Quantum Weakest Preconditions

Ellie D'Hondt [*] Prakash Panangaden [†‡]

Abstract

We develop a notion of predicate transformer and, in particular, the weakest precondition, appropriate for quantum computation. We show that there is a Stone-type duality between the usual state-transformer semantics and the weakest precondition semantics. Rather than trying to reduce quantum computation to probabilistic programming we develop a notion that is directly taken from concepts used in quantum computation. The proof that weakest preconditions exist for completely positive maps follows from the Kraus representation theorem. As an example we give the semantics of Selinger's language in terms of our weakest preconditions.

1 Introduction

Quantum computation is a field of research that is rapidly acquiring a place as a significant topic in computer science. To be sure, there are still essential technological and conceptual problems to overcome in building functional quantum computers. Nevertheless there are fundamental new insights into quantum computability [Deu85, DJ92], quantum algorithms [Gro96, Gro97, Sho94, Sho97] and into the nature of quantum mechanics itself [Per95], particularly with the emergence of quantum information theory [NC00].

These developments inspire one to consider the problems of programming full-fledged general purpose quantum computers. Much of the theoretical research is aimed at using the new tools available - superposition, entanglement and linearity - for algorithmic efficiency. However, the fact that quantum algorithms are currently done at a very low level only - comparable to classical computing 60 years ago - is a much less explored aspect of the field. The present paper is

[*]Centrum Leo Apostel , Vrije Universiteit Brussel, Brussels, Belgium, eldhondt@vub.ac.be.
[†]Computing Laboratory, Oxford University, Oxford, U.K., prakash@cs.mcgill.ca
[‡]On sabbatical leave from: School of Computer Science, McGill University, Montreal, Canada.

situated in the nascent area of quantum programming methodology and the design and semantics of quantum programming languages. More precisely we consider the well-known paradigm of *weakest-preconditions* [Dij76] as the basis of a goal-directed programming methodology and as a semantics for programming languages. We formulate the quantum analogue of predicate transformers and, in particular, the weakest precondition.

Rather than reducing quantum computation to probabilistic computation and using well-known ideas from this setting [Koz81, Koz85] we develop the ideas directly for the quantum setting. It turns out that the same beautiful Stone-type duality between state-transformer (forwards) and predicate-transformer (backwards) that one has seen in the traditional [Smy83, Plo83] and the probabilistic settings [Koz85] appears in the quantum setting.

The influence of Dijkstra's work on weakest preconditions [Dij76] has been deep and pervasive and has even led to textbook level expositions of the subject [Gri81]. The main point is that it leads to a *goal-directed* program or algorithm development strategy. Hitherto quantum algorithms have been invented by brilliant new insights. As more and more algorithms accumulate and a stock of techniques start to accumulate there will be need for a systematic program development strategy. It is this that we hope will come out of the present work.

A second goal is to present a semantical paradigm for quantum computation. Quantum programming languages have started to appear. Perhaps the best known is one due to Selinger [Sel03] which is based on the slogan: "Quantum data and classical control." While this is not the final word on the subject, it is an excellent start with a clean semantics and a clear conceptual basis. We give a semantics for this language based on our notion of weakest precondition. It should be noted, however, that the definition of weakest preconditions and the basic existence results are *language independent*.

There is a remarkable duality [Smy83] between the forward operational semantics - expressed as state transformers - and the backwards predicate transformer semantics. This duality is part of a web of dualities known to mathematicians as Stone-type dualities [Joh82], the prototype of which is the duality between boolean algebras and certain topological spaces called Stone spaces. In categorical terms such a duality is captured by an adjoint equivalence mediated by a pairing. In this case the pairing is the satisfaction relation between states and predicates. Kozen - following suggestions of Plotkin - found such a duality in the context of probabilistic programs [Koz85]. In the present paper we show that such a duality exists in the quantum setting as well.

In the present paper we make two contributions:

- we develop the appropriate quantum analogue of weakest-precondition semantics and develop the duality theory, and
- we write the detailed weakest-precondition semantics for Selinger's language including iteration and recursion.

We are fortunate in that we can prove existence of weakest preconditions for completely positive maps in a very general way using a powerful mathematical result called the Kraus representation theorem; see for example, chapter 8 section 2 in the recent book by Nielsen and Chuang [NC00].

The structure of this paper is as follows. In Sec. 2 the general setup is laid out, in particular quantum state transformers and quantum predicates. Next, in Sec. 3 we define quantum weakest preconditions and healthy predicate transformers. We also prove their existence for every completely positive map and every observable. Sec. 4 summarizes the basic structure of Selinger's language, and in Secs. 5 and 6 we develop its weakest precondition semantics. We develop an application to Grover's algorithm in Sec. 7. We conclude with Sec.8.

2 General setup

In the operational semantics for a traditional imperative language one has a notion of *state*, such that the commands in the language are interpreted as state transformers. If the language is deterministic the state transformation is given by a function, and composition of commands corresponds to functional composition. The flow is forwards through the program. This type of semantics is intended to give meaning to programs that have already been written. It is useful for guiding implementations of programming languages and is, perhaps, less useful for program development.

