Bayesian Inference in Quantum Programs

Christina Gehnen
RWTH Aachen University, Germany

Dominique Unruh
RWTH Aachen University, Germany

Joost-Pieter Katoen
RWTH Aachen University, Germany

Abstract

Conditioning is a key feature in probabilistic programming to enable modeling the influence of data (also known as observations) to the probability distribution described by such programs. Determining the posterior distribution is also known as Bayesian inference. This paper equips a quantum while-language with conditioning, defines its denotational and operational semantics over infinite-dimensional Hilbert spaces, and shows their equivalence. We provide sufficient conditions for the existence of weakest (liberal) precondition-transformers and derive inductive characterizations of these transformers. It is shown how w(l)p-transformers can be used to assess the effect of Bayesian inference on (possibly diverging) quantum programs.

2012 ACM Subject Classification Theory of computation → Quantum information theory; Theory of computation → Logic and verification; Theory of computation → Program semantics

Keywords and phrases Quantum Computing, Weakest Precondition, Conditioning, Program Semantics

Funding This work is supported by the ERC Grant 819317 CerQuS and the Interdisciplinary Doctoral Program in Quantum Systems Integration funded by the BMW group.

Contents

1 Introduction

Quantum verification is a important part of the rapidly evolving field of quantum computing and information. The importance comes from several factors. Firstly, quantum computers operate in a completely different way than classical computers do. Principles of quantum mechanics are important to algorithm designers but in general unintuitive to most people. This leads to a higher risk of introducing logical errors. Secondly, quantum algorithms are often used in safely critical areas such as cryptography and optimization where those mistakes can lead to serious issues. Classical testing and debugging methods do not directly apply to quantum computing. Testing on quantum computers is challenging due to high execution costs, probabilistic outcomes, and noise from environmental interactions. While simulators help, they have limitations such as scalability. Debugging is also difficult, as measuring quantum variables alters their state, preventing traditional inspection methods.

Testing only verifies specific inputs without guaranteeing overall correctness, whereas formal verification ensures correctness for all inputs. Weakest preconditions define input states that ensure a given postcondition holds after execution. Inspired by the importance of conditioning and Bayesian inference in probabilistic programs, we extend the calculus from [7, 27] to incorporate "observations". Combining weakest preconditions for total correctness and weakest liberal preconditions for partial correctness, we determine whether a predicate holds assuming all observations hold, i.e, compute a conditional probability.

This new statement could be used to aid in debugging to locate logical mistakes. Assume having a theoretical algorithm and a wrong implementation. To figure out which parts are wrong, fixing variable values by observations can help identify errors by comparing implementation samples with the expected distribution. Similarly, given a complex (possibly wrong) algorithm, adding observations can help understanding parts of the algorithm by comparing it to its intuitive understanding. For instance, in a random walk algorithm with a random starting point, analyzing success probability from a "good" starting point can help to understand the algorithm. Unlike traditional assertions, observations can be useful even when they don't always hold. Another possible application is error correction, where outputs are often analysed assuming no more than t qubit errors occurred per step to ensure successful error correction.

Related Work

The general idea of weakest preconditions was first developed by Dijkstra for classical programs [10, 9], then for probabilistic programs [15, 19] and later for quantum programs [7].

D'Hondt and Panangaden [7] defined predicates as positive operators as we do and focused on total correctness and finite-dimensional Hilbert spaces. [27] extended this approach to partial correctness and gave an explicit representation of the predicate transformer for the quantum while-language. An alternative to define predicates is to use projections [29]. There have been several extensions like adding classical variables [6, 13] or non-determinism [12].

A runtime assertion scheme using projective predicates for testing and debugging has been introduced in [17]. In contrast, our approach enables debugging, but in addition provides formal guarantees on the correctness based on the satisfaction of assertions and allows infinite-dimensional Hilbert spaces. A survey about studies and approaches of debugging of quantum programs is given in [8]. Another idea to locate bugs is to use incorrectness logic with projective predicates [26]. The idea of conditional weakest preconditions has been introduced in [21, 22] for probabilistic programs.

The concept of choosing specific measurement outcomes is also known as *postselection*. [1] shows that the class of problems solvable by quantum programs with postselection in polynomial time, called Postselected Bounded-Error Quantum Polynomial-Time (PostBQP), is the same as the ones in the complexity class Probabilistic Polynomial-Time (PP). This equivalence is shown by solving a representative PP-complete problem, MAJ-SAT, using a quantum program with postselection. We confirm the correctness of this program in Section 5 by using conditional weakest preconditions.

Main Contributions

- Conditional weakest-precondition transformers: We define a weakest precondition calculus for reasoning about programs with an "observe" statement. The conditional weakest precondition, defined in terms of weakest (liberal) preconditions transformers, reveals the probability of a postcondition given all observations succeed.

  The definition of the transformers is semantic, i.e., formulated in a generic way based on the denotational semantics and not tied to a specific syntax of programs (but we also give explicit rules for our syntax by recursion over the structure of a program).
- Semantics: We develop both denotational and operational semantics of a simple quantum while-language with "observe" statements and show their equivalence.
- Our definition of weakest (liberal) preconditions is a conservative extension of [27], supporting "observe" statements. Further differences include: Our definition is semantic and we support infinite-dimensional quantum systems (e.g., to support quantum integers)*.

Structure

We first recall important definitions in Section 2. The main contributions are in Section 3 and Section 4: Section 3 introduces the "observe" statement and its semantics whereas Section 4 defines weakest (liberal) preconditions and finally conditional weakest (liberal) preconditions. Two examples in Section 5 illustrate our approach, followed by conclusions in Section 6.

*Notice that [27] also defines a language with quantum integers. However, they do not explicitly specify the various notions of convergence of operators (e.g., operator topologies, convergence of infinite sums, existence of suprema), making it difficult to verify whether their rules are sound in the infinite-dimensional case.

2 Preliminaries

2.1 Hilbert Spaces

Let $\langle \cdot \mid \cdot \rangle$ denote the inner product over a vector space $\mathcal{V}$. The *norm (or length) of a vector* u, denoted $\|u\|$, is defined as $\sqrt{\langle u \mid u \rangle}$. The vector u is called a unit vector if $\|u\| = 1$. Vectors u, v are *orthogonal* ($u \perp v$) if $\langle u \mid v \rangle = 0$. The sequence $\{u_i\}_{i \in \mathbb{N}}$ of vectors $u_i \in \mathcal{V}$ is a *Cauchy sequence*, if for any $\epsilon > 0$, there exists a positive integer N such that $\|u_n - u_m\| < \epsilon$ for all $n, m \geq N$. If for any $\epsilon > 0$, there exists a positive integer N such that $\|u_n - u\| < \epsilon$ for all $n \geq N$, then u is the limit of $\{u_i\}_{i \in \mathbb{N}}$, denoted $u = \lim_{i \to \infty} u_i$.

A family $\{u_i\}_{i \in I}$ of vectors in $\mathcal{V}$ is *summable* with the sum $v = \sum_{i \in I} u_i$ if for every $\epsilon > 0$ there exists a finite $J \subseteq I$ such that $\|v - \sum_{i \in K} u_i\| < \epsilon$ for every finite $K \subseteq I$ and $J \subseteq K$.

A *Hilbert space* $\mathcal{H}$ is a complete inner product space, i.e, every Cauchy sequence of vectors in $\mathcal{H}$ has a limit [27]. An orthonormal *basis* of a Hilbert space $\mathcal{H}$ is a (possibly infinite) family $\{u_i\}_{i \in I}$ of unit vectors if they are pairwise orthogonal (i.e., $u_i \perp u_j$ for $i \neq j, i, j \in I$) and every $v \in \mathcal{H}$ can be written as $v = \sum_{i \in I} \langle u_i \mid v \rangle \cdot u_i$ (in the sense above). The cardinality of I, denoted $|I|$, is the dimension of $\mathcal{H}$. Hilbert spaces and its elements can be combined using the *tensor product* $\otimes$ [23, Def. IV.1.2].

We use Dirac notation $|\phi\rangle$ to denote vectors of a vector space where $\langle\phi|$ is the dual vector of $|\phi\rangle$ [11], i.e., $\langle\phi| = |\phi\rangle^\dagger$.

▶ **Example 1.** A typical Hilbert space over the set X is

$$l^2(X) = \{\sum_{n \in X} \alpha_n |n\rangle \mid \alpha_n \in \mathbb{C} \text{ for all } n \in X \text{ and } \sum_{n \in X} |\alpha_n|^2 < \infty\}$$

where the inner product is defined as $(\sum_{n \in X} \alpha_n |n\rangle, \sum_{n \in X} \alpha'_n |n\rangle) = \sum_{n \in X} \overline{\alpha_n} \alpha'_n$. By $\overline{x + yi} = x - yi$ we denote the complex conjugate of $x + yi \in \mathbb{C}$. An orthonormal basis, also called *computational basis*, is $\{|n\rangle \mid n \in X\}$. For (countably) infinite sets X, the basis is (countably) infinite and thus $l^2(X)$ is a (countably) infinite Hilbert space. $l^2(\mathbb{Z})$ can be used for quantum integers and is also denoted by $\mathcal{H}_\infty$. For qubits, we use $l^2(\{0,1\})$ and denote it as $\mathcal{H}_2$.

2.2 Operators

In the following, all vector spaces will be over $\mathbb{C}$. For vector spaces $\mathcal{V}, \mathcal{W}$, a function $f : \mathcal{V} \to \mathcal{W}$ is called *linear* if $f(ax + y) = af(x) + f(y)$ for $x, y \in \mathcal{V}$ and $a \in \mathbb{C}$. If $\mathcal{V}, \mathcal{W}$ are normed vector spaces then f is called *bounded linear* if f is linear and $\|f(x)\| \leq c \cdot \|x\|$ for some constant $c \geq 0$ for all $x \in \mathcal{V}$. If $\mathcal{H}$ is a Hilbert space, we call bounded linear functions on $\mathcal{H} \to \mathcal{H}$ *operators*. Let $B(\mathcal{H})$ denote the space of all operators on $\mathcal{H}$ and $A|\phi\rangle$ the result of applying operator A to $|\phi\rangle \in \mathcal{H}$. For this work, we additionally generalize the notion of linearity to functions that are defined on subsets of the vector space: For (normed) vector spaces $S \subseteq \mathcal{V}, T \subseteq \mathcal{W}$ with *span*$(S) = \mathcal{V}$ and *span*$(T) = \mathcal{W}$, we call $f : S \to T$ *(bounded) linear* iff there exists a (bounded) linear function $\bar{f} : \mathcal{V} \to \mathcal{W}$ such that $\bar{f}(s) = f(s)$ for $s \in S$. *span*(S) includes all finite linear combinations of S.

Let A and B be operators on $\mathcal{H}_1$ and $\mathcal{H}_2$ with $|\phi\rangle \in \mathcal{H}_1, |\psi\rangle \in \mathcal{H}_2$. By [23, Def. IV.1.3], the tensor product $A \otimes B$ is the unique operator that satisfies $(A \otimes B)(|\phi\rangle \otimes |\psi\rangle) = A|\phi\rangle \otimes B|\psi\rangle$ For matrices, the tensor product is also called the *Kronecker product*.

