mathcal{F}_{V\to W}(xp.S.\Theta);$$

(3) *the function $xp.S$ is monotonic; that is, for all $\Theta_1 \sqsubseteq \Theta_2$, $xp.S.\Theta_1 \sqsubseteq xp.S.\Theta_2$;*
(4) *the function $wp.S$ is linear; that is, for all $\Theta_1, \Theta_2 \in \mathcal{A}_V$,*

$$wp.S.(\lambda_1\Theta_1 + \lambda_2\Theta_2) = \lambda_1 wp.S.\Theta_1 + \lambda_2 wp.S.\Theta_2;$$

(5) *the function $wlp.S$ is affine-linear; that is, for all $\Theta_1, \Theta_2 \in \mathcal{A}_V$ and $\lambda_1 + \lambda_2 = 1$,*

$$wlp.S.(\lambda_1\Theta_1 + \lambda_2\Theta_2) = \lambda_1 wlp.S.\Theta_1 + \lambda_2 wlp.S.\Theta_2.$$

Note that from Lemmas 4.14 and 3.10, if $qv(\Theta) = qv(\Psi) \supseteq qv(S)$, then $\models_{tot} \{\Theta\}\ S\ \{\Psi\}$ iff $\Theta \sqsubseteq wp.S.\Psi$, and $\models_{par} \{\Theta\}\ S\ \{\Psi\}$ iff $\Theta \sqsubseteq wlp.S.\Psi$. To conclude this section, we extend this result (and a similar one for partial correctness) to the general case.

Lemma 4.17. *Let $S$ be a cq-program, and $\Theta$ and $\Psi$ are cq-assertions. Then*

$$\models_{tot} \{\Theta\}\ S\ \{\Psi\} \quad \textit{iff} \quad \Theta \lesssim wp.S.(\Psi \otimes I_{qv(S)\setminus qv(\Psi)})$$
$$\models_{par} \{\Theta\}\ S\ \{\Psi\} \quad \textit{iff} \quad \Theta \lesssim wlp.S.(\Psi \otimes I_{qv(S)\setminus qv(\Psi)}).$$

## 5 HOARE LOGIC FOR CQ-PROGRAMS

The core of Hoare logic is a proof system consisting of axioms and proof rules which enable syntax-oriented and modular reasoning of program correctness. In this section, we propose a Hoare logic for cq-programs.

### 5.1 Partial correctness

We propose in Table 4 the proof system for partial correctness of cq-programs, which looks quite similar to the standard Hoare logic, thanks to the novel definition of cq-assertions. Several cases deserve explanation. The side conditions in rules (Init), (Unit), and (Meas) are introduced to guarantee the well-definedness of the corresponding preconditions. They can always be satisfied by introducing 'dull' quantum variables (i.e. tensor product with appropriate identity operators) with the help of rule (Imp). To use rule (If), we first split the precondition into two parts: $\Theta = b\wedge\Theta + \neg b\wedge\Theta$. In the first one, all the classical states satisfy $b$, thus the first premise $\{b \wedge \Theta\}\ S_1\ \{\Psi\}$ is employed; in the second part, all classical states satisfy $\neg b$, thus the second premise is employed. The cq-assertion $\Theta$ in rule (While) plays a similar role of 'loop invariant' as in classical programs. Finally, as the pre- and postconditions can act on different quantum variables, we need the pre-order $\lesssim$ for rule (Imp) rather than the Löwner order in [Ying 2012] etc.

Note also that in the rules in Table 4, substitutions (say, $\Theta[i/x]$ in (Meas)) and Boolean operations (say, $b\wedge\Theta$ in (If)) are applied on the classical part of $\Theta$, while super-operators (say, $U_{\bar{q}}^{\dagger}\Theta U_{\bar{q}}$ in (Unit)) are on the quantum part only. For example, let $\Theta \triangleq \bigoplus_{k\in K}\langle p_k, N_k\rangle$. Then rule (Meas) actually claims that if $q \in qv(\Theta)$,

$$\left\{\sum_{i\in J}\sum_{k\in K}\langle p_k[i/x], M_i^{\dagger}N_kM_i\rangle\right\} x := \textbf{meas}\ \mathcal{M}[\bar{q}]\{\Theta\}.$$

We write $\vdash_{par} \{\Theta\}\ S\ \{\Psi\}$ if the correctness formula $\{\Theta\}\ S\ \{\Psi\}$ can be derived using the axioms and rules presented in Table 4.

(Skip) $\{\Theta\}\ \mathbf{skip}\ \{\Theta\}$

(Assn) $\{\Theta[e/x]\}\ x := e\ \{\Theta\}$

(Rassn) $\left\{\sum_{d\in D_{type(x)}} g(d)\cdot\Theta[d/x]\right\} x :=_{\$} g\{\Theta\}$

(Init) $$\frac{q\in qv(\Theta)}{\{\sum_{i=0}^{d_q-1}|i\rangle_q\langle 0|\Theta|0\rangle_q\langle i|\}\ q := 0\ \{\Theta\}}$$

(Unit) $$\frac{\bar{q}\subseteq qv(\Theta)}{\{U_{\bar{q}}^\dagger\Theta U_{\bar{q}}\}\ \bar{q}\ *= U\ \{\Theta\}}$$

(Meas) $$\frac{q\in qv(\Theta)}{\{\sum_{i\in J}M_i^\dagger\Theta[i/x]M_i\}\ x := \mathbf{meas}\ \mathcal{M}[\bar{q}]\ \{\Theta\}}$$

(Seq) $$\frac{\{\Theta\}\ S_0\ \{\Theta'\},\ \{\Theta'\}\ S_1\ \{\Psi\}}{\{\Theta\}\ S_0; S_1\ \{\Psi\}}$$

(If) $$\frac{\{b\wedge\Theta\}\ S_1\ \{\Psi\},\ \{\neg b\wedge\Theta\}\ S_0\ \{\Psi\}}{\{\Theta\}\ \mathbf{if}\ b\ \mathbf{then}\ S_1\ \mathbf{else}\ S_0\ \mathbf{end}\ \{\Psi\}}$$

(While) $$\frac{\{b\wedge\Theta\}\ S\ \{\Theta\}}{\{\Theta\}\ \mathbf{while}\ b\ \mathbf{do}\ S\ \mathbf{end}\ \{\neg b\wedge\Theta\}}$$

(Imp) $$\frac{\Theta\lesssim\Theta',\ \{\Theta'\}\ S\ \{\Psi'\},\ \Psi'\lesssim\Psi}{\{\Theta\}\ S\ \{\Psi\}}$$

Table 4. Proof system for partial correctness.

Recall the proof rule for loop programs in [Ying 2012]:

$$\frac{\{M\}\ S\ \{\mathcal{E}_{\bar{q}}^0(N)+\mathcal{E}_{\bar{q}}^1(M)\}}{\{\mathcal{E}_{\bar{q}}^0(N)+\mathcal{E}_{\bar{q}}^1(M)\}\ \mathbf{while}\ \mathcal{M}[\bar{q}]\ \mathbf{do}\ S\ \mathbf{end}\ \{N\}} \tag{5}$$

where $\mathcal{E}_{\bar{q}}^i(M)\triangleq M_i^\dagger M M_i$ where $\mathcal{M}=\{M_0,M_1\}$. Now we show how this rule can be derived in our proof system, when the assertions like $M$ in Eq.(5) are replaced by cq-assertions of the form $\langle\mathbf{true},M\rangle$. That is, we are going to show

$$\vdash_{par}\{\langle\mathbf{true},M\rangle\}\ S\ \{\langle\mathbf{true},\mathcal{E}_{\bar{q}}^0(N)+\mathcal{E}_{\bar{q}}^1(M)\rangle\} \tag{6}$$

implies

$$\vdash_{par}\{\langle\mathbf{true},\mathcal{E}_{\bar{q}}^0(N)+\mathcal{E}_{\bar{q}}^1(M)\rangle\}\ x := \mathbf{meas}\ \mathcal{M}[\bar{q}];\mathbf{while}\ x=1\ \mathbf{do}\ S; x := \mathbf{meas}\ \mathcal{M}[\bar{q}];\ \mathbf{end}\ \{\langle\mathbf{true},N\rangle\}. \tag{7}$$

First we have

$\{\langle x=1,M\rangle\}$

$\{\langle\mathbf{true},M\rangle\}$ (*Imp*)

$S$;

$\{\langle\mathbf{true},\mathcal{E}_{\bar{q}}^0(N)+\mathcal{E}_{\bar{q}}^1(M)\rangle\}$ *Eq.*(6)

$\{\langle 0=1,\mathcal{E}_{\bar{q}}^0(M)\rangle+\langle 0\neq 1,\mathcal{E}_{\bar{q}}^0(N)\rangle+\langle 1=1,\mathcal{E}_{\bar{q}}^1(M)\rangle+\langle 1\neq 1,\mathcal{E}_{\bar{q}}^1(N)\rangle\}$ (*Imp*)

$x := \mathbf{meas}\ \mathcal{M}[\bar{q}]$;

$\{\langle x=1,M\rangle+\langle x\neq 1,N\rangle\}$. (*Meas*)

Then, using the (While) rule,

$$\vdash_{par}\{\langle x=1,M\rangle+\langle x\neq 1,N\rangle\}\ \mathbf{while}\ x=1\ \mathbf{do}\ S; x := \mathbf{meas}\ \mathcal{M}[\bar{q}];\ \mathbf{end}\ \{\langle x\neq 1,N\rangle\}.$$

Finally, the following reasoning

$$\begin{aligned}
&\{\langle \mathbf{true}, \mathcal{E}_{\bar{q}}^0(N) + \mathcal{E}_{\bar{q}}^1(M)\rangle\}\\
&x := \mathbf{meas}\ \mathcal{M}[\bar{q}];\\
&\{\langle x = 1, M\rangle + \langle x \neq 1, N\rangle\} && (Meas, Imp)\\
&\mathbf{while}\ x = 1\ \mathbf{do}\ S;\ x := \mathbf{meas}\ \mathcal{M}[\bar{q}];\ \mathbf{end}\\
&\{\langle x \neq 1, N\rangle\}\\
&\{\langle \mathbf{true}, N\rangle\} && (Imp)
\end{aligned}$$

gives us the proof of Eq.(7) as desired.

Now we show the soundness and (relative) completeness of the proof system in the sense of partial correctness.

Theorem 5.1. *The proof system in Table 4 is both sound and complete with respect to the partial correctness of cq-programs.*

### 5.2 Total correctness

Ranking functions play a central role in proving total correctness of while loop programs. Recall that in the classical case, a ranking function maps each reachable state in the loop body to an element of a well-founded ordered set (say, the set $\mathbb{N}$ of nonnegative integers), such that the value decreases strictly after each iteration of the loop. Our proof rule for total correctness of while loops also heavily relies on the notion of ranking assertions.