By contrast, in a predicate transformer semantics the meaning is constructed by flowing backwards through the program, starting from the final intended result and proceeding backwards to determine what must be true of the original program. This type of semantics is useful for *goal-directed* programming. Of course the two types of semantics are intimately related, as they should be! In a sense to be made precise later they are dual to each other.

In the world of probabilistic programs one sees the same duality in action. Here the role of state is played by *distributions*. There are, of course, states as before but, though in a particular execution there is only one state at every stage, in order to describe all the possible outcomes (and their relative probabilities) one keeps track of the probability distribution over the state space and how it changes

during program execution. What plays the role of "predicates"? Kozen has argued [Koz85] that predicates are measurable functions – or random variables, to use the probability terminology. A special case of random variables are characteristic functions, which are more easily recognizable as the analogues of predicates (in fact they *are* predicates). Rather than "truth" one has "expectation" value and "truth values" now lie in $[0, 1]$ rather than in $\{0, 1\}$. The pairing between measurable functions and distributions is given by the integral just as the pairing between predicates and states is given by satisfaction.

For the quantum world we again need a notion of state - or distribution over possible states - a notion of predicate and a pairing. Our choices are very much guided by the probabilistic case, but we are absolutely not claiming that quantum computation can be seen as a subcase of classical probabilistic computation. There are crucial differences that we discuss briefly below.

Typically a quantum system is described by a Hilbert space, physical observables are described by hermitian operators on this space and transformations of the system are effected by unitary operators [Per95]. However, we need to describe not only so-called "pure" states but also mixed states. These arise as soon as one has to deal with "partial information" in a quantum setting. For example, a system may be prepared as a statistical mixture, it may be mixed as a result of interactions with a noisy environment (decoherence) or by certain parts of the system being unobservable. For all these reasons we need to work with distributions over the states in a Hilbert space.

We take density matrices to be the analogue of distributions. These describe probabilistic mixtures of quantum states. A good expository discussion appears in Nielsen and Chuang's book [NC00]. For predicates we take (a certain restricted class of) hermitian operators. These will then be "observables" of the system. In discussions of quantum logic and foundations of quantum theory the actual numerical values of the observables are not important, what matters is what subspace of the Hilbert space is being observed or tested. We thus take as our observables not all conceivable hermitian operators but (essentially) ones that describe subspaces of the Hilbert space. This is very much in the spirit of a "predicate as a description of a subspace of the state space." The third notion we need is that of a pairing. Given a density matrix ρ and an observable M the expectation value of M in ρ is given by $Tr(M\rho)$; where the Tr stand for the usual trace from linear algebra. Throughout this paper we will work with finite-dimensional Hilbert spaces and one can think of M and ρ as matrices. The situation is summarized in the following table:

Deterministic	Probabilistic	Quantum
s	μ	ρ
p	f	M
$s \models p$	$\int f d\mu$	$\mathrm{Tr}(M\rho)$

2.1 Quantum state transformers

In a quantum setting, forward semantics is described by a *quantum-state* transformer. The properties of such state transformers are now well understood. A quantum state is represented by a density matrix ρ on a Hilbert space $\mathcal{H}$, which is a *positive operator*, i.e. $\forall x \in \mathcal{H}.\langle x,\ \rho x\rangle \geq 0$, such that $\mathrm{Tr}(()\rho) \leq 1$[1]. A physical transformation must take a density matrix to a density matrix. Let us not worry about the trace for the moment and focus on the positivity requirement. A linear map that takes a positive operator to a positive operator is called a *positive map*. It seems reasonable to require that physical effects correspond to positive maps. However, a remarkable thing happens. It is possible for a positive map to be tensored with another positive map - even an identity map - and for the result to fail to be positive.

Physically this is a disaster. If $\mathcal{E}_1$ is a physical transformation of a system in state ρ and we *formally* regard this system as part of another far away system which we do not touch (i.e. to which we apply the identity) then suddenly we have an unphysical transformation. A simple example is provided by the transpose operation; it is a positive map but its tensor with an identity will not be. We impose the stronger requirement that a physical effect corresponds to a *completely positive map* which are defined as follows.

Definition 2.1 *A* completely positive map $\mathcal{E}$ *is a linear map such that* $\mathcal{E}(\rho)$ *is a density matrix for all density matrices* ρ*, and such that*

$$\forall \rho \in \mathcal{DM}(\mathcal{H}).(I_{\mathcal{H}_1} \otimes \mathcal{E})(\rho) \in \mathcal{DM}(\mathcal{H}) \tag{1}$$

where $I_{\mathcal{H}_1}$ *is the identity on* $\mathcal{H}_1$*,* $\mathcal{E} \in \mathcal{CP}(\mathcal{H}_2)$ *and* $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$*.*

$\mathcal{CP}(\mathcal{H})$ denotes the set of all completely positive maps operating on a Hilbert space $\mathcal{H}$. We frequently rely on the Kraus representation theorem for completely positive maps, which goes as follows.