For every operator A on $\mathcal{H}$, there exists an operator $A^\dagger$ on $\mathcal{H}$ with $\langle |\phi\rangle, A|\psi\rangle\rangle = \langle A^\dagger |\phi\rangle, |\psi\rangle\rangle$ for all $|\phi\rangle, |\psi\rangle \in \mathcal{H}$. An operator A on $\mathcal{H}$ is called *positive* if $\langle\psi| A |\psi\rangle \geq 0$ for all states $|\psi\rangle \in \mathcal{H}$ [20]. The *identity operator* $\mathbf{I}_\mathcal{H}$ on $\mathcal{H}$ is defined by $\mathbf{I}_\mathcal{H} |\phi\rangle = |\phi\rangle$. The *zero* *operator* on $\mathcal{H}$, denoted by $\mathbf{0}_{\mathcal{H}}$, maps every vector to the zero vector. We omit $\mathcal{H}$ if it is clear from the context. An *unitary operator* U is an operator such that its inverse is its adjoint $U^{-1} = U^\dagger$, i.e., $U^\dagger U = \mathbf{I}$ and $UU^\dagger = \mathbf{I}$ [16]. An *(ortho)projector* is an operator $P : \mathcal{H} \to \mathcal{H}$ such that $P^2 = P = P^\dagger$. For every closed subspace S, there exists a projector P_S with image S [5, Prop. II.3.2 (b)].

An operator A is a *trace class* operator if there exists an orthonormal basis $\{|\psi_i\rangle\}_{i\in I}$ such that $\{\langle\psi_i| \cdot |A| \cdot |\psi_i\rangle\}_{i\in I}$ is summable where $|A|$ is the unique positive operator B with $B^\dagger B = A^\dagger A$. Then the trace of A is defined as $tr(A) = \sum_{i\in I} \langle\psi_i| \cdot A \cdot |\psi_i\rangle$ where $\{|\psi_i\rangle\}_{i\in I}$ is an orthonormal basis. For a trace class operator A, it can be shown that $tr(A)$ is independent of the chosen base [27]. The trace is cyclic, i.e., $tr(AB) = tr(BA)$ [25], linear, i.e., $tr(A + B) = tr(A) + tr(B)$, scalar, i.e., $tr(cA) = c \cdot tr(A)$ for a constant c [4] and multiplicative, i.e., $tr(A \otimes B) = tr(A)tr(B)$ holds [25] for trace class operators A, B. We use $T(\mathcal{H})$ to denote the space of trace class operators on $\mathcal{H}$. Positive trace class operators with $tr(\rho) \leq 1$ are called *partial density operators*. The set of partial density operators is denoted $\mathcal{D}^-(\mathcal{H})$ with $span(\mathcal{D}^-(\mathcal{H})) = T(\mathcal{H})$. *Density operators* are partial density operators with $tr(\rho) = 1$. They are denoted as $\mathcal{D}(\mathcal{H})$. The *support* of a partial density operator ρ is the smallest closed subspace S such that $P_S \rho P_S = \rho$.

Let us consider some properties of functions that map operators to operators. $f : T_1 \to T_2$ with $T_1 \subseteq T(\mathcal{H}_1), T_2 \subseteq T(\mathcal{H}_2)$ is *trace-reducing* if $tr(f(\rho)) \leq tr(\rho)$ for all positive $\rho \in T_1$. $f : B_1 \subseteq B(\mathcal{H}_1) \to B_2 \subseteq B(\mathcal{H}_2)$ is *positive* if $f(a)$ is positive for positive $a \in B_1$ and *subunital* if $f(\mathbf{I}_{\mathcal{H}_1}) \sqsubseteq \mathbf{I}_{\mathcal{H}_2}$ and $\mathbf{I}_{\mathcal{H}_1} \in B_1$, where $\sqsubseteq$ is defined just below.

2.2.1 The Loewner Partial Order

To order operators, the *Loewner partial order* is used. For any operators A, B, it is defined by $A \sqsubseteq B$ iff $B - A$ is a positive operator. This is equivalent to $tr(A\rho) \leq tr(B\rho)$ for all partial density operators $\rho \in \mathcal{D}^-(\mathcal{H})$ [27]. The Loewner order is compatible w.r.t. addition (also known as monotonic), i.e., $A \sqsubseteq B$ implies $A+C \sqsubseteq B+C$ for any C, and w.r.t. multiplication of non-negative scalars, i.e., $A \sqsubseteq B$ implies $cA \sqsubseteq cB$ for $c \geq 0$ [3].

Using this order, we can define predicates [7]. A *quantum predicate* on a Hilbert space $\mathcal{H}$ is defined as an operator P on $\mathcal{H}$ with $\mathbf{0}_{\mathcal{H}} \sqsubseteq P \sqsubseteq \mathbf{I}_{\mathcal{H}}$. The set of quantum predicates on $\mathcal{H}$ is denoted by $\mathcal{P}(\mathcal{H})$ and $span(\mathcal{P}(\mathcal{H})) = B(\mathcal{H})$.

The Loewner partial order is an ω-complete partial order (ω-cpo) on the set of partial density operators [28]. Thus each increasing sequence of partial density operators has a least upper bound. This also holds for the set of predicates [7].

An important property that we need is continuity of the trace operator. First of all, we note that the trace-operator is order-continuous on partial density operators with respect to $\sqsubseteq$, i.e., $\bigvee_{i\in\mathbb{N}} tr(\rho_i) = tr(\bigvee_{i\in\mathbb{N}} \rho_i)$ for any increasing sequence of partial density operators $\{\rho_i\}_{i\in\mathbb{N}}$. Without going further into details, this holds because for an increasing sequence of real numbers, the least upper bound and the limit coincide, the same also holds for partial density operators [25] and because the trace is linear and bounded, it is also trace-norm continuous. Continuity w.r.t. predicates means $\bigvee_{i\in\mathbb{N}} tr(P_i\rho) = tr((\bigvee_{i\in\mathbb{N}} P_i)\rho)$ for every $\rho \in \mathcal{D}^-(\mathcal{H})$ and increasing sequence of predicates $\{P_i\}_{i\in\mathbb{N}}$. Without going into further detail, we can show that a function $f : \mathcal{B}(\mathcal{H}) \to \mathbb{C}$ defined by $f(A) = tr(A\rho)$ for a fixed $\rho \in \mathcal{D}^-(\mathcal{H})$ is weak*-continuous and convergence of positive bounded operators in the weak*-topology coincides with the supremum [25]. Similar, the same property holds for decreasing sequences of predicates $\{P_i\}_{i\in\mathbb{N}}$ and the greatest lower bound $\bigwedge_{i\in\mathbb{N}} P_i$.

2.3 Quantum-specific Preliminaries

Due to a postulate of quantum mechanics, the state space of an isolated quantum system can be described as a Hilbert space where states correspond to unit vectors (up to a phase shift) in its state space [27]. A quantum state is called *pure* if it can be described by a vector in the Hilbert space; otherwise *mixed*, i.e., it is a probabilistic distribution over pure states. We use partial density operators to describe mixed states, in particular to capture the current state of a program. If a quantum system is in a pure state $|\psi_i\rangle$ with probability p_i (with $\sum_i p_i \leq 1$), then this is represented by the partial density operator $\rho = \sum_i p_i |\psi_i\rangle \langle\psi_i|$.

To obtain the current value of e.g. a quantum variable, we cannot simply look at it. In quantum mechanics, each measurement can impact the current state of a qubit.

A *measurement* is a (possible infinite) family of operators $\{M_m\}_{m\in I}$ where m is the measurement outcome and $\sum_{m\in I} M_m^\dagger M_m = \mathbf{I}$ *. If the quantum system is in state $\rho \in \mathcal{D}^-(\mathcal{H})$ before the measurement $\{M_m\}$, then the probability for result m is $p(m) = tr(M_m \rho M_m^\dagger)$ and the post-measurement state is $\rho_m = \frac{M_m \rho M_m^\dagger}{p(m)}$. An important kind of measurement is the *projective measurement*. It is a set of projections $\{P_m\}$ over $\mathcal{H}$ with $\sum_m P_m = \mathbf{I}$. An important property of projective measurements is that if a state ρ is measured by a projective measurement $\{P, I - P\}$ and $supp(\rho) \subseteq P$ holds, then ρ is not changed.

2.4 Markov Chains

A *Markov chain* (MC) is a tuple $\mathcal{M} = (\Sigma, \mathbf{P}, s_{init})$ where

- Σ is a nonempty (possibly uncountable) set of states,
- $\mathbf{P} : \Sigma \times \Sigma \to [0, 1]$ with $\sum_{s'\in\Sigma} \mathbf{P}(s, s') = 1$ is the transition probability function. Let $s \xrightarrow{p} s'$ denote $\mathbf{P}(s, s') = p$.
- $s_{init} \in \Sigma$ is the initial state.

Note that in comparison to [2, 22], Σ can be uncountable. However, in our setting the reachable set of states will be countable as every state s can only have a countable number of successor states s' with $\mathbf{P}(s, s') > 0$. Therefore, even if Σ is uncountable, the set of reachable states is countable and all results from [2, 22] still apply.

A path of a MC $\mathcal{M}$ is an infinite sequence $s_0 s_1 s_2 \ldots \in \Sigma^\omega$ with $s_0 = s_{init}$ and $\mathbf{P}(s_i, s_{i+1}) > 0$ for all i. We use $Paths(\mathcal{M})$ to denote the set of paths in $\mathcal{M}$ and $Paths_{fin}(\mathcal{M})$ for the finite path prefixes. If it is clear from the context, we omit $\mathcal{M}$. The probability distribution $Pr^{\mathcal{M}}$ on $Paths(\mathcal{M})$ is defined using cylinder sets as in [2]. In a slight abuse of notation, we write $Pr^{\mathcal{M}}(\hat{\pi})$ for $Pr^{\mathcal{M}}(Cyl(\hat{\pi}))$ for $\hat{\pi} \in Paths_{fin}(\mathcal{M})$ where $Cyl(\hat{\pi})$ denotes the cylinder set of $\hat{\pi}$. We write $s_0 \to_p^* s_n$ where $p = \sum_{s_0\ldots s_n \in Paths_{fin}(\mathcal{M})} \mathbf{P}(s_0 \ldots s_n)$ is the probability to reach s_n from s_0. Given a target set of reachable states $T \subseteq \Sigma$, let $\Diamond T$ be the (measurable) set of infinite paths that reach the target set T. The probability of reaching T is $Pr^{\mathcal{M}}(\Diamond T) = \sum_{\hat{\pi}\in Paths_{fin}(\mathcal{M})\cap(\Sigma\setminus T)^* T} Pr^{\mathcal{M}}(\hat{\pi})$. Analogously, let $\neg\Diamond T$ be the set of paths that never reach T; $Pr^{\mathcal{M}}(\neg\Diamond T) = 1 - Pr^{\mathcal{M}}(\Diamond T)$.

*As in [25], we mean convergence of sums with respect to SOT (strong operator topology) which is the topology where $\lim_{i\to\infty} a_i = a$ holds iff for all ϕ: $\lim_{i\to\infty} a_i\phi = a\phi$ [5, Prop. IX.1.3(c)].