*Definition 5.2.* Let $\Theta \in \mathcal{A}_V$. A decreasing sequence (w.r.t. $\sqsubseteq$) of cq-assertions $\{\Theta_n : n \geq 0\}$ in $\mathcal{A}_V$ are $\Theta$-*ranking assertions* for **while** $b$ **do** $S$ **end** if

(1) $\Theta \sqsubseteq \Theta_0$ and $\bigwedge_n \Theta_n = \bot_V$;
(2) for any $n \geq 0$ and $\Delta \in \mathcal{S}_{qv(S)\cup V}$,

$$\mathrm{Exp}([\![S]\!](\Delta|_b) \models \Theta_n) \leq \mathrm{Exp}(\Delta \models \Theta_{n+1}).$$

An alternative definition of $\Theta$-ranking assertions, which uses the weakest precondition semantics instead of the denotational one, is to replace the second clause above by $b \wedge wp.S.\Theta_n \sqsubseteq \Theta_{n+1}$. It is easy to show that these two definitions are equivalent.

With the notion of ranking assertions, we can state the proof rule for while loops in total correctness as follows:

$$\text{(WhileT)}\quad \frac{\{b \wedge \Theta\}\ S\ \{\Theta\} \qquad \Theta\text{-ranking assertions exist for } \mathbf{while}\ b\ \mathbf{do}\ S\ \mathbf{end}}{\{\Theta\}\ \mathbf{while}\ b\ \mathbf{do}\ S\ \mathbf{end}\ \{\neg b \wedge \Theta\}}$$

The proof system for total correctness is then defined as for partial correctness, except that the rule (While) is replaced by (WhileT). We write $\vdash_{tot} \{\Theta\}\ S\ \{\Psi\}$ if the correctness formula $\{\Theta\}\ S\ \{\Psi\}$ can be derived using the proof system for total correctness.

Recall that in [Ying 2012], a notion of bound function is proposed for proving total correctness of purely quantum programs. Let $M \in \mathcal{P}(\mathcal{H})$ and $\epsilon > 0$. A function

$$t : \mathcal{D}(\mathcal{H}) \to \mathbb{N}$$

is called a $(M, \epsilon)$-bound one for the loop **while** $\mathcal{M}[\bar{q}]$ **do** $S$ **end** where $\mathcal{M} = \{M_0, M_1\}$ if for any $\rho \in \mathcal{D}(\mathcal{H})$,

(1) $t([\![S]\!](\mathcal{E}_{\bar{q}}^1(\rho))) \leq t(\rho)$,
(2) if $\mathrm{tr}(M\rho) \geq \epsilon$ then $t([\![S]\!](\mathcal{E}_{\bar{q}}^1(\rho))) < t(\rho)$.

With the bound functions, the proof rule for total correctness of quantum loops reads as follows:

$$\frac{\begin{array}{l}\{M\}\ S\ \{\mathcal{E}_{\bar{q}}^0(N)+\mathcal{E}_{\bar{q}}^1(M)\}\\ \text{for each } \epsilon>0,\ t_\epsilon \text{ is a } (\mathcal{E}_{\bar{q}}^1(M),\epsilon)\text{-bound function for } \textbf{while } \mathcal{M}[\bar{q}] \textbf{ do } S \textbf{ end}\end{array}}{\{\mathcal{E}_{\bar{q}}^0(N)+\mathcal{E}_{\bar{q}}^1(M)\}\ \textbf{while } \mathcal{M}[\bar{q}] \textbf{ do } S \textbf{ end}\ \{N\}}$$

As our ranking assertions are essentially *linear* functions on $\mathcal{D}(\mathcal{H})$, they normally have a more compact representation, and hopefully are easier to use in applications than the bounded functions in [Ying 2012].

Again, we can prove the soundness and (relative) completeness of the proof system for total correctness.

Theorem 5.3. *The proof system for total correctness is both sound and complete with respect to the total correctness of cq-programs.*

To conclude this section, let us point out an alternative statement for the (WhileT) rule.

Lemma 5.4. *Let $\Theta \in \mathcal{A}_V$. The loop* **while** $b$ **do** $S$ **end** *has $\Theta$-ranking assertions iff there is an increasing sequence $\{\Psi_n : n \geq 0\}$ of cq-assertions in $\mathcal{A}_V$ such that*

(1) $\Theta \sqsubseteq \top_V - \Psi_0$ *and* $\bigvee_n \Psi_n = \top_V$;
(2) $\models_{par} \{b \wedge \Psi_{n+1}\}\ S\ \{\Psi_n\}$.

With the above lemma, we can restate rule (WhileT) as follows:

$$\text{(WhileT}') \quad \frac{\begin{array}{l}\{b \wedge \Theta\}\ S\ \{\Theta\}\\ \{\Theta_n : n \geq 0\} \text{ increasing}, \Theta \sqsubseteq \top_V - \Theta_0,\ \bigvee_n \Theta_n = \top_V\\ \vdash_{par} \{b \wedge \Theta_{n+1}\}\ S\ \{\Theta_n\}\end{array}}{\{\Theta\}\ \textbf{while } b \textbf{ do } S \textbf{ end}\ \{\neg b \wedge \Theta\}}$$

Interestingly, proof of partial correctness is also employed in this new rule for total correctness. Note that however, there are infinitely many premises in the rule which might not be convenient for automated reasoning, unless parametrised reasoning is supported somehow.

## 6 AUXILIARY RULES

We have provided sound and relatively complete proof systems for both partial and total correctness of cq-programs. Thus in principle, these proof rules are sufficient for proving desired properties as long as they can be described faithfully with Hoare triple formulas. However, in practice, using these rules directly might be complicated. To simplify reasoning, in this section we introduce some auxiliary proof rules which are listed in Table 5. For the sake of convenience, we write $\langle p, |\psi\rangle\rangle$ for $\langle p, |\psi\rangle\langle\psi|\rangle$, and $p$ for $p \wedge \top$.

The rules (Top) and (Bot) deals with special cq-assertions. Rules (Meas0), (Init0), and (Unit0) simplify the corresponding ones in Table 4 when the evolved quantum variables do not appear in the postcondition. Extended commands with syntactic sugars are also considered in these rules, as well as in the rule (Param).

Rule (SupOper) essentially says that any valid operation applied on the quantum variables not involved in $S$ does not affect the correctness of $S$. Note that a weaker version of this rule, where $V$ and $W$ are taken equal, was presented in [Ying 2019]. However, the current version is much more expressive, evidenced by the fact that (SupPos), (Tens), and (Trace) are just its special cases. Rule (SupPos) deals with superposition of quantum states, and it is useful in proving the correctness of quantum circuits which consist of solely unitary operators. A natural way to verify such circuits is to do so for each quantum pure state from an orthonormal basis. Specifically, let $V = qv(S)$ and

(Top) $\{\Theta\}\ S\ \{\top_V\}$ (Bot) $\{\bot_V\}\ S\ \{\Psi\}$ (Init0) $\dfrac{\bar{q}\cap qv(\Theta)=\emptyset}{\{\Theta\}\ \bar{q}:=0\ \{\Theta\}}$

(Meas0) $\dfrac{\bar{q}\cap qv(\Theta)=\emptyset}{\{\sum_{i\in J}\Theta[i/x]\otimes M_i^\dagger M_i\}\ x:=\textbf{meas}\ \mathcal{M}[\bar{q}]\ \{\Theta\}}$ (Unit0) $\dfrac{\bar{q}\cap qv(\Theta)=\emptyset}{\{\Theta\}\ \bar{q}\ *\!=U\ \{\Theta\}}$

(Param) $\left\{\sum_{i_1,\cdots,i_k=1,distinct}^{|\bar{q}|}\sum_{i=1}^{K}\bigwedge_{j=1}^{k}(e_j=i_j)\wedge(e=i)\wedge\mathcal{U}^i_{i_1,\cdots,i_k}(\Theta)\right\}\bar{q}[e_1,\cdots,e_k]\ *\!=U(e)\ \{\Theta\}$

where $\mathcal{U}^i_{i_1,\cdots,i_k}(\Theta)\triangleq U(i)^\dagger_{\bar{r}}\Theta U(i)_{\bar{r}}$. and $\bar{r}\triangleq q_{i_1},\cdots,q_{i_k}$.

(SupOper) $$\frac{\{\Theta\}\ S\ \{\Psi\},\ (V\cup W)\cap qv(S)=\emptyset}{\{\mathcal{F}_{V\to W}(\Theta)\}\ S\ \{\mathcal{F}_{V\to W}(\Psi)\}}$$

where $\mathcal{F}_{V\to W}$ is a completely positive and sub-unital super-operator from $\mathcal{H}_V$ to $\mathcal{H}_W$.

(SupPos) $$\frac{\{\langle p,\sum_{i\in I}|i\rangle_V|i\rangle_W\rangle\}\ S\ \{\langle p',\sum_{i\in I}|\psi_i\rangle_V|i\rangle_W\rangle\},qv(S)\subseteq V,V\simeq W}{\{\langle p,\sum_{i\in I}\alpha_i|i\rangle_V\rangle\}\ S\ \{\langle p',\sum_{i\in I}\alpha_i|\psi_i\rangle_V\rangle\}}$$

where $\{|i\rangle : i\in I\}$ is an orthonormal basis, $\alpha_i\in\mathbb{C}$ and $\sum_{i\in I}|\alpha_i|^2=1$.

(Tens) $\dfrac{\{\Theta\}\ S\ \{\Psi\},\ \ W\cap qv(S,\Theta,\Psi)=\emptyset}{\{M_W\otimes\Theta\}\ S\ \{M_W\otimes\Psi\}}$ (Trace) $\dfrac{\{\Theta\}\ S\ \{\Psi\},\ \ V\subseteq qv(\Theta)\cap qv(\Psi)}{\{\frac{1}{\dim(\mathcal{H}_V)}\mathrm{tr}_V(\Theta)\}\ S\ \{\frac{1}{\dim(\mathcal{H}_V)}\mathrm{tr}_V(\Psi)\}}$

(Exist) $\dfrac{\{\langle p,M\rangle\}\ S\ \{\Psi\},\ x\notin var(S)\cup free(\Psi)}{\{\langle\exists x.p,M\rangle\}\ S\ \{\Psi\}}$ (Inv) $\dfrac{\{\Theta\}\ S\ \{\Psi\},free(p)\cap change(S)=\emptyset}{\{p\wedge\Theta\}\ S\ \{p\wedge\Psi\}}$

(Disj) $\dfrac{\{\langle p,M\rangle\}\ S\ \{\Psi\},\ \{\langle p',M\rangle\}\ S\ \{\Psi\}}{\{\langle p\vee p',M\rangle\}\ S\ \{\Psi\}}$ (Sum) $\dfrac{\{\langle p,M\rangle\}\ S\ \{\Psi\},\ \{\langle p',N\rangle\}\ S\ \{\Psi\},\ p'\to\neg p}{\{\langle p,M\rangle+\langle p',N\rangle\}\ S\ \{\Psi\}}$

(L-sum) $\dfrac{\{\Theta_i\}\ S\ \{\Psi_i\},\ \lambda_i\geq 0}{\{\sum_i\lambda_i\Theta_i\}\ S\ \{\sum_i\lambda_i\Psi_i\}}$ (ProbComp) $\dfrac{\{p'\}\ S_1\ \{\langle p,|\psi\rangle_{\bar{q}}\langle\psi|\rangle\},\{\langle p,M_{\bar{q}}\rangle\}\ S_2\ \{\Psi\}}{\{\langle\psi|M|\psi\rangle\cdot p'\}\ S_1;S_2\ \{\Psi\}}$

(C-WhileT) $$\frac{\{b\wedge\Theta\}\ S\ \{\Theta\},\ \{b\wedge p\wedge t=z\}\ S\ \{t<z\},\ p\to t\geq 0}{\{\Theta\}\ \textbf{while}\ b\ \textbf{do}\ S\ \textbf{end}\ \{\neg b\wedge\Theta\}}$$

where $type(z)=type(t)=\textbf{Integer}$, $z\notin var(p,b,t,S)$, $\Theta=\bigoplus_{i\in I}\langle p_i,M_i\rangle$ and $p\triangleq\bigvee_{i\in I}p_i$.