[1] In most books they insist that $\mathrm{Tr}(()\rho) = 1$ but we will not assume that everything is always normalized so we allow the trace to be less than 1.

Theorem 2.2 *The map $\mathcal{E}$ is a completely positive map, such that $\mathrm{Tr}(\mathcal{E}(\rho)) \leq 1$ for all states ρ, if and only if*

$$\forall \rho \in \mathcal{DM}(\mathcal{H}).\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger \tag{2}$$

for some set of operators $\{E_i\}$ which map the input Hilbert space to the output Hilbert space, with $\sum_i E_i^\dagger E_i \leq I$.

The condition on the E_i assures us that trace of the density matrix will never increase. Eq.(2) is also known as the *operator-sum representation*. The proof to this theorem can be found, for example, in [NC00]. Note there is nothing in the theorem that says that the E_i are unique.

2.2 Quantum Predicates

In this section, we formally define quantum predicates and the associated order structure required for the development of our theory. Concretely, we need an ordering on predicates so as to define *weakest* preconditions, and this order should be *continuous* in order to deal with programming language aspects such as recursion and iteration.

As argumented above, quantum predicates are given by hermitian operators. More precisely, we have the following definition.

Definition 2.3 *A* predicate *is a Hermitian operator with trace bounded in absolute value by some $B \in \mathbf{R}$, where we take $B = 1$ without loss of generality.*

The reason for taking predicates to be *bounded* hermitian operators is clarified below. We denote the set of all predicates on a Hilbert space $\mathcal{H}$ by $\mathcal{P}(\mathcal{H})$.

Definition 2.4 *For matrices M and N in $\mathbf{C}^{n\times n}$ we define $M \sqsubseteq N$ if $N - M$ is positive.*

This order is known in the literature as the *Löwner partial order* [Löw34]. Note that this definition can be rephrased in the following way, where $\mathcal{DM}(\mathcal{H})$ denotes the set of all density matrices.

Proposition 2.5 *$M \sqsubseteq N$ if and only if $\forall \rho \in \mathcal{DM}(\mathcal{H}).\mathrm{Tr}(M.\rho) \leq \mathrm{Tr}(N.\rho)$*

Proof Indeed, $N-M$ positive means that for all $x \in \mathcal{H}$ we have $\langle x|(N-M).x\rangle \geq 0$, or, equivalently, $\mathrm{Tr}((N-M).|x\rangle\langle x|) \geq 0$. By linearity of the trace operation and the fact that the spectral theorem holds for all $\rho \in \mathcal{DM}(\mathcal{H})$ we obtain Eq.2.5. ∎

Put otherwise, $M \sqsubseteq N$ if and only if the expectation value of N exceeds that of M.

With the above definitions, we have the following result.

Proposition 2.6 *The poset $(\mathcal{P}(\mathcal{H}), \sqsubseteq)$ is a directed complete partial order (DCPO), i.e. it contains least upper bounds of increasing sequences.*

Proof Take an increasing sequence of predicates $M_1 \sqsubseteq M_2 \sqsubseteq \ldots \sqsubseteq M_i \sqsubseteq \ldots$. Subtracting M_1 everywhere, we obtain the sequence $0 \sqsubseteq M_2 - M_1 \sqsubseteq \ldots \sqsubseteq M_i - M_1 \sqsubseteq \ldots$. This is a sequence of positive operators with trace bounded by 1, or in other words, density matrices. Since $(\mathcal{DM}(\mathcal{H}), \sqsubseteq)$ is a DCPO [Sel03], this sequence has a least upper bound M. Therefore the original predicate sequence has a least upper bound $M + M_1 \in \mathcal{P}(\mathcal{H})$. It follows that $(\mathcal{P}(\mathcal{H}), \sqsubseteq)$ is a DCPO. ∎

In discussions of quantum logic and foundations of quantum theory it is common practice to consider only the subspace of the Hilbert space that is being observed, rather than the actual numerical values of the observables. In this spirit, we define predicates as essentially those hermitian operators that describe subspaces of the Hilbert space. However, taking predicates to be *bounded* hermitian operators also leads to the above Prop.2.6, which guarantees the existence of fixpoints and thus allows for the formal treatment of iteration and recursion (see Sec.6).

3 Quantum weakest preconditions and duality

3.1 Definition

In a quantum setting, the role of the satisfaction relation is taken over by the *expectation value* of an observable M, just as for probabilistic computation. The quantum expectation value of a predicate M is given by the trace expression $\mathrm{Tr}(M\rho)$. So preconditions for a quantum program $\mathcal{Q}$, describing a hitherto unspecified quantum evolution, are defined as follows.