3 Quantum Programs with Observations

We assume *Var* to be a finite set of quantum variables with two types: Boolean and integer. As in [27], the corresponding Hilbert spaces are

$$\mathcal{H}_2 = \{\alpha\,|0\rangle + \beta\,|1\rangle \mid \alpha, \beta \in \mathbb{C}\},$$
$$\mathcal{H}_\infty = \Big\{\sum_{n\in\mathbb{Z}} \alpha_n\,|n\rangle \mid \alpha_n \in \mathbb{C} \text{ for all } n \in \mathbb{Z} \text{ and } \sum_{n\in\mathbb{Z}} |\alpha_n|^2 < \infty\Big\}.$$

Each variable $q \in \mathit{Var}$ has a type $\mathit{type}(q) \in \{\mathit{Bool}, \mathit{Int}\}$. Its state space $\mathcal{H}_q$ is $\mathcal{H}_2$ if $\mathit{type}(q) = \mathit{Bool}$ and $\mathcal{H}_\infty$ otherwise. The state space of a quantum register $\bar{q} = q_1, ..., q_n$ is defined by the tensor product $\mathcal{H}_{\bar{q}} = \bigotimes_{i=1}^{n} \mathcal{H}_{q_i}$ of state spaces of q_1 through q_n.

3.1 Syntax

A quantum while-program has the following syntax:

$$S ::= \textbf{skip} \mid q := 0 \mid \bar{q} := U\bar{q} \mid \textbf{observe}\ (\bar{q}, O) \mid S_1; S_2 \mid \textbf{measure}\ M[\bar{q}] : \bar{S} \mid \textbf{while}\ M[\bar{q}] = 1\ \textbf{do}\ S$$

where

- q is a quantum variable,
- $\bar{q}$ is a quantum register,
- U from statement $\bar{q} := U\bar{q}$ is a unitary operator on $\mathcal{H}_{\bar{q}}$ and $\bar{q}$ is the same on both sides,
- O in **observe** $(\bar{q}, O)$ is a projection on $\mathcal{H}_{\bar{q}}$
- the measurement $M = \{M_m\}_{m\in I}$ in **measure** $M[\bar{q}] : \bar{S}$ is on $\mathcal{H}_{\bar{q}}$ and $\bar{S} = \{S_m\}_{m\in I}$ is a family of quantum programs where each S_m corresponds to an outcome $m \in I$,
- the measurement in **while** $M[\bar{q}] = 1$ **do** S on $\mathcal{H}_{\bar{q}}$ has the form $M = \{M_0, M_1\}$.

Our programs extend [27] with the new statement **observe** $(\bar{q}, O)$. We only allow projective predicates O for observations. It is conceivable that it can also be based on more general predicates $O \in \mathcal{P}(\mathcal{H})$ but it is not clear what the intuitive operational meaning of such O would be, so we choose to pursue the simpler case.

We use **if** $M[\bar{q}] = 1$ **then** S_1 **else** S_0 as syntactic sugar for a measurement statement with $M = \{M_0, M_1\}$ and $\bar{S} = \{S_0, S_1\}$.

By $\equiv$ we denote syntactic equality of quantum programs. We use $\mathit{var}(S)$ to denote the set of variables occurring in program S. The Hilbert space of $\mathit{var}(S)$ is denoted by $\mathcal{H}_{all}$. If the set of variables is clear from the context, we just write $\mathcal{H}$.

For $\bar{q} = q_1, ..., q_n$ and operator A on $\mathcal{H}_{\bar{q}}$, we define its cylinder extension by $A \otimes I_{Var\setminus\{\bar{q}\}}$ on $\mathcal{H}_{all}$ and abbreviate it by A if it is clear from the context. Let $|\phi\rangle\langle\psi|_q$ denote the value of quantum variable q in the state $|\phi\rangle\langle\psi|$. We sometimes refer to it meaning its cylinder extension on $\mathcal{H}_{all}$ [27]. This notation is equivalent to $q(|\phi\rangle\langle\psi|)$ in [25].

3.2 Semantics

In this section, we define an operational and denotational semantics for quantum while-programs with observations and show their equivalence.

3.2.1 Operational Semantics

We start by defining the operational semantics of a program S as a Markov chain inspired by [22] instead of non-deterministic relations in comparison to [27]. A quantum *configuration* is a tuple $\langle S, \rho\rangle$ with density operator $\rho \in \mathcal{D}(\mathcal{H})$. Note that we consider normalized density

$$\frac{}{\langle \mathbf{skip}, \sigma\rangle \xrightarrow{1} \langle \downarrow, \sigma\rangle} \qquad \frac{type(q) = Int \wedge \sigma' = \sum_{n\in\mathbb{Z}} |0\rangle \langle n|_q \, \sigma \, |n\rangle \langle 0|_q}{\langle q := 0, \sigma\rangle \xrightarrow{1} \langle \downarrow, \sigma'\rangle}$$

$$\frac{type(q) = Bool \wedge \sigma' = |0\rangle \langle 0|_q \, \sigma \, |0\rangle \langle 0|_q + |0\rangle \langle 1|_q \, \sigma \, |1\rangle \langle 0|_q}{\langle q := 0, \sigma\rangle \xrightarrow{1} \langle \downarrow, \sigma'\rangle} \qquad \frac{}{\langle \bar{q} := U\bar{q}, \sigma\rangle \xrightarrow{1} \langle \downarrow, U\sigma U^\dagger\rangle}$$

$$\frac{tr(O\sigma O^\dagger) > 0}{\langle \mathbf{observe}\ (\bar{q}, O), \sigma\rangle \xrightarrow{tr(O\sigma O^\dagger)} \langle \downarrow, \frac{O\sigma O^\dagger}{tr(O\sigma O^\dagger)}\rangle} \qquad \frac{tr(O\sigma O^\dagger) < 1}{\langle \mathbf{observe}\ (\bar{q}, O), \sigma\rangle \xrightarrow{1-tr(O\sigma O^\dagger)} \langle \lightning\rangle}$$

$$\frac{M = \{M_m\}_{m\in I} \wedge m \in I \wedge tr(M_m\sigma M_m^\dagger) > 0}{\langle \mathbf{measure}\ M[\bar{q}] : \bar{S}, \sigma\rangle \xrightarrow{tr(M_m\sigma M_m^\dagger)} \langle S_m, \frac{M_m\sigma M_m^\dagger}{tr(M_m\sigma M_m^\dagger)}\rangle} \qquad \frac{\langle S_1, \sigma\rangle \xrightarrow{p} \langle \lightning\rangle}{\langle S_1; S_2, \sigma\rangle \xrightarrow{p} \langle \lightning\rangle}$$

$$\frac{\langle S_1, \sigma\rangle \xrightarrow{p} \langle S_1', \sigma'\rangle}{\langle S_1; S_2, \sigma\rangle \xrightarrow{p} \langle S_1'; S_2, \sigma'\rangle} \qquad \frac{tr(M_0\sigma M_0^\dagger) > 0}{\langle \mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S, \sigma\rangle \xrightarrow{tr(M_0\sigma M_0^\dagger)} \langle \downarrow, \frac{M_0\sigma M_0^\dagger}{tr(M_0\sigma M_0^\dagger)}\rangle}$$

$$\frac{tr(M_1\sigma M_1^\dagger) > 0}{\langle \mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S, \sigma\rangle \xrightarrow{tr(M_1\sigma M_1^\dagger)} \langle S; \mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S, \frac{M_1\sigma M_1^\dagger}{tr(M_1\sigma M_1^\dagger)}\rangle}$$

$$\frac{}{\langle \lightning\rangle \xrightarrow{1} \langle sink\rangle} \qquad \frac{}{\langle \downarrow, \sigma\rangle \xrightarrow{1} \langle sink\rangle} \qquad \frac{}{\langle sink\rangle \xrightarrow{1} \langle sink\rangle}$$

Figure 1 Transition probability function of MC $\mathfrak{R}_\rho[\![S]\!]$ for all $\sigma \in \mathcal{D}(\mathcal{H})$ where $\downarrow; S_2 \equiv S_2$

operators $\mathcal{D}(\mathcal{H})$ instead of partial density operators $\mathcal{D}^-(\mathcal{H})$. Intuitively, S is the program that is left to evaluate and ρ is the current state. We use $\downarrow$ to denote that there is no program left to evaluate. The set of all configurations over $\mathcal{H}$ is denoted as $\mathcal{C}(\mathcal{H})$. The quantum configuration for violated observations is $\langle \lightning\rangle$ and for termination is $\langle sink\rangle$.

▶ **Definition 2.** *The* operational semantics *of a program S with initial state $\rho \in \mathcal{D}(\mathcal{H})$ is defined as the Markov chain $\mathfrak{R}_\rho[\![S]\!] = (\Sigma, \boldsymbol{P}, s_{init})$ where:*

- $\Sigma = \mathcal{C}(\mathcal{H}) \cup \{\langle \lightning\rangle, \langle sink\rangle\}$,
- $s_{init} = \langle S, \rho\rangle$,
- *$\boldsymbol{P}$ is the smallest function satisfying the inference rules in Figure 1 where $c \xrightarrow{p} c'$ means $\boldsymbol{P}(c, c') = p > 0$. For all other pairs of states the transition probability is* 0.

The meaning of a transition $\langle S, \sigma\rangle \xrightarrow{p} \langle S', \sigma'\rangle$ is that after evaluating program S on state σ, with probability p the new state is σ' and the program left to execute is S'. For the observe statement, there are two successors, $\langle \mathbf{observe}\ (\bar{q}, O), \sigma\rangle \xrightarrow{tr(O\sigma O^\dagger)} \langle \downarrow, \frac{O\sigma O^\dagger}{tr(O\sigma O^\dagger)}\rangle$ and $\langle \mathbf{observe}\ (\bar{q}, O), \sigma\rangle \xrightarrow{1-tr(O\sigma O^\dagger)} \langle \lightning\rangle$. The observation O is satisfied by state σ with probability $tr(O\sigma O^\dagger)$ and then it terminates successfully. If the observation is violated (with probability $1 - tr(O\sigma O^\dagger)$), the successor state is $\langle \lightning\rangle$, the state that captures paths with violated observations. For details of the other rules we refer to [27].

3.2.2 Denotational Semantics

We now provide a denotational semantics for quantum while-programs. To handle observations and distinguish between non-terminating runs and those that violate observations, we introduce denotational semantics in a slightly different way than [27]. To do so, we start with defining some basics:

For tuples $(\rho, p), (\sigma, q) \in \mathcal{D}^-(\mathcal{H}) \times \mathbb{R}_{\geq 0}$, we define multiplication with a constant $a \in \mathbb{R}_{\geq 0}$ and addition entrywise: $a(\rho, p) := (a\rho, ap)$ and $(\rho, p) + (\sigma, q) := (\rho + \sigma, p + q)$.

The least upper bound (lub) of a set of tuples is defined as the entrywise lub provided it exists, i.e., $\bigvee_{n=0}^{\infty}(\rho_n, p_n) := (\bigvee_{n=0}^{\infty} \rho_n, \bigvee_{n=0}^{\infty} p_n)$ where $\bigvee_{n=0}^{\infty} \rho_n$ is the lub w.r.t. the Loewner partial order $\sqsubseteq$ and $\bigvee_{n=0}^{\infty} p_n$ is the lub w.r.t. to the classical ordering $\leq$ on $\mathbb{R}_{\geq 0}$.