Table 5. Auxiliary rules.

$\{|i\rangle : i\in I\}$ be an orthonormal basis of $\mathcal{H}_V$. We may expect that $\vdash\{\langle p,|i\rangle_V\rangle\}\ S\ \{\langle p',|\psi_i\rangle_V\rangle\}$ for all $i\in I$ implies

$$\vdash\left\{\langle p,\sum_{i\in I}\alpha_i|i\rangle_V\rangle\right\}S\left\{\langle p',\sum_{i\in I}\alpha_i|\psi_i\rangle_V\rangle\right\}$$

for any superposed state $\sum_{i\in I}\alpha_i|i\rangle$. This is, however, not correct. For example, let $type(q) = \mathbf{Qubit}$. We have

$$\{\langle \mathbf{true}, |0\rangle_q\rangle\}\ q *= Z\ \{\langle \mathbf{true}, |1\rangle_q\rangle\} \quad \text{and} \quad \{\langle \mathbf{true}, |1\rangle_q\rangle\}\ q *= Z\ \{\langle \mathbf{true}, |1\rangle_q\rangle\}$$

since $Z|1\rangle = -|1\rangle$ and and $(-|1\rangle)(-\langle 1|) = |1\rangle\langle 1|$. However, $\{\langle \mathbf{true}, |+\rangle_q\rangle\}\ q *= Z\ \{\langle \mathbf{true}, |+\rangle_q\rangle\}$ is certainly not true. The reason is that observables (thus cq-assertions) cannot distinguish quantum states which differ only in the global phases. When taking the superposition of them with other states, global phases become local ones, and the superposed states may be distinguishable. To overcome this difficulty, in rule (SupPos) we combine all the basis states $|i\rangle_V$ into an (unnormalised) maximally entangled state $\sum_{i\in I}|i\rangle_V|i\rangle_W$ in a larger Hilbert space with $V \simeq W$ (there exists a one-to-one correspondence between $V$ and $W$, with the corresponding quantum variables being the same type). In this way, all the global phases caused by applying $S$ on $|i\rangle_V$'s are preserved.

Rules (Exist) and (Inv) are merely classical ones where the logic operators are performed on the classical part of the cq-assertions. The three rules (Disj), (Sum), and (L-sum) all extend the rule

$$\frac{\{p_1\}\ S\ \{p_1'\},\ \{p_2\}\ S\ \{p_2'\}}{\{p_1 \vee p_2\}\ S\ \{p_1' \vee p_2'\}}$$

in classical Hoare logic dealing with disjunctions of assertions. In the first two rules, the disjunction is applied only on the classical part: rule (Disj) allows disjunction of any classical assertions $p$ and $p'$, but their quantum part must be the same; rule (Sum) allows different quantum parts, but the classical assertions must be disjoint. For the general case, a weighted sum (for both the pre- and the postconditions) is used in (L-sum).

Rule (ProbComp) reasons about sequential composition of two programs $S_1$ and $S_2$. Note that rule (Seq) in Table 4 assumes the postcondition of $S_1$ is the same as, or stronger than, when (Imp) is employed, the precondition of $S_2$. In contrast, rule (ProbComp) can handle the case where such an assumption does not hold. As can be seen from the case studies, this rule is very useful in calculating the success probability of quantum algorithms.

Finally, we present rule (C-WhileT) for the special case when a classical ranking function can be found to guarantee the (finite) termination of cq-programs. As shown in the case studies in Appendix, this rule is useful in simplifying the analysis of many practical quantum algorithms.

Lemma 6.1. (1) *All the auxiliary rules presented in Table 5, except (Top), are sound with respect to total correctness.*

(2) *If we require $\sum_i \lambda_i \leq 1$ in (L-sum), then all the auxiliary rules, except (ProbComp) and (C-WhileT), presented in Table 5 are sound with respect to partial correctness.*

## 7 CASE STUDIES

To illustrate the effectiveness of the proposed proof systems, we employ them to verify Grover's search algorithm presented in Example 4.5 and Shor's factorisation algorithm.

### 7.1 Grover's serach algorithm

We have proved in Examples 4.5 and 4.11, by employing the denotational semantics and the definition of correctness formulas respectively, that Grover's algorithm succeeds in finding a desired solution with probability $p_{succ}$ in Eq.(3). We now re-prove this result using the proof system for total correctness. As stated in Example 4.11, the goal is to show

$$\vdash_{tot} \{p_{succ}\}\ Grover\ \{y \in Sol\}. \tag{8}$$

Let $b \triangleq x < K$ and $\Theta \triangleq \sum_{k=0}^{K} \langle x = k, \Psi_{K-k} \rangle$, where $\Psi_k = |\psi_k\rangle\langle\psi_k|$ and

$$|\psi_k\rangle = \cos\left(\frac{\pi}{2} - k\theta\right)|\alpha\rangle + \sin\left(\frac{\pi}{2} - k\theta\right)|\beta\rangle.$$

Note that $|\psi_0\rangle = |\beta\rangle$ and $G|\psi_k\rangle = |\psi_{k-1}\rangle$. Intuitively, $\Theta$ records the quantum states at each iteration, and we show that it serves as an invariant of the while-loop in Grover's algorithm. Observe from (Unit) and (Assn) that

$$\vdash_{tot} \left\{\sum_{k=0}^{K-1} \langle x = k, \Psi_{K-k} \rangle\right\} \bar{q} *= G;\ x := x + 1; \left\{\sum_{k=0}^{K-1} \langle x = k+1, \Psi_{K-k-1} \rangle\right\}$$

Together with the fact

$$\sum_{k=0}^{K-1} \langle x = k+1, \Psi_{K-k-1} \rangle = \sum_{k=1}^{K} \langle x = k, \Psi_{K-k} \rangle \sqsubseteq \Theta,$$

we deduce from rule (Imp) that $\vdash_{tot} \{b \wedge \Theta\}\ \bar{q} *= G; x := x + 1;\ \{\Theta\}$. Let $z \notin \{x, y\}$, $t \triangleq K - x$, and $p \triangleq (0 \le x \le K)$. Then $p \to t \ge 0$. Thus by rule (C-WhileT)

$$\vdash_{tot} \{\Theta\}\ \textbf{while}\ b\ \textbf{do}\ \bar{q} *= G; x := x + 1;\ \textbf{end}\ \{\neg b \wedge \Theta\}. \tag{9}$$

Furthermore, we have

$$\begin{aligned}
&\left\{|\langle +|^{\otimes n}|\psi_K\rangle|^2\right\} \\
&\bar{q} := \mathbf{0};\ \bar{q} *= H^{\otimes n}; \\
&\{\langle \mathbf{true}, \Psi_K \rangle\} && (\mathit{Init}, \mathit{Unit}) \\
&x := 0; \\
&\left\{\sum_{k=0}^{K} \langle x = k, \Psi_{K-k} \rangle\right\} && (\mathit{Assn}, \mathit{Imp}) \\
&\textbf{while}\ b\ \textbf{do}\ \bar{q} *= G; x := x + 1;\ \textbf{end} \\
&\{\{\langle x = K, \Psi_0 \rangle\}\} && \mathit{Eq}.(9) \\
&\{\langle \mathbf{true}, \sum_{i \in Sol} |i\rangle\langle i| \rangle\} && (\mathit{Imp}) \\
&y := \mathbf{meas}\ \bar{q} \\
&\{y \in Sol\} && (\mathit{Meas0})
\end{aligned}$$

Finally, it is easy to show that $|\langle +|^{\otimes n}|\psi_K\rangle|^2 = p_{succ}$, from which Eq.(8) follows.

### 7.2 Shor's factorisation algorithm

Given a positive integer $N$ which is composite, the factorisation problem asks to find all the factors of $N$. When $N$ is sufficiently large, no classical algorithm can solve this problem in polynomial (in $\log(N)$, the number of bits to encode $N$) time. The difficulty of this problem is at the heart of widely used cryptographic algorithms such as RSA [Rivest et al. 1978].

One of the killer apps of quantum computing is Shor's algorithm [Shor 1994], which solves the factorisation problem (actually, a polynomial-time equivalent one which finds a non-trivial prime factor of $N$) in $\log^3(N)$ time, achieving an exponential speed up over the best classical algorithms. Shor's algorithm uses the order-finding algorithm as a subroutine in an inline manner, and is depicted in Fig.6 (left column), where $\mathit{Unif}(2, N-1)$ is the uniform distribution over $\{2, \cdots, N-1\}$.

| | |
|---|---|
| $Shor(N) \triangleq$ | $\{p_{OF}(1-1/2^m)\cdot cmp(N)\}$ |
| $x :=_\$ Unif(2, N-1);$ | $x :=_\$ Unif(2, N-1);$ |
| **if** $gcd(x,N) > 1$ **then** | $\{cmp(N) \wedge \Theta\}$ |
| $y := gcd(x,N)$ | **if** $gcd(x,N) > 1$ **then** $\{F(gcd(x,N))\}$ |
| **else** | $y := gcd(x,N)\ \{F(y)\}$ |
| $OF(x,N);$ | **else** |
| **if** ($z$ is even $\wedge\ x^{z/2} \not\equiv_N -1$) **then** | $\{p_{OF}\cdot(cmp(N) \wedge E(r) \wedge gcd(x,N)=1)\}$ |
| $y_1 := gcd(x^{z/2}-1, N);$ | $OF(x,N);$ |
| $y_2 := gcd(x^{z/2}+1, N)$ | $\{cmp(N) \wedge E(r) \wedge z = r\}$ Eq.(10), (Inv) |
| **else** | **if** $E(z)$ **then** |
| **abort** | $\{F(gcd(x^{z/2}-1,N)) \vee F(gcd(x^{z/2}+1,N))\}$ Eq.(11) |
| **end** | $y_1 := gcd(x^{z/2}-1, N);$ |
| **if** ($y_1$ non-trivial factor of $N$) **then** | $y_2 := gcd(x^{z/2}+1, N)$ |
| $y := y_1$ | **else** $\{\bot\}$ **abort end** |
| **else** | $\{F(y_1) \vee F(y_2)\}$ |
| **if** ($y_2$ non-trivial factor of $N$) **then** | **if** $F(y_1)$ **then** |
| $y := y_2$ | $\{F(y_1)\}\ y := y_1$ |
| **else** | **else** $\{F(y_2)\}$ **if** $F(y_2)$ **then** |
| **abort** | $\{F(y_2)\}\ y := y_2$ |
| **end** | **else** $\{\bot\}$ **abort end** |
| **end** | **end** $\{F(y)\}$ |
| **end** | **end** $\{F(y)\}$ |

Table 6. Shor's factorisation algorithm and its proof outline.