Definition 3.1 *The predicate M is said to be a* precondition *for the predicate N, with respect to a quantum program $\mathcal{Q}$, denoted $M\{\mathcal{Q}\}N$, if*

$$\forall \rho \in \mathcal{DM}(\mathcal{H}).\mathrm{Tr}(M\rho) \leq \mathrm{Tr}(N\mathcal{Q}(\rho)) \tag{3}$$

The exact form of the quantum program $\mathcal{Q}$ is being kept vague deliberately , as we want to state these definitions without commiting to any particular framework.

From this we define weakest preconditions in the usual way.

Definition 3.2 *The* weakest precondition *for a predicate M with respect to a quantum program $\mathcal{Q}$, denoted $\mathrm{wp}(\mathcal{Q})(M)$, is such that for all preconditions $L\{\mathcal{Q}\}N$ implies $L \sqsubseteq \mathrm{wp}(\mathcal{Q})(M)$.*

Note that *weakest* in this context is equal to *largest*; indeed, a larger predicate holds for more initial states ρ, and thus corresponds to a weaker constraint. To any given quantum program $\mathcal{Q}$ corresponds a weakest precondition predicate transformer $\mathrm{wp}(\mathcal{Q}) : \mathcal{P}(\mathcal{H}) \to \mathcal{P}(\mathcal{H})$.

3.2 Healthiness conditions

In analogy with [Dij76], we want to formulate *healthiness conditions* for quantum predicate transformers. These are important because they characterise exactly those programs that can be given an alternative weakest precondition semantics, which is then dual to the forwards state transformer semantics embodied by the program. Moreover, healthiness conditions allow one to prove general laws for reasoning about programs. For standard (i.e. classical nonprobabilistic) programs the healthiness conditions are conjunctivity, feasibility, monotonicity, disjunctivity and continuity. We propose the following in the quantum case, where X is a predicate transformer, $\alpha, \beta \in \mathbf{C}$ and $M, N \in \mathcal{P}(\mathcal{H})$. As we shall see in the following section these conditions all hold in the framework where quantum programs correspond to completely positive maps.

- **linearity:**
$$X(\alpha M + \beta N) = \alpha X(M) + \beta X(N) \tag{4}$$
Linearity is the generalisation of conjunctivity, and is certainly a requirement in the inherently linear context of quantum mechanics.

- **monotonicity:**
$$M \sqsubseteq N \Rightarrow X(M) \sqsubseteq X(N) \tag{5}$$

- **continuity:**
$$M_1 \sqsubseteq M_2 \sqsubseteq ... \sqsubseteq M_i \sqsubseteq ... \Rightarrow X(\sqcup_i M_i) = \sqcup_i X(M_i) \tag{6}$$

These last two conditions together imply order continuity.

The last condition pertains to composite systems. Suppose $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$, with $\otimes$ the usual tensor product on vector spaces, such that $I_{\mathcal{H}_1}$ is the identity on $\mathcal{H}_1$ and $X \in \mathcal{P}(\mathcal{H}_2)$. The fifth healthiness condition is then

- **monoidal:**

$$(I_{\mathcal{H}_1} \otimes X)(M) \in \mathcal{P}(\mathcal{H}) \tag{7}$$

The requirement in Cond.(7), i.e. that predicate transformers be *monoidal* on $\mathcal{P}(\mathcal{H})$, is a very natural one. Indeed, if X is a predicate transformer, which acts only on part of a composite Hilbert space $\mathcal{H}$, then composing it with the identity predicate transformer working on the rest of the Hilbert space should still result in a valid predicate transformer.

So we have the following definition.

Definition 3.3 *A* healthy *predicate transformer $\alpha : \mathcal{P}(\mathcal{H}) \to \mathcal{P}(\mathcal{H})$ is a predicate transformer satisfying the healthiness conditions (4)–(7). We denote the space of all healthy predicate transformers on a Hilbert space $\mathcal{H}$as $\mathcal{PT}(\mathcal{H})$.*

We equip $\mathcal{PT}(\mathcal{H})$ with an order structure by extending the Löwner order on predicates in the obvious way, thus obtaining the following result.

Proposition 3.4 *The poset $(\mathcal{PT}(\mathcal{H}), \sqsubseteq)$ is a DCPO.*

Proof Take an arbitrary increasing sequence of predicate transformers $P_1 \sqsubseteq P_2 \sqsubseteq \ldots \sqsubseteq P_i \sqsubseteq \ldots$. From Prop.(2.6) it follows that P, defined for all $M \in \mathcal{P}(\mathcal{H})$ as $P(M) = \sqcup_i(P_i(M))$, is the least upper bound to this series, so $P = \sqcup_i P_i$. P is a linear map from predicates to predicates; it remains to be checked that it is continuous. Take an increasing series of predicates M_j with least upper bound $\sqcup_j M_j$. We then have

$$\begin{aligned} P(\sqcup_j Mj) &= \sqcup_i P_i(\sqcup_j M_j) \\ &= \sqcup_i \sqcup_j P_i(M_j) \qquad \text{(by Cond.5)} \\ &= \sqcup_j \sqcup_i P_i(M_j) \\ &= \sqcup_j P(M_j) \end{aligned} \tag{8}$$

So $P = \sqcup_i P_i \in \mathcal{PT}(\mathcal{H})$. It follows that $(\mathcal{PT}(\mathcal{H}), \sqsubseteq)$ is a DCPO. ∎

Note that the DCPO structure as defined on predicates $\mathcal{P}(\mathcal{H})$ and associated predicate transformers $\mathcal{PT}(\mathcal{H})$ is completely analogous to that for density matrices $\mathcal{DM}(\mathcal{H})$ and associated completely positive maps $\mathcal{CP}(\mathcal{H})$, as defined in [Sel03].