As the probability of violating observations depends on the density operator, we introduce $\mathcal{DR} = \{(\rho, p) \in \mathcal{D}^-(\mathcal{H}) \times \mathbb{R}_{\geq 0} \mid tr(\rho) + p \leq 1\} \subseteq T(\mathcal{H}) \times \mathbb{C}$. $T(\mathcal{H}) \times \mathbb{C}$ is isomorphic to the set of operators of the form $\begin{pmatrix} \rho & \\ & p \end{pmatrix} \in T(\mathcal{H} \otimes \mathbb{C})$. Thus the trace and the norm from $T(\mathcal{H} \otimes \mathbb{C})$ apply. Specifically, $\tilde{tr}(\rho, p) := tr(\rho) + p$ and $\|(\rho, p)\| := \|\rho\| + |p|$ for $(\rho, p) \in T(\mathcal{H}) \times \mathbb{C}$.

▶ **Definition 3.** *The* denotational semantics *of a quantum program S is defined as a mapping $[\![S]\!] : \mathcal{DR} \to \mathcal{DR}$. For $(\rho, p) \in \mathcal{DR}$, ρ is used for density-transformer semantics as defined in [27] and p for the probability of an observation violation.*

The denotational semantics for $(\rho, p) \in \mathcal{DR}$ is given by

- $[\![\mathbf{skip}]\!](\rho, p) = (\rho, p)$.
- $[\![q := 0]\!](\rho, p) = \begin{cases} (|0\rangle \langle 0|_q \rho |0\rangle \langle 0|_q + |0\rangle \langle 1|_q \rho |1\rangle \langle 0|_q, p) & \textit{, if } type(q) = Bool \\ (\sum_{n \in \mathbb{Z}} |0\rangle \langle n|_q \rho |n\rangle \langle 0|_q, p) & \textit{, if } type(q) = Int. \end{cases}$
- $[\![\bar{q} := U\bar{q}]\!](\rho, p) = (U\rho U^\dagger, p)$.
- $[\![\mathbf{observe}\ (\bar{q}, O)]\!](\rho, p) = (O\rho O^\dagger, p + tr(\rho) - tr(O\rho O^\dagger))$.
- $[\![S_1; S_2]\!](\rho, p) = [\![S_2]\!]([\![S_1]\!](\rho, p))$.
- $[\![\mathbf{measure}\ M[\bar{q}] : \bar{S}]\!](\rho, p) = \sum_m [\![S_m]\!](M_m \rho M_m^\dagger, 0) + (\mathbf{0}, p)$ *with* $M = \{M_m\}_{m \in I}$ *and* $\bar{S} = \{S_m\}_{m \in I}$.
- $[\![\mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S]\!](\rho, p) = \bigvee_{n=0}^{\infty} [\![(\mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S)^n]\!](\rho, p)$ *with* $M = \{M_0, M_1\}$ *where loop unfoldings are defined inductively*

$$(\mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S)^0 \equiv \Omega$$
$$(\mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S)^{n+1} \equiv \mathbf{if}\ M[\bar{q}] = 1\ \mathbf{then}\ S; (\mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S)^n\ \mathbf{else\ skip}$$

where Ω is a syntactic quantum program with $[\![\Omega]\!](\rho, p) = (\mathbf{0}, p)$ as in [27].

We write $[\![S]\!]_\rho(\rho, p)$ and $[\![S]\!]_{\lightning}(\rho, p)$ to denote the first/second component of $[\![S]\!](\rho, p)$. It follows directly that our definition is a conservative extension of [27]:

▶ **Proposition 4.** *For an observe-free program S, input state $\rho \in \mathcal{D}^-(\mathcal{H})$ and $p \in \mathbb{R}_{\geq 0}$, is $[\![S]\!](\rho, p) = ([\![S]\!]_{og}(\rho), p)$ where $[\![S]\!]_{og}(\rho)$ is the denotational semantics as defined in [27].*

Some intuition behind those tuples: If $[\![S]\!](\rho, 0) = (\rho', p')$ for a program S with initial pair $(\rho, 0)$, then the probability of violating an observation while executing S on $\rho \in \mathcal{D}(\mathcal{H})$ is p'. The probability of terminating normally (without violating an observation) is given by $tr(\rho')$ and the probability for non-termination is $1 - tr(\rho') - p'$. As in the observe-free case, ρ' is the (non-normalized) state after S has been executed (and terminated) on ρ. It is easy to see that only the observation statement can change the value of the second entry.

▶ **Proposition 5.** *For $(\rho, p), (\rho, q) \in \mathcal{DR}$ and program S:*

1. $[\![S]\!]_\rho(\rho, p) = [\![S]\!]_\rho(\rho, q)$
2. $p \leq [\![S]\!]_{\lightning}(\rho, p)$
3. *if* $(\rho, q + p) \in \mathcal{DR}$ *then* $[\![S]\!]_{\lightning}(\rho, q + p) = [\![S]\!]_{\lightning}(\rho, q) + p$
4. $\tilde{tr}([\![S]\!](\rho, p)) \leq \tilde{tr}(\rho, p)$
5. $[\![S]\!]$ *is well defined, i.e.,* $[\![S]\!](\rho, p) \in \mathcal{DR}$ *and the least upper bound exists.*
6. $[\![S]\!]$ *is linear*

Proof. TOPROVE 0 ◀

As $[\![S]\!]_\rho(\rho, p) = [\![S]\!]_\rho(\rho, q)$, we use $[\![S]\!]_\rho(\rho)$ instead. Three consequences of Proposition 5:

▶ **Lemma 6.** *For $(\rho, p), (\sigma, q) \in \mathcal{DR}$ with $(\rho + \sigma, p + q) \in \mathcal{DR}$ and programs S, S_1, S_2:*
1. $tr([\![S]\!]_\rho(\rho, p)) \leq tr(\rho)$, *i.e.,* $[\![S]\!]_\rho$ *is trace-reducing*
2. $[\![S_1; S_2]\!]_\lightning(\rho, q + p) = [\![S_2]\!]_\lightning([\![S_1]\!]_\rho(\rho, 0), q) + [\![S_1]\!]_\lightning(\rho, p)$
3. $[\![S]\!]_\rho$ *is bounded linear*

The proof can be found in the Appendix A.1.

3.2.3 Equivalence of Semantics

The following lemma asserts the equivalence of our operational and denotational semantics. Intuitively, the denotational semantics gives a distribution over final states and its second component captures the probability to reach $\langle \lightning \rangle$, the state for violated observations. As the operational semantics is only defined for $tr(\rho) = 1$, we only consider this case:

▶ **Lemma 7.** *For any program S and initial state $\rho \in \mathcal{D}(\mathcal{H})$*
- $[\![S]\!](\rho, 0) = (\sum_{\rho'} Pr^{\mathfrak{R}_\rho[\![S]\!]}(\Diamond\langle\downarrow, \rho'\rangle) \cdot \rho', Pr^{\mathfrak{R}_\rho[\![S]\!]}(\Diamond\langle\lightning\rangle))$
- $Pr^{\mathfrak{R}_\rho[\![S]\!]}(\Diamond\langle sink\rangle) = tr([\![S]\!]_\rho(\rho, 0)) + [\![S]\!]_\lightning(\rho, 0)$

Proof. TOPROVE 1 ◀

4 Weakest Preconditions

In this section, we consider how we can extend the weakest precondition calculus to capture observations and thus compute conditional probabilities of quantum programs using deductive verification. Recall that a predicate P satisfies $\mathbf{0} \sqsubseteq P \sqsubseteq \mathbf{I}$. Let $tr(P\rho)$ by the probability that ρ satisfies P. Note that if P is a projector, then $tr(P\rho)$ equals the probability that ρ gives answer "yes" in a measurement defined by P. Even if P is not a projection, $tr(P\rho)$ is the average value of measuring ρ with the measurement described by the observable P. If not given directly, all proofs can be found in the Appendix A.2.

4.1 Total and Partial Correctness

Defining the semantics in a different way also changes the definition of Hoare logic with total and partial correctness [27]:

▶ **Definition 8.** *Let $P, Q \in \mathcal{P}(\mathcal{H})$, S a program, $\rho \in \mathcal{D}^-(\mathcal{H})$ and $\{P\}S\{Q\}$ a correctness formula. Then*
1. *(total correctness)* $\models_{tot} \{P\}S\{Q\}$ *iff* $tr(P\rho) \leq tr(Q[\![S]\!]_\rho(\rho, 0))$
2. *(partial correctness)* $\models_{par} \{P\}S\{Q\}$ *iff* $tr(P\rho) \leq tr(Q[\![S]\!]_\rho(\rho, 0)) + tr(\rho) - tr([\![S]\!]_\rho(\rho, 0)) - [\![S]\!]_\lightning(\rho, 0)$

Let us explain this definition. Assume $tr(\rho) = 1$, otherwise all probabilities mentioned in the following are non-normalized. Recall that $tr(P\rho)$ is the probability that state ρ satisfies predicate P and $tr(Q[\![S]\!]_\rho(\rho, 0))$ is the probability that the state after execution of S starting with ρ satisfies predicate Q. Total correctness entails that the probability of a state satisfying precondition P is at most the probability that it satisfies postcondition Q after execution of S. This only involves terminating runs. In the formula of partial correctness, the summand $[\![S]\!]_\lightning(\rho, 0)$ captures the probability that an observation is violated during executing program

S on state ρ. As before, $tr(\rho) - tr([\![S]\!]_\rho(\rho, 0))$ captures the probability that S on state ρ does not terminate.

Similar to [27], we have some nice but different properties:

▶ **Proposition 9.** 1. $\models_{tot} \{P\}S\{Q\}$ *implies* $\models_{par} \{P\}S\{Q\}$
2. $\models_{tot} \{\mathbf{0}\}S\{Q\}$. *However,* $\models_{par} \{P\}S\{\boldsymbol{I}\}$ *does not hold in general.*
3. *For* $P_1, P_2, Q_1, Q_2 \in \mathcal{P}(\mathcal{H})$ *and* $\lambda_1, \lambda_2 \in \mathbb{R}_{\geq 0}$ *with* $\lambda_1 P_1 + \lambda_2 P_2, \lambda_1 Q_1 + \lambda_2 Q_2 \in \mathcal{P}(\mathcal{H})$*:* $\models_{tot} \{P_1\}S\{Q_1\} \wedge \models_{tot} \{P_2\}S\{Q_2\}$ *implies* $\models_{tot} \{\lambda_1 P_1 + \lambda_2 P_2\}S\{\lambda_1 Q_1 + \lambda_2 Q_2\}$

Proof. TOPROVE 2 ◀

4.2 Weakest (Liberal) Preconditions

Given a postcondition and a program, we are interested in the best (weakest) precondition w.r.t. total and partial correctness:

▶ **Definition 10.** *Let program S and predicate $P \in \mathcal{P}(\mathcal{H})$.*
1. *The* weakest precondition *is defined as* $qwp[\![S]\!](P) = \sup\{Q \mid \models_{tot} \{Q\}S\{P\}\}$. *Thus* $\models_{tot} \{qwp[\![S]\!](P)\}S\{P\}$ *and* $\models_{tot} \{Q\}S\{P\}$ *implies* $Q \sqsubseteq qwp[\![S]\!](P)$ *for all* $Q \in \mathcal{P}(\mathcal{H})$.
2. *The* weakest liberal precondition *is defined as* $qwlp[\![S]\!](P) = \sup\{Q \mid \models_{par} \{Q\}S\{P\}\}$. *Thus* $\models_{par} \{qwlp[\![S]\!](P)\}S\{P\}$ *and* $\models_{par} \{Q\}S\{P\}$ *implies* $Q \sqsubseteq qwlp[\![S]\!](P)$ *for all* $Q \in \mathcal{P}(\mathcal{H})$.