The detailed description of the order-finding algorithm $OF(x,N)$ can be found in the Appendix, where we show that

$$\vdash_{tot} \{p_{OF}\cdot(gcd(x,N)=1)\}OF(x,N)\{z=r\} \tag{10}$$

for some constant $p_{OF} \in (0,1]$, where $r = r(x) \triangleq ord(x,N)$ be the order of $x$ modulo $N$. Recall that given positive co-prime positive integers $x$ and $N$ with $x < N$, the order of $x$ modulo $N$ is the least positive integer $r$ such that $x^r \equiv_N 1$ where $\equiv_N$ denotes equality modulo $N$. Note also that $change(OF(x,N)) = \{z', z\}$.

Let $F(y) \triangleq (y$ is a non-trivial factor of $N) \equiv [1 < y < N \wedge (y\ div\ N)]$, and $cmp(N) \triangleq N > 0 \wedge N$ is composite. Then the correctness of $Shor(N)$ can be stated as

$$\{p_{Shor}\cdot cmp(N)\}\ Shor(N)\ \{F(y)\}$$

for some constant success probability $p_{Shor} \in (0,1]$. As the only non-classical part of Shor's algorithm is the order-finding subroutine, it can be verified by simply employing theorems from number theory. To be specific, let $E(z) \triangleq (z$ is even $\wedge\ x^{z/2} \not\equiv_N -1)$. The following lemma is crucial:

Lemma 7.1 ([Nielsen and Chuang 2002]). *Let $N > 0$ be a composite integer.*

(1) *If $s$ is a non-trivial solution to the equation $s^2 \equiv_N 1$, then at least one of $gcd(s-1,N)$ and $gcd(s+1,N)$ is a non-trivial factor of $N$.*
(2) *Let $m$ be the number of prime factors of $N$. If $x$ is chosen uniformly at random from the set $\{x : 1 < x < N\}$. Then the conditional probability*

$$\Pr[E(r) \mid gcd(x,N) = 1] \geq 1 - \frac{1}{2^m}.$$

Note that $E(r)$ implies $x^{r/2} \not\equiv_N 1$. Thus the first clause of Lemma 7.1 says

$$cmp(N) \wedge E(r) \longrightarrow F(gcd(x^{r/2}-1,N)) \vee F(gcd(x^{r/2}+1,N)). \tag{11}$$

Furthermore, let $x$ be chosen uniformly at random from the set $\{n : 1 < n < N\}$ and $\lambda \triangleq \Pr(gcd(x, N) > 1)$. Then from the second clause of Lemma 7.1 we have

$$\begin{aligned}&\Pr[gcd(x,N) > 1] + p_{OF} \cdot \Pr[gcd(x,N) = 1 \wedge E(r)] \\ &\geq \lambda + p_{OF} \cdot (1-\lambda)(1 - \frac{1}{2^m}) \geq p_{OF} \cdot (1 - \frac{1}{2^m}).\end{aligned}$$

Note that each $n$ from 2 to $N-1$ is taken with probability $\frac{1}{N-2}$. We have

$$\begin{aligned}&\{p_{OF}(1 - 1/2^m)\} \\ &\left\{\frac{1}{N-2}\sum_{n=2}^{N-1} gcd(n,N) > 1 \oplus p_{OF} \cdot (gcd(n,N) = 1 \wedge E(r(n)))\right\} \\ &x :=_{\$} Unif(2, N-1) \\ &\{gcd(x,N) > 1 \oplus p_{OF} \cdot (gcd(x,N) = 1 \wedge E(r))\}. \qquad (Rassn)\end{aligned}$$

The rest of the proof is sketched in the right column of Fig. 6, where $\Theta \triangleq [gcd(x,N) > 1 \oplus p_{OF} \cdot (gcd(x,N) = 1 \wedge E(r))]$.

## 8 CONCLUSION

We studied in this paper a simple quantum while-language where (infinite type) classical variables are explicitly involved. This language supports deterministic and probabilistic assignments of classical variables; initialisation, unitary transformation, and measurements of quantum variables; conditionals and while loops. Simultaneous initialisation of multiple quantum variables, and application of parametrised unitary operations on parametrised variables are also supported as syntactic sugars. These features make the description of practical quantum algorithms easy and compact, as shown by various examples.

With novel definition of cq-states and assertions, we defined for our language a small-step structural operational semantics, and based on it, a denotational one. Partial and total correctness of cq-programs are then introduced in the form of Hoare triples. We proposed Hoare-type logic systems for partial and total correctness respectively, and showed their soundness and relative completeness. Case studies including Grover's algorithm, quantum Fourier transformation, phase estimation, order finding, and Shor's algorithm illustrate the expressiveness of our language as well as the capability of the Hoare logic.

As future work, we would like to develop a software tool to implement the proof systems proposed in this paper, and use it to analyse more quantum algorithms and protocols from the area of quantum computation and communication. Another direction we are going to pursue is to enhance the expressiveness of our assertion language to include, say, join and meet of cq-assertions. At the current form, these logic operations can only be done either at the classical side in the form of, say $p \wedge \Theta$, or by taking the linear-sum of cq-assertions.

## ACKNOWLEDGMENTS

This work is partially supported by the National Key R&D Program of China (Grant No: 2018YFA0306701) and the Australian Research Council (Grant No: DP180100691). Y. F. also acknowledges the support of Center for Quantum Computing, Peng Cheng Laboratory, Shenzhen during his visit.

## REFERENCES

Krzysztof Apt, Frank S De Boer, and Ernst-Rüdiger Olderog. 2010. *Verification of sequential and concurrent programs.* Springer Science & Business Media.

Krzysztof R Apt and Ernst-Rüdiger Olderog. 2019. Fifty years of Hoare's logic. *Formal Aspects of Computing* 31, 6 (2019), 751–807.

Gilles Barthe, Thomas Espitau, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2018. An Assertion-Based Program Logic for Probabilistic Programs. In *European Symposium on Programming*. Springer, Cham, 117–144.

Gilles Barthe, Benjamin Grégoire, and Santiago Zanella Béguelin. 2009. Formal certification of code-based cryptographic proofs. In *Proceedings of the 36th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. 90–101.

Gilles Barthe, Justin Hsu, Mingsheng Ying, Nengkun Yu, and Li Zhou. 2019. Relational proofs for quantum programs. *Proceedings of the ACM on Programming Languages* 4, POPL (2019), 1–29.

Gilles Barthe, Boris Köpf, Federico Olmedo, and Santiago Zanella Béguelin. 2012. Probabilistic relational reasoning for differential privacy. In *Proceedings of the 39th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. 97–110.

Charles H Bennett. 1992. Quantum cryptography using any two nonorthogonal states. *Physical review letters* 68, 21 (1992), 3121.

Charles H Bennett and Gilles Brassard. 1984. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of the International Conference on Computers, Systems and Signal Processing*.

Charles H Bennett, Gilles Brassard, Claude Crépeau, Richard Jozsa, Asher Peres, and William K Wootters. 1993. Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. *Physical review letters* 70, 13 (1993), 1895.

Charles H Bennett and Stephen J Wiesner. 1992. Communication via one-and two-particle operators on Einstein-Podolsky-Rosen states. *Physical review letters* 69, 20 (1992), 2881.

Nick Benton. 2004. Simple relational correctness proofs for static analyses and program transformations. *ACM SIGPLAN Notices* 39, 1 (2004), 14–25.

Stefano Bettelli, Tommaso Calarco, and Luciano Serafini. 2003. Toward an architecture for quantum programming. *The European Physical Journal D-Atomic, Molecular, Optical and Plasma Physics* 25, 2 (2003), 181–200.

Rohit Chadha, Luís Cruz-Filipe, Paulo Mateus, and Amílcar Sernadas. 2007. Reasoning about probabilistic sequential programs. *Theoretical Computer Science* 379, 1-2 (2007), 142–165.

Rohit Chadha, Paulo Mateus, and Amílcar Sernadas. 2006a. Reasoning about imperative quantum programs. *Electronic Notes in Theoretical Computer Science* 158 (2006), 19–39.

Rohit Chadha, Paulo Mateus, and Amílcar Sernadas. 2006b. Reasoning about states of probabilistic sequential programs. In *International Workshop on Computer Science Logic*. Springer, 240–255.

JI Den Hartog and Erik P de Vink. 2002. Verifying probabilistic programs using a Hoare like logic. *International journal of foundations of computer science* 13, 03 (2002), 315–340.

Ellie D'Hondt and Prakash Panangaden. 2006a. Quantum weakest preconditions. *Mathematical Structures in Computer Science* 16, 3 (2006), 429–451.

Ellie D'Hondt and Prakash Panangaden. 2006b. Quantum weakest preconditions. *Math. Struct. Comp. Sci.* 16 (6 2006), 429–451. Issue 03. https://doi.org/10.1017/S0960129506005251

Edsger Wybe Dijkstra, Edsger Wybe Dijkstra, Edsger Wybe Dijkstra, Etats-Unis Informaticien, and Edsger Wybe Dijkstra. 1976. *A discipline of programming*. Vol. 613924118. Prentice-Hall Englewood Cliffs.

Lov K Grover. 1996. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 212–219.

Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. 2009. Quantum algorithm for linear systems of equations. *Physical review letters* 103, 15 (2009), 150502.

Charles Antony Richard Hoare. 1969. An axiomatic basis for computer programming. *Commun. ACM* 12, 10 (1969), 576–580.

Shih-Han Hung, Kesha Hietala, Shaopeng Zhu, Mingsheng Ying, Michael Hicks, and Xiaodi Wu. 2019. Quantitative robustness analysis of quantum programs. *Proceedings of the ACM on Programming Languages* 3, POPL (2019), 1–29.

Yoshihiko Kakutani. 2009. A Logic for Formal Verification of Quantum Programs. *Lecture Notes in Computer Science* (2009), 79–93. https://doi.org/10.1007/978-3-642-10622-4_7

Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Federico Olmedo. 2018. Weakest precondition reasoning for expected runtimes of randomized algorithms. *Journal of the ACM (JACM)* 65, 5 (2018), 1–68.

Karl Kraus, Arno Böhm, John D Dollard, and WH Wootters. 1983. States, effects, and operations: fundamental notions of quantum theory. *Lecture notes in physics* 190 (1983).

Yangjia Li and Dominique Unruh. 2019. Quantum Relational Hoare Logic with Expectations. *arXiv preprint arXiv:1903.08357* (2019).

Junyi Liu, Bohua Zhan, Shuling Wang, Shenggang Ying, Tao Liu, Yangjia Li, Mingsheng Ying, and Naijun Zhan. 2019. Formal verification of quantum algorithms using quantum Hoare logic. In *International conference on computer aided*

*verification*. Springer, 187–207.

Dominic Mayers. 2001. Unconditional security in quantum cryptography. *Journal of the ACM (JACM)* 48, 3 (2001), 351–406.

Annabelle McIver, Carroll Morgan, and Charles Carroll Morgan. 2005. *Abstraction, refinement and proof for probabilistic systems*. Springer Science & Business Media.

Carroll Morgan, Annabelle McIver, and Karen Seidel. 1996. Probabilistic predicate transformers. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 18, 3 (1996), 325–353.

Michael A Nielsen and Isaac Chuang. 2002. Quantum computation and quantum information.

Bernhard Ömer. 1998. *A procedural formalism for quantum computing*. Master thesis.

Lyle Harold Ramshaw. 1979. *Formalizing the analysis of algorithms*. Technical Report. STANFORD UNIV CA DEPT OF COMPUTER SCIENCE.