Furthermore, for healthy predicate transformers, we have the following generalization of Kraus's theorem.

Theorem 3.5 *The operator α is a healthy predicate transformer if and only if one has that*

$$\forall M \in \mathcal{P}(\mathcal{H}).\alpha(M) = \sum_u A_u^\dagger M A_u \tag{9}$$

for some set of linear operators $\{A_u\}$ such that $\sum_u A_u^\dagger A_u \leq I$.

Proof The proof is completely analogous to that of Kraus's theorem for completely positive maps, which crucially uses the spectral theorem on the one hand and complete positivity on the other hand. Since the spectral theorem holds for predicates also, the only thing we really require is something analogous to complete positivity, i.e. Cond.(6). ∎

3.3 Predicate transformers for completely positive maps

Let us now consider the following framework: every quantum program is described by a completely positive map $\mathcal{E} \in \mathcal{CP}(\mathcal{H})$. In this section we prove an existence theorem of weakest preconditions for completely positive maps, and show that they satisfy the healthiness conditions given in Sec.3.2, i.e. that they are healthy predicate transformers.

Proposition 3.6 *$\forall \mathcal{E} \in \mathcal{CP}(\mathcal{H})$ and $N \in \mathcal{P}(\mathcal{H})$, $\mathrm{wp}(\mathcal{E})(N)$ exists and is unique. Furthermore, we have that*

$$\forall \rho.\mathrm{Tr}(\mathrm{wp}(\mathcal{E})(N)\rho) = \mathrm{Tr}(N\mathcal{E}(\rho)) \tag{10}$$

Proof To prove existence, take a random predicate N and effect $\mathcal{E}$. Due to the Kraus representation theorem[2], one has for every effect $\mathcal{E}$ that

$$\mathcal{E}(\rho) = \sum_m E_m \rho E_m^\dagger \tag{11}$$

[2] See appendix.

with $\sum_m E_m E_m^\dagger \leq I$. Using this, together with the fact that the trace is linear and invariant to cyclic permutations, we obtain for a predicate N that

$$\mathrm{Tr}(N\mathcal{E}(\rho)) = \mathrm{Tr}((\textstyle\sum_m E_m^\dagger N E_m)\rho) \tag{12}$$

If we then take

$$M = \sum_m E_m^\dagger N E_m \tag{13}$$

in Eq.(12), we obtain

$$\forall\rho.\mathrm{Tr}(M\rho) = \mathrm{Tr}(N\mathcal{E}(\rho)) \tag{14}$$

So M is a precondition for N with respect to $\mathcal{E}$. Now take any other precondition M' for N with respect to $\mathcal{E}$. In other words

$$\forall\rho.\mathrm{Tr}(M'\rho) \leq \mathrm{Tr}(N\mathcal{E}(\rho)) \tag{15}$$

but due to Eq.(14), this implies that $M' \sqsubseteq M$. So M is the weakest precondition for N with respect to $\mathcal{E}$, denoted $\mathrm{wp}(\mathcal{E})(N)$.

To prove uniqueness, suppose the predicate P is also a weakest precondition for N with respect to $\mathcal{E}$. Then we have $M \sqsubseteq P$, but also, since M is a weakest precondition, $P \sqsubseteq M$. But then, since $\sqsubseteq$ is an order, we have $M = P$. ∎

We now prove weakest preconditions for completely positive maps satisfy the healthiness conditions (4) – (7). In order to do this we need the following lemma.

Lemma 3.7 Tr *is continuous on* $\mathcal{P}(\mathcal{H})$.