The following lemmas show that these suprema indeed exist. Both proofs are based on the Schrödinger-Heisenberg duality [25].

▶ **Lemma 11.** *For a function* $[\![S]\!] : \mathcal{DR} \to \mathcal{DR}$ *with properties as in Proposition 5, the weakest precondition* $qwp[\![S]\!] : \mathcal{P}(\mathcal{H}) \to \mathcal{P}(\mathcal{H})$ *exists and is bounded linear and subunital. It satisfies* $tr(qwp[\![S]\!](P)\rho) = tr(P[\![S]\!]_\rho(\rho, 0))$ *for all* $\rho \in \mathcal{D}^-(\mathcal{H}), P \in \mathcal{P}(\mathcal{H})$ *and it is the only function of this type with this property.*

This lemma (and the following one) does not require $[\![S]\!]$ to be a denotational semantics of some program S. In contrast to [27], this result thus still holds if the language is extended as long as the conditions still holds.

▶ **Lemma 12.** *For a function* $[\![S]\!] : \mathcal{DR} \to \mathcal{DR}$ *with properties as in Proposition 5, the weakest liberal precondition* $qwlp[\![S]\!] : \mathcal{P}(\mathcal{H}) \to \mathcal{P}(\mathcal{H})$ *exists and is subunital. It satisfies*

$$tr(qwlp[\![S]\!](P)\rho) = tr(P[\![S]\!]_\rho(\rho, 0)) + tr(\rho) - tr([\![S]\!]_\rho(\rho, 0)) - [\![S]\!]_{\lightning}(\rho, 0)$$

for each $\rho \in \mathcal{D}^-(\mathcal{H}), P \in \mathcal{P}(\mathcal{H})$ *and it is the only function of this type with this property.*

This general theorem about the existence of weakest liberal preconditions also applies for programs without observations (because $[\![S]\!]_{\lightning}(\rho, 0) = 0$ and $[\![S]\!]_\rho(\rho, 0) = [\![S]\!]_{og}(\rho)$ for each ρ for observation-free program S, Proposition 4). Lemma 11 and 12 extend [7] to the infinite-dimensional case and to partial correctness, i.e., the existence of weakest liberal preconditions. Now we consider some healthiness properties about weakest (liberal) preconditions:

▶ **Proposition 13.** *For every program S, the function* $qwp[\![S]\!] : \mathcal{P}(\mathcal{H}) \to \mathcal{P}(\mathcal{H})$ *satisfies:*
- *Bounded linearity*
- *Subunitality:* $qwp[\![S]\!](\boldsymbol{I}) \sqsubseteq \boldsymbol{I}$
- *Monotonicity:* $P \sqsubseteq Q$ *implies* $qwp[\![S]\!](P) \sqsubseteq qwp[\![S]\!](Q)$
- *Order-continuity:* $qwp[\![S]\!](\bigvee_{i=0}^{\infty} P_i) = \bigvee_{i=0}^{\infty} qwp[\![S]\!](P_i)$ *if* $\bigvee_{i=0}^{\infty} P_i$ *exists*

▶ **Proposition 14.** *For every program S, the function $qwlp[\![S]\!] : \mathcal{P}(\mathcal{H}) \to \mathcal{P}(\mathcal{H})$ satisfies:*

- *Affinity: The function $f : \mathcal{P}(\mathcal{H}) \to \mathcal{P}(\mathcal{H})$ with $f(P) = qwlp[\![S]\!](P) - qwlp[\![S]\!](\mathbf{0})$ is linear. Note that this implies convex-linearity and sublinearity.*
- *Subunitality: $qwlp[\![S]\!](\boldsymbol{I}) \sqsubseteq \boldsymbol{I}$*
- *Monotonicity: $P \sqsubseteq Q$ implies $qwlp[\![S]\!](P) \sqsubseteq qwlp[\![S]\!](Q)$*
- *Order-continuity: $qwlp[\![S]\!](\bigvee_{i=0}^{\infty} P_i) = \bigvee_{i=0}^{\infty} qwlp[\![S]\!](P_i)$ if $\bigvee_{i=0}^{\infty} P_i$ exists*

For our denotational semantics $[\![S]\!]$, we can also give an explicit representation of $qwp[\![S]\!]$:

▶ **Proposition 15.** *Let $P \in \mathcal{P}(\mathcal{H})$:*

- $qwp[\![\mathbf{skip}]\!](P) = P$
- $qwp[\![q := 0]\!](P) = \begin{cases} |0\rangle_q \langle 0| P |0\rangle \langle 0|_q + |1\rangle \langle 0|_q P |0\rangle \langle 1|_q & \text{, if } type(q) = Bool \\ \sum_{n \in \mathbb{Z}} |n\rangle \langle 0|_q P |0\rangle \langle n|_q & \text{, if } type(q) = Int \end{cases}$
- $qwp[\![\bar{q} := U\bar{q}]\!](P) = U^\dagger P U$
- $qwp[\![\mathbf{observe}\ (\bar{q}, O)]\!](P) = O^\dagger P O$
- $qwp[\![S_1; S_2]\!](P) = qwp[\![S_1]\!](qwp[\![S_2]\!](P))$
- $qwp[\![\mathbf{measure}\ M[\bar{q}] : \bar{S}]\!](P)\rho = \sum_m M_m^\dagger (qwp[\![S_m]\!](P)) M_m$
- $qwp[\![\mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S']\!](P) = \bigvee_{n=0}^{\infty} P_n$ *with*

$$P_0 = \mathbf{0}, \qquad P_{n+1} = [M_0^\dagger P M_0] + [M_1^\dagger (qwp[\![S']\!](P_n)) M_1]$$

and $\bigvee_{n=0}^{\infty}$ denoting the least upper bound w.r.t. $\sqsubseteq$.

Proof. TOPROVE 3 ◀

In this and the following proposition we mean convergence of sums with respect to the SOT, more details can be found in [25]. For weakest liberal preconditions the explicit representation looks quite similar:

▶ **Proposition 16.** *Let $P \in \mathcal{P}(\mathcal{H})$. For most cases, $qwlp[\![S]\!](P)$ is defined analogous to $qwp[\![S]\!](P)$ (replacing every occurrence of qwp by qwlp). The only significant difference is the while-loop: $qwlp[\![\mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S']\!](P) = \bigwedge_{n=0}^{\infty} P_n$ with*

$$P_0 = \boldsymbol{I}, \qquad P_{n+1} = [M_0^\dagger P M_0] + [M_1^\dagger (qwlp[\![S']\!](P_n)) M_1]$$

and $\bigwedge_{n=0}^{\infty}$ denoting the greatest lower bound w.r.t. $\sqsubseteq$.

Proof. TOPROVE 4 ◀

Both explicit representations above are conservative extensions of the weakest (liberal) precondition calculus in [27].

For the following explanations, assume $tr(\rho) = 1$, otherwise the probabilities are not normalized. To understand those definitions, consider $tr(qwp[\![S]\!](P)\rho)$. Due to the duality from Lemma 11, $tr(qwp[\![S]\!](P)\rho) = tr(P[\![S]\!]_\rho(\rho, 0))$, so it is the probability that the result of running program S (without violating any observations) on the initial state ρ satisfies predicate P. Similarly, $tr(qwlp[\![S]\!](P)\rho)$ adds the probability of non-termination too. This is equivalent to the standard interpretation of weakest (liberal) preconditions in [27].

For programs with observations $tr(qwlp[\![S]\!](P)\rho)$ does not include runs that violate an observation. Thus, $tr(qwlp[\![S]\!](\mathbf{I})\rho)$ gives the probability that no observation is violated during the run of S on input state ρ (while for programs without observations and in [27], this will always be $tr(\rho) = 1$). The probability that a program state ρ will satisfy the postcondition P after executing program S while not violating any observation is then a conditional probability. To handle this case, we introduce conditional weakest preconditions inspired by [22] in the next section.

4.3 Conditional Weakest Preconditions

In the following, we consider pairs of predicates. Addition and multiplication are interpreted pointwise, i.e., $(P, Q) + (P', Q') = (P + P', Q + Q')$ and $M \cdot (P, Q) = (MP, MQ)$ resp. $(P, Q) \cdot M = (PM, QM)$ where M can be a constant or an operator. Multiplication binds stronger than addition.

We define a natural ordering on pairs of predicates that is used for example to express healthiness conditions:

▶ **Definition 17.** *We define* $\trianglelefteq$ *on* $\mathcal{P}(\mathcal{H})^2$ *as follows:* $(P, Q) \trianglelefteq (P', Q') \Leftrightarrow P \sqsubseteq P' \wedge Q' \sqsubseteq Q$ *where* $\sqsubseteq$ *is the Loewner partial order. The least element is* $(\mathbf{0}, \boldsymbol{I})$ *and the greatest element* $(\boldsymbol{I}, \mathbf{0})$*. The least upper bound of an increasing chain* $\{(P_i, Q_i)\}_{i\in\mathbb{N}}$ *for* $(P_i, Q_i) \in \mathcal{P}(\mathcal{H})^2$ *is given pointwise by* $\bigvee_{i=0}^{\infty}(P_i, Q_i) = (\bigvee_{i=0}^{\infty} P_i, \bigwedge_{i=0}^{\infty} Q_i)$*.*

▶ **Lemma 18.** $\trianglelefteq$ *is an* ω*-cpo on* $\mathcal{P}(\mathcal{H})^2$*.*

Proof. TOPROVE 5 ◀

Combining weakest preconditions and liberal weakest preconditions, we can define *conditional weakest preconditions* similar to the probabilistic case [22]:

▶ **Definition 19.** *The* conditional weakest precondition *transformer is a mapping* $qcwp[\![S]\!] : \mathcal{P}(\mathcal{H})^2 \to \mathcal{P}(\mathcal{H})^2$ *defined as* $qcwp[\![S]\!](P, Q) := (qwp[\![S]\!](P), qwlp[\![S]\!](Q))$*.*

Similar to the weakest precondition calculus, we can also give an explicit representation which can be found in the Appendix A.2, Lemma 29.