Robert Rand. 2019. Verification logics for quantum programs. *arXiv preprint arXiv:1904.04304* (2019).

Ronald L Rivest, Adi Shamir, and Leonard Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* 21, 2 (1978), 120–126.

Jeff W Sanders and Paolo Zuliani. 2000. Quantum programming. In *International Conference on Mathematics of Program Construction*. Springer, 80–99.

Peter Selinger. 2004. Towards a quantum programming language. *Mathematical Structures in Computer Science* 14, 4 (2004), 527–586.

Peter W Shor. 1994. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*. IEEE, 124–134.

Dominique Unruh. 2019a. Quantum hoare logic with ghost variables. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE, 1–13.

Dominique Unruh. 2019b. Quantum relational Hoare logic. *Proceedings of the ACM on Programming Languages* 3, POPL (2019), 1–31.

John Von Neumann. 1955. *Mathematical Foundations of Quantum Mechanics*. Princeton University Press, Princeton, NJ.

Mingsheng Ying. 2012. Floyd–Hoare logic for quantum programs. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 33, 6 (2012), 1–49.

Mingsheng Ying. 2016. *Foundations of Quantum Programming*. Morgan Kaufmann.

Mingsheng Ying. 2019. Toward automatic verification of quantum programs. *Formal Aspects of Computing* 31, 1 (2019), 3–25.

Mingsheng Ying, Li Zhou, and Yangjia Li. 2018. Reasoning about parallel quantum programs. *arXiv preprint arXiv:1810.11334* (2018).

Li Zhou, Nengkun Yu, and Mingsheng Ying. 2019. An applied quantum Hoare logic. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*. 1149–1162.

# A APPENDIX

In this appendix, we give the missing proofs and elaborate more case studies.

## A.1 Missing proofs

Proof of Lemma 3.3. The result follows directly from the fact that for any $V \subseteq qVar$, $\mathcal{D}(\mathcal{H}_V)$ is an $\omega$-CPO under the Löwner order $\sqsubseteq_V$, with $\mathbf{0}_{\mathcal{H}_V}$ being its least element [Selinger 2004]. □

Proof of Lemma 3.9. We only prove the last clause; the others are easy from definitions. For simplicity, we assume $V = W$. Then

$$\begin{aligned}
\mathrm{Exp}(\Delta|_p \models \Theta) &= \sum_{\sigma \in \lceil \Delta|_p \rceil} \mathrm{tr}\left[\Theta(\sigma) \cdot \Delta|_p(\sigma)\right] \\
&= \sum_{\sigma \in \lceil \Delta \rceil, \sigma \models p} \mathrm{tr}\left[\Theta(\sigma) \cdot \Delta(\sigma)\right] \\
&= \sum_{\sigma \in \lceil \Delta \rceil} \mathrm{tr}\left[(p \wedge \Theta)(\sigma) \cdot \Delta(\sigma)\right] \\
&= \mathrm{Exp}(\Delta \models p \wedge \Theta)
\end{aligned}$$

where the third inequality comes from the fact that for any $\sigma \in \Sigma$, $(p \wedge \Theta)(\sigma) = \Theta(\sigma)$ if $\sigma \models p$, and $\mathbf{0}_{\mathcal{H}_W}$ otherwise. □

Proof of Lemma 3.10. We take the converse part of Clause (1) as an example. Suppose $\Delta \not\sqsubseteq \Delta'$. Then there exists a $\sigma \in \Sigma$ and $|\psi\rangle \in \mathcal{D}(\mathcal{H}_V)$ such that $\langle\psi|\Delta(\sigma)|\psi\rangle > \langle\psi|\Delta'(\sigma)|\psi\rangle$. If we can find a classical assertion $p$ which distinguishes $\sigma$ from other states in $\lceil\Delta'\rceil$. Then obviously the cq-assertion $\langle p, |\psi\rangle_V\langle\psi|\rangle$ serves as a counter-example for the assumption.

Note that the formula $p_* \triangleq \bigwedge_{x \in Var}(x = \sigma(x))$ uniquely determines $\sigma$. However, it is not a valid classical assertion, as the set $Var$ is infinite. To convert it to a finite conjunction, let $\epsilon \triangleq \langle\psi|\Delta(\sigma)|\psi\rangle - \langle\psi|\Delta'(\sigma)|\psi\rangle$. For this $\epsilon$, there exists a finite subset $A$ of $\lceil\Delta'\rceil$ such that $\mathrm{tr}(\Delta'|_A) > \mathrm{tr}(\Delta') - \epsilon$. For any $\sigma' \in A$ with $\sigma' \neq \sigma$, there exists $x_{\sigma'} \in Var$ such that $\sigma'(x_{\sigma'}) \neq \sigma(x_{\sigma'})$. Now let $X \triangleq \{x_{\sigma'} : \sigma' \in A, \sigma' \neq \sigma\}$. Then the classical assertion $p \triangleq \bigwedge_{x \in X}(x = \sigma(x))$ distinguishes $\sigma$ from other states in $A$. Finally, let $\Theta \triangleq \langle p, |\psi\rangle_V\langle\psi|\rangle$. Then $\mathrm{Exp}(\Delta'|_A \models \Theta) = \langle\psi|\Delta'(\sigma)|\psi\rangle$, and $\mathrm{Exp}(\Delta' - \Delta'|_A \models \Theta) \leq \mathrm{tr}(\Delta' - \Delta'|_A) < \epsilon$. Thus

$$\mathrm{Exp}(\Delta \models \Theta) \geq \langle\psi|\Delta(\sigma)|\psi\rangle = \langle\psi|\Delta'(\sigma)|\psi\rangle + \epsilon > \mathrm{Exp}(\Delta' \models \Theta),$$

contradicting the assumption. □

Proof of Lemma 3.11. We prove (1) as an example; the others are similar. Let $\Delta^* \triangleq \bigvee_{n \geq 0} \Delta_n$. First, from the fact that $\Delta_n \sqsubseteq \Delta^*$ for all $n$, $\mathrm{Exp}(\Delta^* \models \Theta) \geq \sup_{n \geq 0} \mathrm{Exp}(\Delta_n \models \Theta)$ by Lemma 3.10(1). Furthermore, for any $n$,

$$\begin{aligned}
\mathrm{Exp}(\Delta^* \models \Theta) - \mathrm{Exp}(\Delta_n \models \Theta) &= \mathrm{Exp}(\Delta^* - \Delta_n \models \Theta) \\
&\leq \mathrm{Exp}(\Delta^* - \Delta_n \models \top_{\mathcal{H}}) \\
&= \mathrm{tr}(\Delta^* - \Delta_n) = \mathrm{tr}(\Delta^*) - \mathrm{tr}(\Delta_n),
\end{aligned}$$

where the first and second equalities are from Lemma 3.9, and the first inequality from Lemma 3.10(2). Thus $\mathrm{Exp}(\Delta^* \models \Theta) = \sup_{n \geq 0} \mathrm{Exp}(\Delta_n \models \Theta)$ from the fact that $\mathrm{tr}(\Delta^*) = \sup_n \mathrm{tr}(\Delta_n)$. □

Proof of Lemma 3.12. Note that $\sigma \models p[e/x]$ iff $\sigma[\sigma(e)/x] \models p$ for any classical state $\sigma$ and assertion $p$. Let $\Delta = \bigoplus_{i\in I}\langle \sigma_i, \rho_i\rangle$ and $\Theta = \bigoplus_{j\in J}\langle p_j, M_j\rangle$. Then

$$\begin{aligned}\mathrm{Exp}(\Delta \models \Theta[e/x]) &= \sum_{i\in I}\sum_{j\in J,\sigma_i\models p_j[e/x]} \mathrm{tr}(M_j\rho_i)\\ &= \sum_{i\in I}\sum_{j\in J,\sigma_i[\sigma_i(e)/x]\models p_j} \mathrm{tr}(M_j\rho_i),\end{aligned}$$

which is exactly $\mathrm{Exp}(\Delta[e/x] \models \Theta)$. Here we assume that $qv(\Delta) = qv(\Theta)$; the general case can be proved similarly. □

Proof of Lemma 4.2. Note that discrete probability distributions have countable image set. The first clause is easy by induction. The second one is directly from the fact that any configuration with the form $\langle E, \sigma', \rho'\rangle$ has no further transition. □

Proof of Lemma 4.4. Clause (2) is easy. For (1), we prove by induction on $n$ that $\mathrm{tr}(\Delta_n) \le \mathrm{tr}(\rho)$ whenever $\Delta = \langle \sigma, \rho\rangle$ and $\Delta_n$ is defined as in Lemma 4.2(2). Thus the result holds for simple cq-states. The general case follows easily. □

Proof of Lemma 4.6. We only prove (10) as an example; the others are simpler. For any $\langle \sigma, \rho\rangle \in \mathcal{S}_V$ with $V \supseteq qv(\textbf{while})$, let $\Pi$ be the set of all terminating computations of **while** starting in $\langle \sigma, \rho\rangle$. Furthermore, let $\Pi_0 \triangleq \emptyset$, and for $n \ge 1$ let

$$\Pi_n \triangleq \{\pi \in \Pi : \#\{i : prog(\pi[i]) = \textbf{while}\} \le n\}$$

be the set of computations in $\Pi$ in which the loop has iterated for no more than $n$ times before termination. Here $prog(\pi[i])$ is the program (the first component) of the $i$-th configuration of $\pi$. Obviously, $\Pi = \bigcup_{n\ge 0}\Pi_n$ and

$$[\![\textbf{while}]\!](\sigma,\rho) = \sum_{\pi\in\Pi}\langle \sigma_\pi, \rho_\pi\rangle = \bigvee_{n\ge 0}\sum_{\pi\in\Pi_n}\langle \sigma_\pi, \rho_\pi\rangle$$

where we assume each computation $\pi \in \Pi$ ends with $\langle E, \sigma_\pi, \rho_\pi\rangle$. The result then follows from the fact that

$$[\![(\textbf{while})^n]\!](\sigma,\rho) = \sum_{\pi\in\Pi_n}\langle \sigma_\pi, \rho_\pi\rangle$$

which is easy to observe. □

Proof of Lemma 4.10. This lemma is easy from linearity of $[\![S]\!]$ for any cq-program $S$. □

Proof of Lemma 4.14. We prove this lemma by induction on the structure of $S$. The basis cases are easy from the definition. We only show the following two cases for $wp$ as examples.