Proof Monotonicity follows from Eq.(2.5) with $\rho = I$. Now take an increasing sequence of predicates $M_1 \sqsubseteq M_2 \sqsubseteq \ldots \sqsubseteq \sqcup_i M_i$. Taking the trace everywhere we obtain an increasing series of reals, which has a least upper bound $\sqcup_i \mathrm{Tr}(M_i)$. On the other hand, the positive series $M_i - M_1$, and thus its square root, converge in the trace norm, and this limit is equal to the least upper bound [Sel03]. In particular, this means that the sequence $\mathrm{Tr}(M_i)$ converges to $\mathrm{Tr}(\sqcup_i M_i)$. It follows that $\mathrm{Tr}(\sqcup_i M_i) = \sqcup_i \mathrm{Tr}(M_i)$, and thus that the trace operation is continuous. ∎

Proposition 3.8 *For all $\mathcal{E} \in \mathcal{CP}(\mathcal{H})$, $\mathrm{wp}(\mathcal{E}) \in \mathcal{PT}(\mathcal{H})$, i.e. it satisfies the healthiness conditions (4) – (7).*

Proof Throughout the proof we take $\alpha, \beta \in \mathbf{C}$ and $\mathcal{E}, \mathcal{F} \in \mathcal{CP}(\mathcal{H})$, $\rho \in \mathcal{DM}(\mathcal{H})$ and $M_i \in \mathcal{P}(\mathcal{H})$.

- **linearity**
  This follows from the linearity of the trace operation. Indeed, we have

$$\mathrm{Tr}(M(\alpha\mathcal{E} + \beta\mathcal{F})(\rho)) = \mathrm{Tr}(\mathrm{wp}(\alpha\mathcal{E} + \beta\mathcal{F})(M)\rho) \tag{16}$$

  by Eq.(10), while on the other hand

$$\begin{aligned}\mathrm{Tr}(M(\alpha\mathcal{E} + \beta\mathcal{F})(\rho)) &= \alpha\mathrm{Tr}(M\mathcal{E}(\rho)) + \beta\mathrm{Tr}(M\mathcal{F}(\rho)) \\ &= \alpha\mathrm{Tr}(\mathrm{wp}(\mathcal{E})(M)\rho) + \beta\mathrm{Tr}(\mathrm{wp}(\mathcal{F})(M)\rho) \\ &= \mathrm{Tr}((\alpha\mathrm{wp}(\mathcal{E})(M) + \beta\mathrm{wp}(\mathcal{F})(M))\rho) \end{aligned} \tag{17}$$

  Comparing Eqs.(16) and (17) we obtain the desired result.

- **monotonicity**
  Take $M_1 \sqsubseteq M_2$. We then have for all $\rho \in \mathcal{DM}(\mathcal{H})$ and $\mathcal{E} \in \mathcal{CP}(\mathcal{H})$ that

$$\begin{aligned} &\mathrm{Tr}(M_1\mathcal{E}(\rho)) \leq \mathrm{Tr}(M_2\mathcal{E}(\rho)) && \text{(by Prop.2.5)} \\ \Rightarrow &\mathrm{Tr}(\mathrm{wp}(\mathcal{E})(M_1)\rho) \leq \mathrm{Tr}(\mathrm{wp}(\mathcal{E})(M_2)\rho) && \text{(by Prop.3.6)} \\ \Rightarrow &\mathrm{wp}(\mathcal{E})(M_1) \sqsubseteq \mathrm{wp}(\mathcal{E})(M_2) && \text{(by Prop.2.5)} \end{aligned} \tag{18}$$

  So $\mathrm{wp}(\mathcal{E})$ is monotone.

- **continuity**
  Suppose we have a least upper bound $\sqcup_i M_i$. Then for all $\rho \in \mathcal{DM}(\mathcal{H})$

$$\begin{aligned} \mathrm{Tr}(\mathrm{wp}(\mathcal{E})(\sqcup_i M_i)\rho) &= \mathrm{Tr}((\sqcup_i M_i)\mathcal{E}(\rho)) && \text{(by Prop.3.6)} \\ &= \sqcup_i \mathrm{Tr}(M_i\mathcal{E}(\rho)) && \text{(by Prop.3.7)} \\ &= \sqcup_i \mathrm{Tr}(\mathrm{wp}(\mathcal{E})(M_i)\rho) && \text{(by Prop.3.6)} \\ &= \mathrm{Tr}(\sqcup_i \mathrm{wp}(\mathcal{E})(M_i)\rho) && \text{(by Prop.3.7)} \end{aligned} \tag{19}$$

Therefore, by Prop.2.5, we have that $\mathrm{wp}(\mathcal{E})(\sqcup_i M_i) = \sqcup_i \mathrm{wp}(\mathcal{E})(M_i)$.
It follows that $\mathrm{wp}(\mathcal{E})$ is continuous on $\mathcal{P}(\mathcal{H})$.

- **monoidal**
  Suppose $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$. Taking $\mathcal{E} \in \mathcal{CP}(\mathcal{H}_2)$ as in Eq.(11) we have that the weakest precondition for $M_2 \in \mathcal{P}(\mathcal{H}_2)$ with respect to $\mathcal{E}$ is given by $\mathrm{wp}(\mathcal{E})(M_2) = \sum_m E_m^\dagger M_2 E_m$. Now take $M = M_1 \otimes M_2 \in \mathcal{P}(\mathcal{H})$. Then we have

$$\begin{aligned}(I_{\mathcal{H}_1} \otimes \mathrm{wp}(\mathcal{E}))(M_1 \otimes M_2) &= M_1 \otimes \sum_m E_m^\dagger M_2 E_m \\ &= \sum_m (I_{\mathcal{H}_1} \otimes E_m^\dagger)(M_1 \otimes M_2)((I_{\mathcal{H}_1} \otimes E_m)\end{aligned} \tag{20}$$

  and this is again a predicate due to Theorem 3.5. By linearity the same holds for general $M \in \mathcal{P}(\mathcal{H})$, and as such $\mathrm{wp}(\mathcal{E})$ is monoidal.