Again, assume $tr(\rho) = 1$ in the following, otherwise the probabilities are not normalized. Note that $tr(qwlp[\![S]\!](\mathbf{I})\rho)$ is the probability that no observation is violated and $tr(qwp[\![S]\!](P)\rho)$ the probability that P is satisfied after S has been executed on ρ (see above). We are interested in the conditional probability of establishing the postcondition given that no observation is violated, namely $\frac{tr(qwp[\![S]\!](P)\rho)}{tr(qwlp[\![S]\!](\mathbf{I})\rho)}$. Notice that for $qcwp[\![S]\!]((P, \mathbf{I})) = (A, B)$, this is simply $\frac{tr(A\rho)}{tr(B\rho)}$. That means we can immediately read of this conditional probability from $qcwp[\![S]\!]$. Formally, we use

$$\hat{tr}(A\rho, B\rho) := \begin{cases} \frac{tr(A\rho)}{tr(B\rho)}, & \text{if } tr(B\rho) \neq 0 \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

We now establish some properties of conditional weakest preconditions:

▶ **Proposition 20.** *For every program* S*, the function* $qcwp[\![S]\!] : \mathcal{P}(\mathcal{H})^2 \to \mathcal{P}(\mathcal{H})^2$ *satisfies:*

- *Has a linear interpretation: for all* $\rho \in \mathcal{D}^-(\mathcal{H}), a, b \in \mathbb{R}_{\geq 0}$ *and* $P, P' \in \mathcal{P}(\mathcal{H})$ *with* $aP + bP' \in \mathcal{P}(\mathcal{H})$

$$\hat{tr}(qcwp[\![S]\!](aP + bP', \boldsymbol{I}) \cdot \rho) = a \cdot \hat{tr}(qcwp[\![S]\!](P, \boldsymbol{I}) \cdot \rho) + b \cdot \hat{tr}(qcwp[\![S]\!](P', \boldsymbol{I}) \cdot \rho)$$

- *Affinity: The function* $qcwp[\![S]\!](P, Q) - qcwp[\![S]\!](\mathbf{0}, \mathbf{0})$ *is linear. Note that this implies convex-linearity and sublinearity.*
- *Monotonicity:* $(P, P') \trianglelefteq (Q, Q')$ *implies* $qcwp[\![S]\!](P, P') \trianglelefteq qcwp[\![S]\!](Q, Q')$
- *Continuity:* $qcwp[\![S]\!](\bigvee_{i=0}^{\infty}(P_i, Q_i)) = \bigvee_{i=0}^{\infty} qcwp[\![S]\!](P_i, Q_i)$ *if* $\bigvee_{i=0}^{\infty}(P_i, Q_i)$ *exists*

4.4 Conditional Weakest Liberal Preconditions

Similar to the conditional weakest precondition, we can also define the same with weakest liberal preconditions for partial correctness:

▶ **Definition 21.** *The* conditional weakest liberal precondition $qcwlp : \mathcal{P}(\mathcal{H})^2 \to \mathcal{P}(\mathcal{H})^2$ *is defined as* $qcwlp[\![S]\!](P,Q) := (qwlp[\![S]\!](P), qwlp[\![S]\!](Q))$ *for each program S and predicates* $P,Q \in \mathcal{P}(\mathcal{H})$.

▶ **Definition 22.** *We define $\dot{\trianglelefteq}$ on $\mathcal{P}(\mathcal{H})^2$ as follows $(P,Q) \mathrel{\dot{\trianglelefteq}} (P',Q') \Leftrightarrow P \sqsubseteq P' \wedge Q \sqsubseteq Q'$ where $\sqsubseteq$ is the Loewner partial order. The least element is $(\mathbf{0},\mathbf{0})$ and the greatest element $(\boldsymbol{I},\boldsymbol{I})$. The least upper bound of an increasing chain $\{(P_i,Q_i)\}_{i\in\mathbb{N}}$ for $(P_i,Q_i) \in \mathcal{P}(\mathcal{H})^2$ is given pointwise by $\bigvee_{i=0}^{\infty}(P_i,Q_i) = (\bigvee_{i=0}^{\infty} P_i, \bigvee_{i=0}^{\infty} Q_i)$.*

Note that in contrast to $\trianglelefteq$, both components are ordered in the same direction. Here it follows directly that $\dot{\trianglelefteq}$ is an ω-cpo on $\mathcal{P}(\mathcal{H})^2$.

Similar as for *qcwp*, we can now read off the conditional satisfaction of P when we want non-termination to count as satisfaction: $\hat{tr}(qcwlp[\![S]\!](P,\mathbf{I}) \cdot \rho) = \frac{tr(qwlp[\![S]\!](P)\rho)}{tr(qwlp[\![S]\!](\mathbf{I})\rho)}$ which is equal to dividing the probability to satisfy P after execution (including non-termination) by the probability to not violate an observation*.

We can also conclude some properties about conditional weakest liberal preconditions:

▶ **Proposition 23.** *For every program S, the function $qcwlp[\![S]\!] : \mathcal{P}(\mathcal{H})^2 \to \mathcal{P}(\mathcal{H})^2$ satisfies:*

- *Affinity: The function $qcwlp[\![S]\!](P,Q) - qcwlp[\![S]\!](\mathbf{0},\mathbf{0})$ is linear. Note that this implies convex-linearity and sublinearity.*
- *Monotonicity: $(P,Q) \mathrel{\dot{\trianglelefteq}} (P',Q')$ implies $qcwlp[\![S]\!](P,Q) \mathrel{\dot{\trianglelefteq}} qcwlp[\![S]\!](P',Q')$*
- *Continuity: $qcwlp[\![S]\!](\bigvee_{i=0}^{\infty}(P_i,Q_i)) = \bigvee_{i=0}^{\infty} qcwlp[\![S]\!](P_i,Q_i)$ if $\bigvee_{i=0}^{\infty}(P_i,Q_i)$*

4.5 Observation-Free Programs

For observation-free programs, our interpretations coincides with the satisfaction of weakest (liberal) preconditions from [27]:

▶ **Lemma 24.** *For an observation-free program S, predicate $P \in \mathcal{P}(\mathcal{H})$ and state $\rho \in \mathcal{D}(\mathcal{H})$:*

$$\hat{tr}(qcw(l)p[\![S]\!](P,\boldsymbol{I}) \cdot \rho) = tr(qw(l)p[\![S]\!](P)\rho)$$

Proof. TOPROVE 6 ◀

4.6 Correspondence to Operational MC Semantics

The aim of this section is to establish a correspondence between $qcwp[\![S]\!](P,\mathbf{I})$ and the operational semantics of S. In order to reason about P in terminal states of the Markov chain, we use rewards. First, we equip the Markov chain used for the operational semantics with a reward function with regard to a postcondition P:

---

*So far, we considered conditional weakest preconditions for total and partial correctness, i.e., $qcwp[\![S]\!](P,\mathbf{I}) = (qwp[\![S]\!](P), qwlp[\![S]\!](\mathbf{I}))$ and $qcwlp[\![S]\!](P,\mathbf{I}) = (qwlp[\![S]\!](P), qwlp[\![S]\!](\mathbf{I}))$. In [14, Sect. 8.3] it is argued why other combinations such as $(qwp[\![S]\!](P), qwp[\![S]\!](\mathbf{I}))$ and $(qwlp[\![S]\!](P), qwp[\![S]\!](\mathbf{I}))$ only make sense if a program is almost-surely terminating, i.e., without non-termination. Their arguments apply without change in our setting, so we do not consider these combinations either.

▶ **Definition 25.** *For program S and postcondition P, the Markov reward chain $\mathfrak{R}_\rho^P[\![S]\!]$ is the MC $\mathfrak{R}_\rho[\![S]\!]$ extended with a function $r : \Sigma \to \mathbb{R}_{\geq 0}$ such that $r(\langle\downarrow, \rho'\rangle) = tr(P\rho')$ and $r(s) = 0$ for all other states $s \in \Sigma$.*

The (liberal) reward of a path π of $\mathfrak{R}_\rho^P[\![S]\!]$ is defined as $r(\pi) = \begin{cases} tr(P\rho') & \text{, if } \langle\downarrow, \rho'\rangle \in \pi \\ 0 & \text{, else} \end{cases}$ and $lr(\pi) = r(\pi)$ expect if $\langle sink\rangle \notin \pi$, then $lr(\pi) = 1$.

The expected reward of $\Diamond\langle sink\rangle$ is the expected value of $r(\pi)$ for all $\pi \in \Diamond\langle sink\rangle$, i.e., $ER^{\mathfrak{R}_\rho^P[\![S]\!]}(\Diamond\langle sink\rangle) = \sum_{\rho'} Pr^{\mathfrak{R}_\rho^P[\![S]\!]}(\Diamond\langle\downarrow, \rho'\rangle) \cdot tr(P\rho')$. The liberal version adds rewards of non-terminating paths, i.e., $LER^{\mathfrak{R}_\rho^P[\![S]\!]}(\Diamond\langle sink\rangle) = ER^{\mathfrak{R}_\rho^P[\![S]\!]}(\Diamond\langle sink\rangle) + Pr^{\mathfrak{R}_\rho^P[\![S]\!]}(\neg\Diamond\langle sink\rangle)$.

Now we start by showing some auxiliary results, similar to [22, Lemma 5.5, 5.6]:

▶ **Lemma 26.** *For a program S, state $\rho \in \mathcal{D}(\mathcal{H})$, predicate $P \in \mathcal{P}(\mathcal{H})$ we have*

$$Pr^{\mathfrak{R}_\rho^P[\![S]\!]}(\neg\Diamond\langle\lightning\rangle) = tr(qwlp[\![S]\!](\boldsymbol{I})\rho), \qquad (L)ER^{\mathfrak{R}_\rho^P[\![S]\!]}(\Diamond\langle sink\rangle) = tr(qw(l)p[\![S]\!](P)\rho)$$

Proof. TOPROVE 7 ◀

We are interested in the conditional (liberal) expected reward of reaching $\langle sink\rangle$ from the initial state $\langle S, \rho\rangle$, conditioned on not visiting $\langle\lightning\rangle$:

$$C(L)ER^{\mathfrak{R}_\rho^P[\![S]\!]}(\Diamond\langle sink\rangle \mid \neg\Diamond\langle\lightning\rangle) := \frac{(L)ER^{\mathfrak{R}_\rho^P[\![S]\!]}(\Diamond\langle sink\rangle)}{Pr^{\mathfrak{R}_\rho^{\mathbf{I}}[\![S]\!]}(\neg\Diamond\langle\lightning\rangle)}$$

This reward is equivalent to our interpretation of $qcw(l)p$, analogous to [22, Theorem 5.7]:

▶ **Theorem 27.** *For a program S, state $\rho \in \mathcal{D}(\mathcal{H})$, predicates $P, Q \in \mathcal{P}(\mathcal{H})$ we have*

$$C(L)ER^{\mathfrak{R}_\rho^P[\![S]\!]}(\Diamond\langle sink\rangle \mid \neg\Diamond\langle\lightning\rangle) = \hat{tr}(qcw(l)p[\![S]\!](P, \boldsymbol{I}) \cdot \rho)$$

Proof. TOPROVE 8 ◀

5 Examples

In this section we provide two examples on how conditional quantum weakest preconditions can be applied.

5.1 The Quantum Fast-Dice-Roller

In probabilistic programs, generating a uniform distribution using fair coins is a challenge. The fast dice roller efficiently simulates the throw of a fair dice, generating a uniform distribution about N possible outcomes [18]. We solve this problem for $N = 6$ with quantum gates by creating qubits q, p, r with Hadamard gates and using the observe statement to reject the $qp = 11$ case, leaving 6 possible outcomes ($qpr = 000, \ldots, 101$), see Figure 2.

Before verifying its correctness, we consider the operational semantics: We have three binary variables, so $\mathcal{H}_{all} = \mathcal{H}_2^{\otimes 3}$, denoted as $\mathcal{H}$. The first variable is q, the second one p and the last one r and $\rho_0 \in \mathcal{D}(\mathcal{H})$ an initial state. The operational semantics is represented by the Markov chain in Figure 5, Appendix A.3. To prove correctness, we focus on the probability of termination and reaching the desired state without violating the observation. This probability cannot be directly read from the operational semantics, even for this simple program. To specify this property formally, we use the reward MC as defined in Definition 25. The desired probability can be computed using conditional weakest preconditions, see Theorem 27.

$$\begin{aligned}
&q \ := \ Hq;\\
&p \ := \ Hp;\\
&\mathbf{observe}(q \otimes p, \mathbf{I}_4 - |11\rangle\langle 11|);\\
&r \ := \ Hr
\end{aligned}$$

Figure 2 Quantum Fast-Dice-Roller. For the identity operator on $\mathcal{H}_2 \otimes \mathcal{H}_2$ we use $\mathbf{I}_4$.