- Let $S \triangleq$ **if** $b$ **then** $S_1$ **else** $S_0$ **end**. Then

$$\begin{aligned}\mathrm{Exp}(\Delta \models wp.S.\Theta) &= \mathrm{Exp}(\Delta \models b \wedge wp.S_1.\Theta + \neg b \wedge wp.S_0.\Theta)\\ &= \mathrm{Exp}(\Delta|_b \models wp.S_1.\Theta) + \mathrm{Exp}(\Delta|_{\neg b} \models wp.S_0.\Theta)\\ &= \mathrm{Exp}([\![S_1]\!](\Delta|_b) \models \Theta) + \mathrm{Exp}([\![S_0]\!](\Delta|_{\neg b}) \models \Theta)\end{aligned}$$

  which is exactly $\mathrm{Exp}([\![S]\!](\Delta) \models \Theta)$.
- Let $S \triangleq$ **while** $b$ **do** $S'$ **end**. Let $\Theta_0 \triangleq \bot_V$, and for any $n \ge 0$, $\Theta_{n+1} \triangleq \neg b \wedge \Theta + b \wedge wp.S'.\Theta_n$. First, we show by induction that for any $n \ge 0$ and $\Delta' \in \mathcal{S}_V$,

$$\mathrm{Exp}(\Delta' \models \Theta_n) = \mathrm{Exp}([\![S^n]\!](\Delta') \models \Theta).$$

The case of $n = 0$ follows from the definition. We further calculate from Lemmas 3.9 and 4.7 that

$$\begin{aligned}\mathrm{Exp}(\Delta' \models \Theta_{n+1}) &= \mathrm{Exp}(\Delta' \models \neg b \wedge \Theta) + \mathrm{Exp}(\Delta' \models b \wedge wp.S'.\Theta_n)\\ &= \mathrm{Exp}(\Delta'|_{\neg b} \models \Theta) + \mathrm{Exp}(\Delta'|_b \models wp.S'.\Theta_n)\\ &= \mathrm{Exp}(\Delta'|_{\neg b} \models \Theta) + \mathrm{Exp}([\![S']\!](\Delta'|_b) \models \Theta_n)\\ &= \mathrm{Exp}(\Delta'|_{\neg b} \models \Theta) + \mathrm{Exp}([\![S^n]\!]([\![S']\!](\Delta'|_b)) \models \Theta))\\ &= \mathrm{Exp}([\![S^{n+1}]\!](\Delta') \models \Theta).\end{aligned}$$

Thus from Lemma 3.11,

$$\mathrm{Exp}(\Delta \models wp.S.\Theta) = \mathrm{Exp}(\Delta \models \bigvee_{n\geq 0} \Theta_n) = \mathrm{Exp}([\![S]\!](\Delta) \models \Theta).$$

□

Proof of Lemma 4.17. Note that $\models_{tot} \{\Theta\}\ S\ \{\Psi\}$ iff

$$\models_{tot} \{\Theta \otimes I_{qv(S,\Psi)\setminus qv(\Theta)}\}\ S\ \{\Psi \otimes I_{qv(S,\Theta)\setminus qv(\Psi)}\}.$$

Similar result holds for partial correctness as well. Then the lemma follows from Lemma 4.16(2). □

Proof of Lemma 5.4. Let $\{\Theta_n : n \geq 0\}$ be a sequence of $\Theta$-ranking assertions for **while** $b$ **do** $S$ **end**. Let $\Psi_i \triangleq \top_V - \Theta_i$, $i \geq 0$. Then clause (1) holds trivially. To prove clause (2), note that

$$\begin{aligned} & b \wedge wp.S.\Theta_n \sqsubseteq \Theta_{n+1}\\ \text{iff}\quad & wp.S.\Theta_n \sqsubseteq b \wedge \Theta_{n+1} + \neg b \wedge \top_V\\ \text{iff}\quad & b \wedge (\top_V - \Theta_{n+1}) \sqsubseteq \top_V - wp.S.\Theta_n\\ \text{iff}\quad & b \wedge \Psi_{n+1} \sqsubseteq wlp.S.\Psi_n\end{aligned}$$

where the last equivalence is from Lemma 4.16(1). □

Proof of Theorem 5.1. Soundness: We need only to show that each rule in Table 4 is valid in the sense of partial correctness. Take the rule (While) as an example; the others are simpler. Let $\models_{par} \{b \wedge \Theta\}\ S\ \{\Theta\}$. Without loss of generality, we assume $qv(S) \subseteq qv(\Theta)$. Then $b \wedge \Theta \sqsubseteq wlp.S.\Theta$. We now prove by induction on $n$ that $\Theta \sqsubseteq \Theta_n$ for any $n \geq 0$, where $\Theta_n$ is defined as in Table 4 for the $wlp$ semantics of **while** b **do** S **end**. The case when $n = 0$ is trivial. Then we calculate

$$\begin{aligned}\Theta_{n+1} &= \neg b \wedge \Theta + b \wedge wlp.S.\Theta_n\\ &\sqsupseteq \neg b \wedge \Theta + b \wedge wlp.S.\Theta\\ &\sqsupseteq \neg b \wedge \Theta + b \wedge b \wedge \Theta = \Theta,\end{aligned}$$

where the first inequality follows from the induction hypothesis and Lemma 4.16(3). Thus

$$\Theta \sqsubseteq wlp.(\textbf{while}\ b\ \textbf{do}\ S\ \textbf{end}).(\neg b \wedge \Theta),$$

and so

$$\models_{par} \{\Theta\}\ \textbf{while}\ b\ \textbf{do}\ S\ \textbf{end}\ \{\neg b \wedge \Theta\}$$

as desired.

Completeness: By Lemma 4.17 and the (Imp) rule, it suffices to show that for any $\Theta$ and $S'$ with $qv(S') \subseteq qv(\Theta)$,

$$\vdash_{par} \{wlp.S'.\Theta\}\ S'\ \{\Theta\}.$$

Again, we take the case for loops as an example. Let **while** $\triangleq$ **while** $b$ **do** $S$ **end** and $\Psi \triangleq wlp.\textbf{while}.\Theta$. By induction, we have $\vdash_{par} \{wlp.S.\Psi\}\ S\ \{\Psi\}$. Note that

$$\Psi = \neg b \wedge \Theta + b \wedge wlp.S.\Psi.$$

Thus $b\wedge\Psi = b\wedge wlp.S.\Psi \sqsubseteq wlp.S.\Psi$ and so $\vdash_{par} \{b\wedge\Psi\}\ S\ \{\Psi\}$ by the (Imp) rule. Now using (While) we have $\vdash_{par} \{\Psi\}$ **while** $\{\neg b\wedge\Psi\}$ and the result follows from the fact that $\neg b\wedge\Psi = \neg b\wedge\Theta \sqsubseteq \Theta$. □

Proof of Theorem 5.3. Soundness: We need only to show that each rule of the proof system is valid in the sense of total correctness. Take rule (WhileT) as an example. Let **while** $\triangleq$ **while** $q$ **do** $S$ **end**,

$$\models_{tot} \{b \wedge \Theta\}\ S\ \{\Theta\}, \tag{12}$$

and $\{\Theta_n : n \geq 0\}$ be a sequence of $\Theta$-ranking assertions for **while**. Assume without loos of generality $qv(S) \subseteq qv(\Theta)$. We prove by induction on $n$ that

$$\Theta \sqsubseteq \Theta_n + \Psi_n$$

for any $n \geq 0$, where $\Psi_0 \triangleq \bot_{qv(\Theta)}$, and for any $n \geq 0$, $\Psi_{n+1} \triangleq \neg b \wedge \Theta + b \wedge wp.S.\Psi_n$. The case when $n = 0$ is from the assumption that $\Theta \sqsubseteq \Theta_0$. For $n \geq 0$, we calculate

$$b \wedge \Theta \sqsubseteq wp.S.\Theta \sqsubseteq wp.S.\Theta_n + wp.S.\Psi_n$$

where the first inequality follows from Eq.(12), and the second one from the induction hypothesis and Lemma 4.16(4). Thus

$$\begin{aligned}\Theta &= b \wedge \Theta + \neg b \wedge \Theta \\ &\sqsubseteq b \wedge \Theta_{n+1} + b \wedge wp.S.\Psi_n + \neg b \wedge \Theta \\ &\sqsubseteq \Theta_{n+1} + \Psi_{n+1},\end{aligned}$$

where the first inequality follows from the definition of ranking assertions, and the second one from that of $\Psi_{n+1}$. Thus

$$\Theta \sqsubseteq wp.(\textbf{while}\ b\ \textbf{do}\ S\ \textbf{end}).(\neg b \wedge \Theta)$$

by noting that $wp.(\textbf{while}\ b\ \textbf{do}\ S\ \textbf{end}).(\neg b \wedge \Theta) = \bigvee_n \Psi_n$ and $\bigwedge_n \Theta_n = \bot_{\mathcal{H}}$, and so

$$\models_{tot} \{\Theta\}\ \textbf{while}\ b\ \textbf{do}\ S\ \textbf{end}\ \{\neg b \wedge \Theta\}$$

as desired.

Completeness: By the (Imp) rule, it suffices to show that for any all $\Theta$ and $S'$ with $qv(S') \subseteq qv(\Theta)$,

$$\vdash_{tot} \{wp.S'.\Theta\}\ S'\ \{\Theta\}.$$

Again, we take the case for while-loops as an example. Let **while** $\triangleq$ **while** $b$ **do** $S$ **end** and $\Psi \triangleq wp.\textbf{while}.\Theta$. By induction, we have $\vdash_{tot} \{wp.S.\Psi\}\ S\ \{\Psi\}$. Note that

$$\Psi = \neg b \wedge \Theta + b \wedge wp.S.\Psi.$$

Thus $b \wedge \Psi = b \wedge wp.S.\Psi \sqsubseteq wp.S.\Psi$, and so $\vdash_{tot} \{b \wedge \Psi\}\ S\ \{\Psi\}$ by rule (Imp).

Let $\Theta_0 = wp.\textbf{while}.\top_{qv(\Theta)}$ and $\Theta_{n+1} = b \wedge wp.S.\Theta_n$. We are going to show that $\{\Theta_n : n \geq 0\}$ are $\Theta$-ranking assertions for **while**. First, note that

$$\Theta_1 = b \wedge wp.S.\Theta_0 \sqsubseteq \neg b \wedge \top_{qv(\Theta)} + b \wedge wp.S.\Theta_0 = \Theta_0.$$

So $\{\Theta_n : n \geq 0\}$ is decreasing by easy induction, using Lemma 4.16(3). Next, as $\Theta \sqsubseteq \top_{qv(\Theta)}$, we have $\Psi \sqsubseteq \Theta_0$.

Finally, we prove that $\bigwedge_n \Theta_n = \bot_{qv(\Theta)}$. We show by induction on $n$ that for any $n \geq 0$ and $\Delta \in \mathcal{S}_{qv(\Theta,\textbf{while})}$,

$$\mathrm{Exp}(\Delta \models \Theta_n) = \mathrm{tr}([\![\textbf{while}]\!](\Delta)) - \mathrm{tr}([\![\textbf{while}^n]\!](\Delta)). \tag{13}$$

The case when $n = 0$ is direct from Lemmas 3.9 and 4.14. We further calculate that

$$\begin{aligned}\mathrm{Exp}(\Delta \models \Theta_{n+1}) &= \mathrm{Exp}(\Delta \models b \wedge wp.S.\Theta_n)\\ &= \mathrm{Exp}(\Delta|_b \models wp.S.\Theta_n)\\ &= \mathrm{Exp}([\![S]\!](\Delta|_b) \models \Theta_n)\\ &= \mathrm{tr}([\![\mathbf{while}]\!]([\![S]\!](\Delta|_b))) - \mathrm{tr}([\![\mathbf{while}^n]\!]([\![S]\!](\Delta|_b)))\\ &= \mathrm{tr}([\![\mathbf{while}]\!](\Delta)) - \mathrm{tr}([\![\mathbf{while}^{n+1}]\!](\Delta)).\end{aligned}$$

Here the second last equality is from induction hypothesis, and the last one from Lemma 4.7. Note that the second term of the r.h.s of Eq.(13) converges to the first one when $n$ goes to infinity. Thus $\lim_n \mathrm{Exp}(\Delta \models \Theta_n) = 0$, and so $\bigwedge_n \Theta_n = \bot_{\mathcal{H}}$ from the arbitrariness of $\Delta$ and Lemma 3.11. Now using rule (WhileT), we have $\vdash_{tot} \{\Psi\}$ **while** $\{\neg b \wedge \Psi\}$ and the result follows from the fact that $\neg b \wedge \Psi = \neg b \wedge \Theta \sqsubseteq \Theta$. □

Proof of Lemma 6.1. In the proof, by $\vdash_*$ we mean the provability by proof systems for both partial and total correctness. Similarly, $\models_*$ denotes validity in the sense of both partial and total correctness. The rules (Top) and (Bot) are from Lemma 4.12. We note from (Meas) that whenever $\bar{q} \cap qv(\Theta) = \emptyset$,

$$\vdash \left\{\sum_{i\in J} \Theta[i/x] \otimes M_i^\dagger M_i\right\} x := \mathbf{meas}\ \mathcal{M}[\bar{q}]\{\Theta \otimes I_{\bar{q}}\}.$$

Then (Meas0) follows by (Imp). The proofs for (Init0) and (Unit0) are similar. (Param) follows from Lemma 4.15(4).