■

As an aside, note that we also have the following continuity result.

Proposition 3.9 *Weakest precondition predicate transformers* $\mathrm{wp}(\centerdot)$ *are continuous on* $\mathcal{CP}(\mathcal{H})$.

Proof Take $\mathcal{E}_1 \sqsubseteq \mathcal{E}_2$. We then have for all $\rho \in \mathcal{DM}(\mathcal{H})$ and hermitian operators $M \in \mathcal{H}erm(\mathcal{H})$ that

$$\begin{aligned}&\mathcal{E}_1(\rho) \sqsubseteq \mathcal{E}_2(\rho) \\ \Rightarrow &\mathrm{Tr}(\mathcal{E}_1(\rho).M)) \leq \mathrm{Tr}(\mathcal{E}_2(\rho).M)) && \text{(By Prop.2.5)} \\ \Rightarrow &\mathrm{Tr}(M.\mathcal{E}_1(\rho))) \leq \mathrm{Tr}(M.\mathcal{E}_2(\rho))) \\ \Rightarrow &\mathrm{Tr}(\mathrm{wp}(\mathcal{E}_1)(M).\rho)) \leq \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_2)(M).\rho)) && \text{(By Prop.3.6)} \\ \Rightarrow &\mathrm{wp}(\mathcal{E}_1) \sqsubseteq \mathrm{wp}(\mathcal{E}_2) && \text{(By Prop.2.5)}\end{aligned} \tag{21}$$

So $\mathrm{wp}(\centerdot)$ is monotone.

Now suppose we have a least upper bound $\sqcup_i \mathcal{E}_i$. Then for all $\rho \in \mathcal{DM}(\mathcal{H})$, $M \in \mathcal{H}erm(\mathcal{H})$

$$\begin{aligned}
\mathrm{Tr}(\mathrm{wp}(\sqcup_i \mathcal{E}_i)(M)\rho) &= \mathrm{Tr}(M(\sqcup_i \mathcal{E}_i)(\rho)) && \text{(by Prop.3.6)} \\
&= \sqcup_i \mathrm{Tr}(M\mathcal{E}_i(\rho)) && \text{(by Prop.3.7)} \\
&= \sqcup_i \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_i)(M)\rho) && \text{(by Prop.3.6)} \\
&= \mathrm{Tr}(\sqcup_i \mathrm{wp}(\mathcal{E}_i)(M)\rho) && \text{(by Prop.3.7)} \qquad (22)
\end{aligned}$$

Therefore, by Prop.2.5, we have that $\mathrm{wp}((\sqcup_i \mathcal{E}_i)M) = \sqcup_i \mathrm{wp}(\mathcal{E}_i)(M)$.

Hence, $\mathrm{wp}(\boldsymbol{.})$ is continuous on $\mathcal{CP}(\mathcal{H})$. ∎

3.4 Duality

In this section, we investigate the duality between the forward axiomatic semantics of completely positive maps as state transformers, and the backwards axiomatic semantics of healthy predicate transformers. Concretely, we define an isomorphism between the set of all completely positive maps $\mathcal{CP}(\mathcal{H})$ and the set of all healthy predicate transformers $\mathcal{PT}(\mathcal{H})$. Associating a healthy predicate transformer with every effect $\mathcal{E}$ follows immediately from Prop.3.6. Indeed, we associate with every effect $\mathcal{E}$ its weakest precondition predicate transformer $\mathrm{wp}(\mathcal{E})$. To complete the duality, we need to associate an effect $\mathcal{A} \in \mathcal{CP}(\mathcal{H})$ with a predicate transformer $\alpha \in \mathcal{PT}(\mathcal{H})$. Using the operator-sum representation for predicate transformers as given in Eq.(9), we have that

$$\begin{aligned}
\mathrm{Tr}(\alpha(M)\rho) &= \mathrm{Tr}((\textstyle\sum_u A_u^\dagger M A_u)\rho) \\
&= \mathrm{Tr}(M.(\textstyle\sum_u A_u \rho A_u^\dagger) \qquad (23)
\end{aligned}$$

If we then take

$$\mathcal{A}(\rho) = \sum_u A_u \rho A_u^\dagger \qquad (24)$$

we obtain

$$\mathrm{Tr}(\alpha(M)\rho) = \mathrm{Tr}(M\mathcal{A}(\rho)) \qquad (25)$$

thus associating an state transformer with every healthy predicate transformer. Analogously to the above, one could say that this expression defines the "strongest poststate" $\mathcal{A}(\rho)$ for a state ρ, with respect to a predicate transformer $\alpha \in \mathcal{PT}(\mathcal{H})$.