To terminate in a state where the probability of all six outcomes is equal and forms a distribution, we verify that we reach the uniform superposition $|\phi\rangle = \sqrt{\frac{1}{6}}\big(|000\rangle + |001\rangle + |010\rangle + |011\rangle + |100\rangle + |101\rangle\big)$ over 6 states. Measuring in the computational basis yields a uniform distribution. After computing the conditional weakest precondition, we can determine the likelihood of each input state reaching the fixed uniform superposition and producing a uniform distribution, assuming the observation is not violated. We use the decoupling of $qcwp[\![S]\!]((P, \mathbf{I}))$ and compute $qwp[\![S]\!](P)$ and $qwlp[\![S]\!](\mathbf{I})$ separately where $P = |\phi\rangle\langle\phi|$ and S stands for our fast-dice roller program (Figure 2). The results of applying rules of Proposition 15 and 16 can be found in Appendix A.3. The probability that an input state ρ will reach the desired uniform superposition is $\hat{tr}(qcwp[\![S]\!]((P, \mathbf{I})) \cdot \rho)$, that is

$$\frac{tr(qwp[\![S]\!](P)\rho)}{tr(qwlp[\![S]\!](\mathbf{I})\rho)} = \begin{cases} 1 & \text{, if } \rho = |000\rangle\langle 000| \\ 0 & \text{, if } \rho = |x\rangle\langle x| \text{ with } x \in \{001, 011, 101, 111\} \\ 0.1111 & \text{, if } \rho = |x\rangle\langle x| \text{ with } x \in \{010, 100, 110\}. \end{cases}$$

$\rho = |000\rangle\langle 000|$ will reach the desired superposition with probability 1 assuming no observation is violated. We can also see that $tr(qwp[\![S]\!](P)\,|000\rangle\langle 000|) \neq 1$ so even with the "best" input, our conditional weakest precondition calculus gives more information than $qwp[\![S]\!](P)$.

5.2 MAJ-SAT

To demonstrate our approach, we will verify the correctness of a program that is used to solve MAJ-SAT. Unlike SAT, which asks whether there exists at least one satisfying assignment of a Boolean formula, MAJ-SAT asks whether a Boolean formula is satisfied by at least half of all possible variable assignments. MAJ-SAT is known to be PP-complete and [1] uses it to prove the equivalence of the complexity classes PostBQP and PP.

Formally, we are faced with the following problem: A formula with n variables can be represented by a function $f : \{0,1\}^n \to \{0,1\}$ with $s = |\{f(x) = 1\}|$. The goal is to determine whether $s < 2^{n-1}$ holds or not. Aaronson [1] presents a PostBQP algorithm for this problem. A PostBQP algorithm is one that runs in polynomial time, is allowed to perform measurements to check whether certain conditions are satisfied (analogous to our observe statement) and is required to produce the correct result with high probability conditioned on those measurements succeeding. The algorithm from [1] is as follows:

for $k = -n, ..., n$:
  repeat n times:
    S_k
  if S_k succeeded more than $\frac{3}{4}n$ times :
    return true
return false

$$
\begin{array}{lll}
\overline{q} & := & 0^{\otimes n}; \\
y & := & 0; \\
\overline{q} & := & H^{\otimes n}\overline{q}; \\
\overline{q}y & := & U_f \overline{q}y; \\
\overline{q} & := & H^{\otimes n}\overline{q}; \\
\multicolumn{3}{l}{\mathbf{observe}(\overline{q}, |0\rangle\langle 0|^{\otimes n});} \\
z & := & 0; \\
z & := & R_k z; \\
zy & := & CH; \\
\multicolumn{3}{l}{\mathbf{observe}(y, |1\rangle\langle 1|)}
\end{array}
$$

Figure 3 Inner loop body S_k of the quantum algorithm solving MAJ-SAT as presented in [1]. y, z are qubits, $\overline{q}$ is an n-qubit sized register (formally n qubits $q_1, \dots, q_n$). We use $\overline{q} := 0^{\otimes n}$ to denote that all n qubits of $\overline{q}$ are set of 0. $R_k = \frac{1}{\sqrt{1+4^k}} \begin{pmatrix} 1 & -2^k \\ 2^k & 1 \end{pmatrix}$ is a rotation matrix depending on the parameter k and CH is a controlled Hadamard.

where S_k is given in Figure 3 and succeeding means that measuring z in the $|+\rangle, |-\rangle$ basis returns $|+\rangle$. The core idea is to show that S_k succeeds with probability $\leq \frac{1}{2}$ for all k if $s \geq 2^{n-1}$ and with probability $\geq \left(\frac{1+\sqrt{2}}{\sqrt{6}}\right)^2 \geq 0.971$ for at least one k otherwise. Hence the overall algorithm solves MAJ-SAT. To keep this example manageable, we focus on the analysis of S_k alone.

We use conditional weakest preconditions and determine $qcwp[\![S_k]\!](P, \mathbf{I}^{\otimes n+2})$ (which depends on the parameters n, s, k). Here the postcondition P corresponds to z being in state $|+\rangle$, formally $P = |+\rangle\langle +|_z \otimes \mathbf{I}$. Then the probability that S_k succeeds is $\Pr_{nsk} := \hat{tr}(qcwp[\![S_k]\!](P, \mathbf{I}^{\otimes n+2}) \odot \rho)$ for initial state ρ.

We computed the cwp symbolically using a computer algebra system, but the resulting formulas were quite unreadable. So for the sake of this example, we present numerical results of computing cwp instead. Since S_k does not contain any loops, the cwp can be computed by mechanic application of the rules for observation, assignment, and application of unitaries. Performing these calculations for selected values of n and s and all $k = -n, \dots, n$, we find that in each case the cwp is of the form $(c\mathbf{I}, c'\mathbf{I})$ for some $c, c' \in \mathbb{R}$. This is to be expected since all variables $\bar{q}, z, y$ are initialized at the beginning of the program, so the cwp should not depend on the initial state, i.e., all matrices should be multiples of the identity. In that case, $\Pr_{nsk} = c/c'$. In Figure 4, we show $\max_k \Pr_{nsk}$ for selected s, n. (The claim from [1] is that the success probability of S_k is ≥ 0.971 for some k if $s < 2^{n-1}$ and $\leq 1/2$ for all k otherwise, so we only care about the maximum over all k.) We see that $\max_k \Pr_{nsk}$ is indeed ≥ 0.971 and $\leq 1/2$ in those two cases. This confirms the calculation from [1], using our logic. (At least for the values of s, n we computed.)

6 Conclusion

We introduced the observe statement in the quantum setting for infinite-dimensional cases, supported by operational, denotational and weakest precondition semantics. We defined conditional weakest preconditions, proved their equivalence to the operational semantics and applied them to an example using Bayesian inference. Future work includes the interpretation of predicates and exploration of alternatives to observe statements such as rejection sampling

	$s=2$	$s=3$	$s=4$	$s=7$	$s=8$	...
$n=2$	0.5	0.3838	0.3286			
$n=3$	<u>0.9714</u>	<u>0.9991</u>	0.5	0.4247	0.4123	...
$n=4$	<u>0.9991</u>	<u>0.9933</u>	<u>0.9714</u>	<u>0.9889</u>	0.5	...
$n=5$	<u>0.9889</u>	<u>0.9828</u>	<u>0.9991</u>	<u>0.9977</u>	<u>0.9714</u>	...
...	...	...	...	...	...	

Figure 4 Maximum of $\Pr_{nsk} = \hat{tr}(qcwp[\![S_k]\!](P, \mathbf{I}^{\otimes n+2}) \odot \rho)$ for $k \in [-n, n]$. The cases where $s < 2^{n-1}$ are underlined.

or hoisting in the probabilistic case. Additionally, the challenge of combining non-determinism with conditioning in probabilistic systems [22] may extend to quantum programs.

References

1. Scott Aaronson. Quantum computing, postselection, and probabilistic polynomial-time. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 461, 09 2005. doi:10.1098/rspa.2005.1546.
2. Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.
3. Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
4. John B. Conway. *A Course in Operator Theory*. Graduate Studies in Mathematics. American Mathematical Society, 2000.
5. John B. Conway. *A Course in Functional Analysis*. Graduate Texts in Mathematics. Springer New York, 2007. doi:10.1007/978-1-4757-4383-8.
6. Yuxin Deng and Yuan Feng. Formal semantics of a classical-quantum language. *Theoretical Computer Science*, 913:73–93, 2022. doi:10.1016/j.tcs.2022.02.017.
7. Ellie D'Hondt and Prakash Panangaden. Quantum weakest preconditions. *Math. Struct. Comput. Sci.*, 16(3):429–451, 2006.
8. Olivia Di Matteo. On the need for effective tools for debugging quantum programs. In *Proceedings of the 5th ACM/IEEE International Workshop on Quantum Software Engineering*, Q-SE 2024, page 17–20, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3643667.3648226.
9. Edsger W. Dijkstra. Guarded commands, nondeterminacy and formal derivation of programs. *Commun. ACM*, 18(8):453–457, 1975. doi:10.1145/360933.360975.
10. Edsger W. Dijkstra. *A Discipline of Programming*. Prentice-Hall, 1976.
11. Paul A. M. Dirac. *The Principles of Quantum Mechanics*. Clarendon Press, Oxford, 1930.
12. Yuan Feng and Yingte Xu. Verification of nondeterministic quantum programs. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, ASPLOS 2023, page 789–805, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3582016.3582039.
13. Yuan Feng and Mingsheng Ying. Quantum Hoare logic with classical variables. *ACM Transactions on Quantum Computing*, 2(4), 2021. doi:10.1145/3456877.
14. Benjamin L. Kaminski. *Advanced Weakest Precondition Calculi for Probabilistic Programs*. PhD thesis, 02 2019. doi:10.18154/RWTH-2019-01829.
15. Dexter Kozen. A probabilistic PDL. *Journal of Computer and System Sciences*, 30(2):162–178, 1985. doi:10.1016/0022-0000(85)90012-1.
16. Marco Lewis, Sadegh Soudjani, and Paolo Zuliani. Formal verification of quantum programs: Theory, tools, and challenges. *ACM Transactions on Quantum Computing*, 2023. doi:10.1145/3624483.

17 Gushu Li, Li Zhou, Nengkun Yu, Yufei Ding, Mingsheng Ying, and Yuan Xie. Projection-based runtime assertions for testing and debugging quantum programs. *Proc. ACM Program. Lang.*, 4(OOPSLA), 2020. `doi:10.1145/3428218`.

18 Jérémie O. Lumbroso. Optimal discrete uniform generation from coin flips, and applications. *CoRR*, abs/1304.1916, 2013.

19 Annabelle McIver and Carroll Morgan. *Abstraction, Refinement and Proof for Probabilistic Systems.* Monographs in Computer Science. Springer, 2005. `doi:10.1007/b138392`.

20 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition.* Cambridge University Press, 2010. `doi:10.1017/CBO9780511976667`.

21 Aditya V. Nori, Chung-Kil Hur, Sriram K. Rajamani, and Selva Samuel. R2: an efficient mcmc sampler for probabilistic programs. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, page 2476–2482. AAAI Press, 2014.