(SupOper): From $\models_* \{\Theta\}\ S\ \{\Psi\}$, we have $\Theta \lesssim wp.S.\Psi$ (or $\Theta \lesssim wlp.S.\Psi$ for partial correctness) by Lemma 4.17. Then $\models_* \{\mathcal{F}_{V\to W}(\Theta)\}\ S\ \{\mathcal{F}_{V\to W}(\Psi)\}$ from Lemma 4.16(2). (SupPos) follows from (SupOper) by taking

$$\mathcal{F}_{W\to\emptyset}(\Theta') = \langle\psi^*|\Theta'|\psi^*\rangle$$

for any $\Theta' \in \mathcal{A}_W$, where $|\psi^*\rangle = \sum_{i\in I} \alpha_i^*|i\rangle_W$. (Tens) follows from (SupOper) by taking $V = \emptyset$ and $\mathcal{F}_{V\to W}(1) = M$. Conversely, in (Trace) we take $W = \emptyset$ and

$$\mathcal{F}_{V\to W}(\Theta) = \frac{1}{\dim(\mathcal{H}_V)} \sum_{i\in J} \langle i|_V \Theta |i\rangle_V$$

where $\{|i\rangle : i \in J\}$ is an orthonormal basis of $\mathcal{H}_V$.

The rules (Exist), (Inv), (Disj), (Sum), and (L-sum) are all easy from definition. Note that to prove (L-sum) for partial correctness, we have to require $\sum_i \lambda_i \leq =1$.

(ProbComp): For any $\sigma$ and $\rho$ with $\sigma \models p'$ and $\mathrm{tr}(\rho) = 1$, let $\Delta' \triangleq [\![S_1]\!](\sigma,\rho)$ and $\Delta'' \triangleq [\![S_2]\!](\Delta')$. We first have from $\models_{tot} \{p'\}\ S_1\ \{\langle p, |\psi\rangle_{\bar{q}}\langle\psi|\rangle\}$ that

$$1 = \mathrm{Exp}(\langle\sigma,\rho\rangle \models p') \leq \sum_{\sigma'\in\lceil\Delta'\rceil, \sigma'\models p} \langle\psi|\Delta'(\sigma')|\psi\rangle \leq 1.$$

Thus for any $\sigma' \in \lceil\Delta'\rceil$, there is some $c_{\sigma'} \geq 0$, $\sum_{\sigma'\in\lceil\Delta'\rceil} c_{\sigma'} = 1$, such that $\sigma' \models p$ and $\Delta'(\sigma') = c_{\sigma'}|\psi\rangle\langle\psi|$. Furthermore, by $\models_{tot} \{\langle p, M_{\bar{q}}\rangle\}\ S_2\ \{\Psi\}$, we have

$$\sum_{\sigma'\in\lceil\Delta'\rceil} c_{\sigma'}\langle\psi|M|\psi\rangle = \mathrm{Exp}(\Delta' \models \langle p, M_{\bar{q}}\rangle) \leq \mathrm{Exp}(\Delta'' \models \Psi).$$

The result then follows from the observation that

$$\mathrm{Exp}(\langle\sigma,\rho\rangle \models \langle\psi|M|\psi\rangle \cdot p') = \langle\psi|M|\psi\rangle.$$

(C-WhileT): first note that $\models_{tot} \{b \wedge p \wedge t = z\}\ S\ \{t < z\}$ implies for any $\sigma \models b \wedge p \wedge t = z$, and any $\sigma'$ in the support of $[\![S]\!](\sigma, \rho)$, we have $\sigma' \models t < z$. Then an argument similar to that for classical programs leads to the conclusion that all computations from $\langle \textbf{while}\ b\ \textbf{do}\ S\ \textbf{end}, \sigma, \rho\rangle$ terminates within $\sigma(t)$ steps, provided that $\sigma \models p$. □

In the following sections, all quantum variables are assumed to have **Qubit** type.

### A.2 Quantum Fourier Transform

Recall that the $n$-qubit quantum Fourier transform (QFT) is a unitary mapping such that for any integer $j$, $0 \leq j \leq 2^n - 1$,

$$|j\rangle \rightarrow |\psi_j\rangle \triangleq \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i jk/2^n} |k\rangle = \bigotimes_{k=n}^{1} |+_{0.j_k \cdots j_n}\rangle$$

where $j_1 \cdots j_n$ is the binary representation of $j$, $|+_{0.j_k \cdots j_n}\rangle \triangleq (|0\rangle + e^{2\pi i 0.j_k \cdots j_n}|1\rangle)/\sqrt{2}$. In particular, $|+_0\rangle = |+\rangle$. QFT serves as an important part for Shor's quantum factorisation and many other quantum algorithms.

The QFT algorithm for $n$ qubits can be described in our cq-language as follows:

$QFT(n) \triangleq$

$x := 1;$
**while** $x \leq n$ **do**
  $\bar{q}[x] *= H;\ y := x + 1;$
  **while** $y \leq n$ **do**
    $\bar{q}[y, x] *= CR(y - x + 1);\ y := y + 1;$
  **end**
  $x := x + 1;$
**end**
$\bar{q} *= SWAP_n$

where for each $1 \leq k \leq n$, $CR(k)$ is the controlled-$R_k$ operator with

$$R_k = |0\rangle\langle 0| + \exp(2\pi i/2^k)|1\rangle\langle 1|,$$

and $SWAP_n$ reverses the order of a list of $n$ qubits; that is, $SWAP_n|i_1, \cdots, i_n\rangle_{\bar{q}} = |i_n, \cdots, i_1\rangle_{\bar{q}}$ for all $|i_j\rangle \in \mathcal{H}_{q_j}$. The correctness of $QFT(n)$ is stated as follows: for any $\alpha_j \in \mathbb{C}$, $\sum_j |\alpha_j|^2 = 1$,

$$\vdash_{tot} \{\langle \textbf{true}, \sum_j \alpha_j |j\rangle_{\bar{q}}\rangle\} QFT(n) \{\langle \textbf{true}, \sum_j \alpha_j |\psi_j\rangle_{\bar{q}}\rangle\}.$$

With the help of rule (SupPos), it suffices to prove

$$\vdash_{tot} \{\langle \textbf{true}, |\alpha\rangle_{\bar{q},\bar{q}'}\rangle\} QFT(n) \{\langle \textbf{true}, |\beta\rangle_{\bar{q},\bar{q}'}\rangle\}$$

where $|\bar{q}'| = |\bar{q}|$, $|\alpha\rangle \triangleq \sum_{j=0}^{2^n-1} |j\rangle|j\rangle$ is the unnormalised maximally entangled state in $\mathcal{H}_2^{\otimes 2n}$, and $|\beta\rangle \triangleq \sum_{j=0}^{2^n-1} |\psi_j\rangle|j\rangle$.

The proof is rather involved. Due to the limit of space, we sketch the main ideas instead.

(1) Let **while′** be the inner loop. We show that

$$\Psi \triangleq \sum_{\ell=1}^{n} \sum_{m=\ell+1}^{n+1} \langle x = \ell \wedge y = m, \sum_{j=0}^{2^n-1} \left[ \bigotimes_{k=1}^{\ell-1} |+_{0.j_k \cdots j_n}\rangle \otimes |+_{0.j_l \cdots j_{m-1}}\rangle \bigotimes_{k=\ell+1}^{n} |j_k\rangle \right]_{\bar{q}} |j\rangle_{\bar{q}'}$$

serves as an invariant for **while**′. Furthermore, let $p \triangleq (1 \le x \le n) \wedge (x+1 \le y \le n+1)$. Then $t \triangleq n+1-y$ serves as a classical ranking function for **while**′. Thus we have from (C-WhileT)

$$\vdash_{tot} \{\Psi\}\ \textbf{while}'\ \{y > n \wedge \Psi\}.$$

(2) Let **while** be the outer while-loop, and

$$\Theta \triangleq \sum_{\ell=1}^{n+1} \langle x = \ell,\ \sum_{j=0}^{2^n-1} \left[ \bigotimes_{k=1}^{\ell-1} |+_{0.j_k \cdots j_n}\rangle \otimes \bigotimes_{k=\ell}^{n} |j_k\rangle \right]_{\bar{q}} |j\rangle_{\bar{q}'}.$$

Then it can be shown that $\Theta$ is an invariant for **while**. Again, it is easy to construct a classical ranking function ($t \triangleq n+1-x$), so

$$\vdash_{tot} \{\Theta\}\ \textbf{while}\ \{x > n \wedge \Theta\}. \tag{14}$$

(3) For the whole program, we have

$\{\langle \textbf{true}, |\alpha\rangle_{\bar{q},\bar{q}'}\rangle\}$

$x := 1;$

$$\left\{ \sum_{\ell=1}^{n+1} \langle x = \ell,\ \sum_{j=0}^{2^n-1} \left[ \bigotimes_{k=1}^{\ell-1} |+_{0.j_k \cdots j_n}\rangle \otimes \bigotimes_{k=\ell}^{n} |j_k\rangle \right]_{\bar{q}} |j\rangle_{\bar{q}'} \right\} \qquad (Assn)$$

**while**

$$\left\{ \langle x = n+1,\ \sum_{j=0}^{2^n-1} \left[ \bigotimes_{k=1}^{n} |+_{0.j_k \cdots j_n}\rangle \right]_{\bar{q}} |j\rangle_{\bar{q}'} \right\} \qquad Eq.(14)$$

$\bar{q} *= SWAP_n$

$$\{\langle \textbf{true}, \sum_{j=0}^{2^n-1} |\psi_j\rangle_{\bar{q}} |j\rangle_{\bar{q}'}\rangle\}. \qquad (Unit, Imp)$$

### A.3 Phase Estimation

Given (the controlled version of) a unitary operator $U$ acting on $m$ qubits and one of its eigenstate $|u\rangle$ with $U|u\rangle = e^{2\pi i\varphi}|u\rangle$ for some $\varphi \in [0,1]$. The phase estimation algorithm computes an $n$-bit approximation $\tilde{\varphi}$ of $\varphi$ with success probability at least $1-\epsilon$, where $n$ and $\epsilon$ are two given parameters.