22 Federico Olmedo, Friedrich Gretz, Nils Jansen, Benjamin L. Kaminski, Joost-Pieter Katoen, and Annabelle Mciver. Conditioning in probabilistic programming. *ACM Trans. Program. Lang. Syst.*, 40(1), 2018. `doi:10.1145/3156018`.

23 Masamichi Takesaki. *Theory of Operator Algebras I.* Number Bd. 1 in Encyclopaedia of Mathematical Sciences. Springer New York, 1979. `doi:10.1007/978-1-4612-6188-9`.

24 Dominique Unruh. The tensor product on Hilbert spaces. *Arch. Formal Proofs*, 2024. URL: `https://www.isa-afp.org/entries/Hilbert_Space_Tensor_Product.html`.

25 Dominique Unruh. Quantum references, 2024. `arXiv:2105.10914v3`.

26 Peng Yan, Hanru Jiang, and Nengkun Yu. On incorrectness logic for quantum programs. *Proc. ACM Program. Lang.*, 6(OOPSLA1), 2022. `doi:10.1145/3527316`.

27 Mingsheng Ying. Floyd-Hoare logic for quantum programs. *ACM Trans. Program. Lang. Syst.*, 2012. `doi:10.1145/2049706.2049708`.

28 Mingsheng Ying, Runyao Duan, Yuan Feng, and Zhengfeng Ji. Predicate transformer semantics of quantum programs. *Semantic Techniques in Quantum Computation*, 2010. `doi:10.1017/CBO9781139193313.009`.

29 Li Zhou, Nengkun Yu, and Mingsheng Ying. An applied quantum Hoare logic. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2019, page 1149–1162, New York, NY, USA, 2019. Association for Computing Machinery. `doi:10.1145/3314221.3314584`.

A Appendix

A.1 Proofs Concerning the Semantics

Before proving the properties of the denotational semantics, we need to show an auxiliary lemma:

▶ **Lemma 28.**
1. *If for every $n \in \mathbb{N}$, $f_n : \mathcal{D}^-(\mathcal{H}) \to \mathcal{D}^-(\mathcal{H})$ is bounded linear and pointwise increasing, that means for every fixed $\rho \in \mathcal{D}^-(\mathcal{H})$ $m > n$ implies $f_n(\rho) \sqsubseteq f_m(\rho)$, then $f_\infty(\rho) := \bigvee_{n=0}^{\infty} f_n(\rho)$ exists and f_∞ is linear.*
2. *If for every $n \in \mathbb{N}$, $e_n : \mathcal{DR} \to \mathbb{R}_{\geq 0}$ is bounded linear and pointwise increasing, then $e_\infty(\rho, p) := \bigvee_{n=0}^{\infty} e_n(\rho, p)$ exists for every $(\rho, p) \in \mathcal{DR}$ and e_∞ is linear.*

Proof. TOPROVE 9 ◀

We now prove Proposition 5:

Proof. TOPROVE 10 ◀

In the following proof we show the equivalence of the operational and denotational semantics, i.e., the proof of Lemma 7.

Proof. TOPROVE 11 ◀

Here we provide the proof of Lemma 6:

Proof. TOPROVE 12 ◀

A.2 Proofs Concerning Weakest Preconditions

Now we show the existence of the weakest precondition, Lemma 11:

Proof. TOPROVE 13 ◀

Now we also prove the existence of weakest liberal precondition, Lemma 12:

Proof. TOPROVE 14 ◀

Proof of Proposition 13:

Proof. TOPROVE 15 ◀

Proof of Proposition 14:

Proof. TOPROVE 16 ◀

Proof of equivalence between weakest preconditions and explicit representation, Proposition 15:

Proof. TOPROVE 17 ◀

Quite similar to the total correctness case, we also prove that the explicit representation of weakest liberal preconditions given in Proposition 16 is correct:

Proof. TOPROVE 18 ◀

Here we give the explicit representation of *qcwp*:

▶ **Lemma 29.** *An explicit representation of the quantum conditional weakest precondition transformer* $qcwp[\![P]\!] : \mathcal{P}(\mathcal{H})^2 \to \mathcal{P}(\mathcal{H})^2$ *is given by:*

- $qcwp[\![\mathbf{skip}]\!](P, Q) = (P, Q)$
- $qcwp[\![q := 0]\!](P, Q) = \begin{cases} \Big(|0\rangle_q \langle 0| \, P \, |0\rangle_q \langle 0| + |1\rangle_q \langle 0| \, P \, |0\rangle_q \langle 1| \,, \\ |0\rangle_q \langle 0| \, Q \, |0\rangle_q \langle 0| + |1\rangle_q \langle 0| \, Q \, |0\rangle_q \langle 1| \Big) & , \text{ if } type(q) = Bool \\ \Big(\sum_{n\in\mathbb{Z}} |n\rangle_q \langle 0| \, P \, |0\rangle_q \langle n| \,, \\ \sum_{n\in\mathbb{Z}} |n\rangle_q \langle 0| \, Q \, |0\rangle_q \langle n| \Big) & , \text{ if } type(q) = Int \end{cases}$
- $qcwp[\![\bar{q} := U\bar{q}]\!](P, Q) = U^\dagger \cdot (P, Q) \cdot U$
- $qcwp[\![\mathbf{observe}\ (\bar{q}, O)]\!](P, Q) = O^\dagger \cdot (P, Q) \cdot O$
- $qcwp[\![S_1; S_2]\!](P, Q) = qcwp[\![S_1]\!](qcwp[\![S_2]\!](P, Q))$
- $qcwp[\![\mathbf{measure}\ M[\bar{q}] : \bar{S}]\!](P, Q) = \sum_m M_m^\dagger \cdot (qcwp[\![S_m]\!](P, Q)) \cdot M_m$
- $qcwp[\![\mathbf{while}\ M[\bar{q}] = 1\ \mathbf{do}\ S']\!](P, Q) = \bigvee_{n=0}^{\infty} P_n$ *with*

$$P_0 = (\mathbf{0}, \boldsymbol{I})$$
$$P_{n+1} = [M_0^\dagger \cdot (P, Q) \cdot M_0] + [M_1^\dagger \cdot qcwp[\![S']\!](P_n) \cdot M_1]$$

and $\bigvee_{n=0}^{\infty}$ *denoting the least upper bound according to* $\trianglelefteq$.

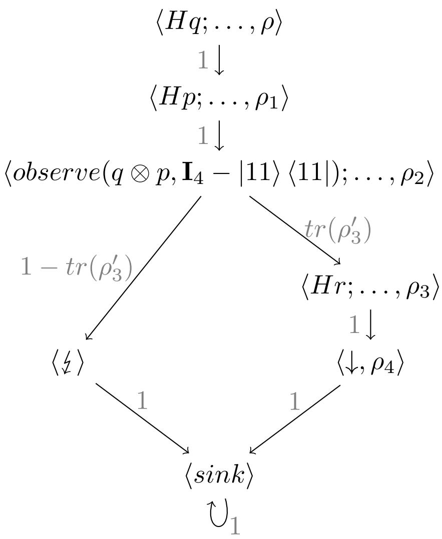

Figure 5 Operational semantics of the Quantum Fast-Dice-Roller with

- $\rho_1 = (H \otimes \mathbf{I}_2 \otimes \mathbf{I}_2)\rho(H \otimes \mathbf{I}_2 \otimes \mathbf{I}_2)^\dagger$
- $\rho_2 = (H \otimes H \otimes \mathbf{I}_2)\rho(H \otimes H \otimes \mathbf{I}_2)^\dagger$
- $\rho_3 = \frac{1}{tr(\rho_3')}\rho_3'$ with $\rho_3' = ((\mathbf{I}_4 - |11\rangle\langle 11|) \otimes \mathbf{I}_2)(H \otimes H \otimes \mathbf{I}_2)\rho(H \otimes H \otimes \mathbf{I}_2)^\dagger((\mathbf{I}_4 - |11\rangle\langle 11|) \otimes \mathbf{I}_2)^\dagger$
- $\rho_4 = \frac{1}{tr(\rho_3')}((\mathbf{I}_4 - |11\rangle\langle 11|) \otimes \mathbf{I}_2)(H \otimes H \otimes H)\rho(H \otimes H \otimes H)^\dagger((\mathbf{I}_4 - |11\rangle\langle 11|) \otimes \mathbf{I}_2)^\dagger$

Proof of Proposition 20:

Proof. TOPROVE 19 ◀

Proof of Proposition 23:

Proof. TOPROVE 20 ◀

Proof of Lemma 26:

Proof. TOPROVE 21 ◀

A.3 Details of the Example

The operational semantics of the Quantum Fast-Dice-Roller can be found in Figure 5.

To compute $qwp[\![S]\!](P)$ and $qwlp[\![S]\!](\mathbf{I})$, we use the rules from Proposition 15 and 16 and obtain

$$
\begin{aligned}
qwp[\![S]\!](P) &= (\mathbf{I}_4 \otimes H)^\dagger((\mathbf{I}_4 - |11\rangle\langle 11|) \otimes \mathbf{I})^\dagger(\mathbf{I} \otimes H \otimes \mathbf{I})^\dagger(H \otimes \mathbf{I}_4)^\dagger P \\
&\quad (H \otimes \mathbf{I}_4)(\mathbf{I} \otimes H \otimes \mathbf{I})((\mathbf{I}_4 - |11\rangle\langle 11|) \otimes \mathbf{I})(\mathbf{I}_4 \otimes H) \\
&= \begin{pmatrix}
3/4 & 0 & 1/4 & 0 & 1/4 & 0 & -1/4 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/4 & 0 & 1/12 & 0 & 1/12 & 0 & -1/12 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/4 & 0 & 1/12 & 0 & 1/12 & 0 & -1/12 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1/4 & 0 & -1/12 & 0 & -1/12 & 0 & 1/12 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix} \\
qwlp[\![S]\!](\mathbf{I}) &= (\mathbf{I}_4 \otimes H)^\dagger((\mathbf{I}_4 - |11\rangle\langle 11|) \otimes \mathbf{I})^\dagger(\mathbf{I} \otimes H \otimes \mathbf{I})^\dagger(H \otimes \mathbf{I}_4)^\dagger \mathbf{I} \\
&\quad (H \otimes \mathbf{I}_4)(\mathbf{I} \otimes H \otimes \mathbf{I})((\mathbf{I}_4 - |11\rangle\langle 11|) \otimes \mathbf{I})(\mathbf{I}_4 \otimes H) \\
&= \begin{pmatrix}
3/4 & 0 & 1/4 & 0 & 1/4 & 0 & -1/4 & 0 \\
0 & 3/4 & 0 & 1/4 & 0 & 1/4 & 0 & -1/4 \\
1/4 & 0 & 3/4 & 0 & -1/4 & 0 & 1/4 & 0 \\
0 & 1/4 & 0 & 3/4 & 0 & -1/4 & 0 & 1/4 \\
1/4 & 0 & -1/4 & 0 & 3/4 & 0 & 1/4 & 0 \\
0 & 1/4 & 0 & -1/4 & 0 & 3/4 & 0 & 1/4 \\
-1/4 & 0 & 1/4 & 0 & 1/4 & 0 & 3/4 & 0 \\
0 & -1/4 & 0 & 1/4 & 0 & 1/4 & 0 & 3/4
\end{pmatrix}
\end{aligned}
$$