Let $t \triangleq n + \lceil \log(2 + \frac{1}{2\epsilon}) \rceil$. The algorithm is detailed as follows:

$$
\begin{array}{l}
PE \triangleq \\
\quad \bar{r} := 0;\ \bar{r} *= U_u;\ \bar{q} := 0;\ x := 1; \\
\quad \textbf{while}\ x \le t\ \textbf{do} \\
\quad\quad \bar{q}[x] *= H;\ y := 0; \\
\quad\quad \textbf{while}\ y < 2^{t-x}\ \textbf{do} \\
\quad\quad\quad \bar{q}[x], \bar{r} *= CU;\ y := y + 1; \\
\quad\quad \textbf{end} \\
\quad\quad x := x + 1; \\
\quad \textbf{end} \\
\quad \bar{q} *= QFT(t)^{\dagger}; \\
\quad z := \textbf{meas}\ \bar{q}
\end{array}
$$

where $|\bar{q}| = t$, $|\bar{r}| = m$, $U_u$ is a unitary operator to prepare $|u\rangle$ from $|0\rangle$, $CU$ is the controlled-$U$ operator, and $QFT(t)^{\dagger}$ is the inverse quantum Fourier transform on $t$ qubits.

The correctness of $PE$ can be stated as

$$\{p_{PE} \cdot \top\} PE \left\{ \left| \frac{z}{2^t} - \varphi \right| \le \frac{1}{2^n} \right\} \tag{15}$$

with $p_{PE} \ge 1 - \epsilon$. Let **while** be the outer while-loop and **while′** be the inner one. The proof consists of three phases.

(1) For the body of **while′**, we have for any $1 \le k \le t$ and $0 \le \ell \le 2^{t-k}$,

$$
\begin{array}{ll}
\{\langle x = k \wedge y = \ell, |+_{\ell\varphi}\rangle_{q_k} |u\rangle_{\bar{r}}\} & \\
\bar{q}[x], r *= CU;\ y := y + 1; & \\
\{\langle x = k \wedge y = \ell + 1, |+_{(\ell+1)\varphi}\rangle_{q_k} |u\rangle_{\bar{r}}\} & (Unit, Assn)
\end{array}
$$

where $|+_a\rangle \triangleq \frac{|0\rangle + e^{2\pi i a}|1\rangle}{\sqrt{2}}$ for any $a \in \mathbb{R}$, and in particular, $|+_0\rangle = |+\rangle$. Furthermore, it is easy to construct a classical ranking function $2^{t-x} - y$. Thus we have from (Sum) and (C-WhileT),

$$\vdash_{tot} \{\Psi\}\ \textbf{while}'\ \{\Psi \wedge y \ge 2^{t-x}\} \tag{16}$$

where $\Psi \triangleq \sum_{\ell=0}^{2^{t-k}} \langle x = k \wedge y = \ell, |+_{\ell\varphi}\rangle_{q_k} |u\rangle_{\bar{r}}\rangle$.

(2) For the body of **while**, we have for any $1 \le k \le t$,

$$
\begin{array}{ll}
\{\langle x = k, |0\rangle_{q_k} |u\rangle_{\bar{r}}\rangle\} & \\
\bar{q}[x] *= H;\ y := 0; & \\
\left\{ \Psi \equiv \displaystyle\sum_{\ell=0}^{2^{t-k}} \langle x = k \wedge y = \ell, |+_{\ell\varphi}\rangle_{q_k} |u\rangle_{\bar{r}}\rangle \right\} & (Unit, Assn) \\
\textbf{while}' & \\
\left\{ \Psi \wedge y \ge 2^{t-x} \equiv \langle x = k, |+_{2^{t-k}\varphi}\rangle_{q_k} |u\rangle_{\bar{r}}\rangle \right\} & Eq.(16) \\
x := x + 1; & \\
\left\{ \langle x = k + 1, |+_{2^{t-k}\varphi}\rangle_{q_k} |u\rangle_{\bar{r}}\rangle \right\} & (Assn)
\end{array}
$$

Furthermore, it is easy to construct a classical ranking function $t+1-x$. Thus from (Tens), (Sum), and (C-WhileT) we have

$$\vdash_{tot} \{\Theta\}\ \textbf{while}\ \{\Theta \wedge x > t\} \tag{17}$$

where

$$\Theta \triangleq \sum_{k=1}^{t+1} \langle x = k, \bigotimes_{j=1}^{k-1} |+_{2^{t-j}\varphi}\rangle \otimes |0\rangle^{\otimes(t-k+1)} |u\rangle_{\bar{r}} \rangle.$$

(3) For the whole program, we have

$$\begin{aligned}
&\{\top\}\\
&\bar{r} := 0; \bar{r} *= U_u;\ \bar{q} := 0;\\
&\left\{\langle \textbf{true}, |0\rangle^{\otimes t} |u\rangle_{\bar{r}}\rangle\right\} && (Init, Unit)\\
&x := 1;\\
&\left\{\sum_{k=1}^{t+1} \langle x = k, \bigotimes_{j=1}^{k-1} |+_{2^{t-j}\varphi}\rangle \otimes |0\rangle^{\otimes(t-k+1)} |u\rangle_{\bar{r}}\rangle\right\} && (Assn)\\
&\textbf{while}\\
&\left\{\langle \textbf{true}, \bigotimes_{j=1}^{t} |+_{2^{t-j}\varphi}\rangle |u\rangle_{\bar{r}}\rangle\right\} && Eq.(17)
\end{aligned}$$

Furthermore, let

$$K \triangleq \left\{0 \le m < 2^t : \left|\frac{m}{2^t} - \varphi\right| \le \frac{1}{2^n}\right\}$$

and for each $m$, $|\psi_m\rangle = 1/\sqrt{2^t} \sum_{j=0}^{2^t-1} e^{2\pi i j m/2^t} |j\rangle$. Then we have

$$\begin{aligned}
&\left\{\langle \textbf{true}, \sum_{m\in K} |\psi_m\rangle_{\bar{q}} \langle\psi_m|\rangle\right\}\\
&\bar{q} *= QFT^{\dagger};\\
&\left\{\langle \textbf{true}, \sum_{m\in K} |m\rangle_{\bar{q}} \langle m|\rangle\right\} && (Unit)\\
&z := \textbf{meas}\ \bar{q}\\
&\left\{\left|\frac{z}{2^t} - \varphi\right| \le \frac{1}{2^n}\right\} && (Meas0)
\end{aligned}$$

Finally, by (ProbComp) we prove Eq.(15) where

$$p_{PE} = \sum_{m\in K} \left| \langle\psi_m| \bigotimes_{j=1}^{t} |+_{2^{t-j}\varphi}\rangle \right|^2.$$

A calculation similar to [Nielsen and Chuang 2002] shows that $p_{PE} \ge 1-\epsilon$.

### A.4 Order-finding

Given positive co-prime positive integers $x$ and $N$ with $x < N$, the order of $x$ modulo $N$ is the least positive integer $r$ such that $x^r \equiv_N 1$. Let $L \triangleq \lceil \log N \rceil$. Suppose we are given (the controlled version of) a unitary operator $U$ such that

$$U|y\rangle = \begin{cases} |xy \bmod N\rangle & \text{if } y < N \\ |y\rangle & \text{otherwise.} \end{cases}$$

The order-finding algorithm computes the order $r$ of $x$ by using $O(L^3)$ operations, with success probability at least $(1-\epsilon)/r$. Let $t \triangleq 2L + 1 + \lceil \log(2 + \frac{1}{2\epsilon}) \rceil$. The algorithm goes as follows:

$$\begin{aligned} OF(x,N) \triangleq\ & \\ & \bar{q} := 0;\ \bar{q} *= H^{\otimes t}; \\ & \bar{q}' := 0;\ \bar{q}' *= U_{+1}; \\ & \bar{q}, \bar{q}' *= CU; \\ & \bar{q} *= QFT(t)^\dagger; \\ & z' := \mathbf{meas}\ \bar{q}; \\ & z := f(z'/2^t) \end{aligned}$$

where $|\bar{q}| = t$, $|\bar{q}'| = L$, $U_{+1}$ is a unitary operator on $\mathcal{H}_2^{\otimes L}$ such that $U_{+1}|0\rangle = |1\rangle$, and $f$ is the continued fractions algorithm.

The correctness of $OF(x,N)$ can be stated as

$$\vdash_{tot} \{p_{OF} \cdot (gcd(x,N) = 1)\} OF(x,N) \{z = r\}. \tag{18}$$

for some $p_{OF} > 0$. Note that $1/\sqrt{r} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle$. We compute

$$\begin{aligned} & \{gcd(x,N) = 1\} \\ & \bar{q} := 0;\ \bar{q} *= H^{\otimes t};\ \bar{q}' := 0;\ \bar{q}' *= U_{+1}; \\ & \{\langle gcd(x,N) = 1, |+^{\otimes t}\rangle_{\bar{q}} |1\rangle_{\bar{q}'}\rangle\} && (Init, Unit) \\ & \left\{ \langle gcd(x,N) = 1, \frac{1}{\sqrt{r2^t}} \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} |j\rangle_{\bar{q}} |u_s\rangle_{\bar{q}'} \rangle \right\} && (Imp) \\ & \bar{q}, \bar{q}' *= CU; \\ & \left\{ \langle gcd(x,N) = 1, \frac{1}{\sqrt{r2^t}} \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} e^{2\pi i j s/r} |j\rangle_{\bar{q}} |u_s\rangle_{\bar{q}'} \rangle \right\} && (Unit) \\ & \bar{q} *= QFT(t)^\dagger; \\ & \left\{ \langle gcd(x,N) = 1, \frac{1}{\sqrt{r}2^t} \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} \sum_{k=0}^{2^t-1} \exp\left[2\pi i j \left(\frac{s}{r} - \frac{k}{2^t}\right)\right] |k\rangle_{\bar{q}} |u_s\rangle_{\bar{q}'} \rangle \right\} && (Unit) \end{aligned}$$

Furthermore, note that for any $0 \le s < r$ and $0 \le k < 2^t$, the continued fractions algorithm $f$ computes $f(k/2^t) = r$ provided

$$\left| \frac{s}{r} - \frac{k}{2^t} \right| \le \frac{1}{2r^2}.$$

Thus

$$\left\{\langle gcd(x,N)=1, \sum_{(s,k)\in H} |k\rangle_{\bar{q}}\langle k|\rangle\right\}$$
$$z' := \textbf{meas}\ \bar{q};$$
$$\{f(z'/2^t)=r\} \qquad (Meas0)$$
$$z := f(z'/2^t)$$
$$\{z=r\} \qquad (Assn)$$

where

$$H \triangleq \left\{(s,k) : 0 \le s < r, 0 \le k < 2^t, \left|\frac{s}{r} - \frac{k}{2^t}\right| \le \frac{1}{2r^2}\right\}.$$

Then by (ProbComp), we have proven Eq.(18) where

$$p_{OF} = \frac{1}{r2^t} \sum_{(s',k)\in H} \left| \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} \exp\left[2\pi i j \left(\frac{s}{r} - \frac{k}{2^t}\right)\right]\right|^2.$$

A tedious calculation shows that when $gcd(x,N)=1$, $p_{OF} \ge (1-\epsilon)/r$.