To see this as a duality more clearly, suppose we define the notation $\rho \models_r M$ to mean that $\mathrm{Tr}(M\rho) \geq r$. Thus we think of this as a quantitative satisfaction relation with the real number r providing a "threshold" above which we deem that ρ satisfies M. Then we have

$$\frac{\mathcal{E}(\rho) \models_r M}{\rho \models_r \mathrm{wp}(\mathcal{E})M}.$$

The fact that we have an order isomorphism between the domain of predicate transformers and the domain of state transformers is clear. This can be expressed categorically of course and made to look just like the usual Stone-type dualities but we omit that in the present paper as it would involve setting up too much more machinery.

4 A summary of the quantum flowchart language

The quantum flowchart language. also known as QPL (for Quantum Programming Language), is a programming language for quantum computation in the spirit of quantum data with classical control [Sel03]. It is formally defined in a categorical context and has a denotational semantics. Concretely, the denotation of a program state is given by a tuple of density matrices. The tuple dimension stems from the present classical bits, i.e. each member of the tuple corresponds to a classical control path, whereas the density matrices correspond to registers of qubits. Program transformations are represented formally as tuples of completely positive maps (called *superoperators* in [Sel03]), which as such can act upon program states.

Syntactically, programs in QPL are represented either by flowcharts or by QPL terms. Several basic language components are present in the syntax, such as allocating or discarding bits or qubits, assignment, branching, merge, measurement and unitary transformation. One can then build more complex programs from these atomic flowchart components through context extension, vertical and horizontal composition, iteration and recursion. The categorical structure ensures that these compositions are well-defined.

Concretely, QPL is described within the category $\mathbf{Q}$, which has signatures (which define complex vector spaces) as its objects and effects as its morphisms. This category is equipped with a CPO-structure, a co-pairing map $\oplus$, a tensor product $\otimes$ and a categorical trace $\mathbf{tr}()$. All the basic flowchart components are morphisms of this category. Context extension, vertical and horizontal composition correspond to tensor product, composition and co-pairing of morphisms respectively, while iteration is interpreted via the monoidal trace. Specifically,

suppose that an effect $\mathcal{E} : \sigma \oplus \tau \to \sigma' \oplus \tau'$, where σ and τ are signatures, has been decomposed into components $\mathcal{E}_{11} : \sigma \to \sigma'$, $\mathcal{E}_{12} : \sigma \to \tau'$, $\mathcal{E}_{21} : \tau \to \sigma'$ and $\mathcal{E}_{22} : \tau \to \tau'$. The effect obtained from $\mathcal{E}$ by iterating over τ is then given by the categorical trace of $\mathcal{E}$, which is defined as follows:

$$\mathbf{tr}(\mathcal{E}) = \mathcal{E}_{11} + \mathcal{E}_{12}; \sum_{i=0}^{\infty} \mathcal{E}_{22}^i; \mathcal{E}_{21} \tag{26}$$

The existence of this limit is ensured by the well-definedness of the categorical trace. Recursion is also well-defined in QPL, as follows. Suppose we have a recursively defined effect $\mathcal{E} = F(\mathcal{E})$, for some flowchart F. In this case, F defines a Scott-continuous function Φ_F on morphisms, such that the interpretation of $\mathcal{E}$ is given as the least fixed point of Φ_F. This fixed point can be calculated as follows:

$$\mathcal{E} = \sqcup_i F_i \qquad \text{with } F_0 = 0 \text{ and } F_{i+1} = \Phi_F(F_i) \tag{27}$$

$$= \sqcup_i \Phi_F^i(0) \tag{28}$$

where 0 is the zero completely positive map, which corresponds to the divergent program. Again, the existence of these fixed points is ensured by the categorical structure.

QPL is a functional, statically-typed programming language with a denotational semantics, and as such is very different from previously defined quantum programming languages, which do not have a formal semantics and are imperative rather than functional. This is because conventionally, quantum programming environments have been developed with the aim of providing an adequate simulating environment for quantum computation. As such the main concern of such models is efficiency - specifically important when one has to simulate exponentially growing quantum superpositions (and operations on them) in a linear way. However, when investigating quantum programming as a *paradigm*, in which case the aim is to discover and investigate quantum programming concepts and abstractions, expressiveness is the main concern - for which functional models with a denotational semantics are far better suited.

In what follows we derive a weakest precondition semantics for all components of the quantum flowchart language in the case that there are no classical bits. In other words, our program states are given by (1-tuples of) density matrices.

5 Predicate transformers for loop-free flowcharts

In our approach we uniformly consider all basic flowcharts components to be effects in the operator-sum representation [NC00], as in Eq.(11). As such Prop.3.6 already provides a weakest precondition semantics for these atomic flowcharts. Weakest precondition relations for all of the composition techniques of QPL can be found below.

5.1 Sequential composition

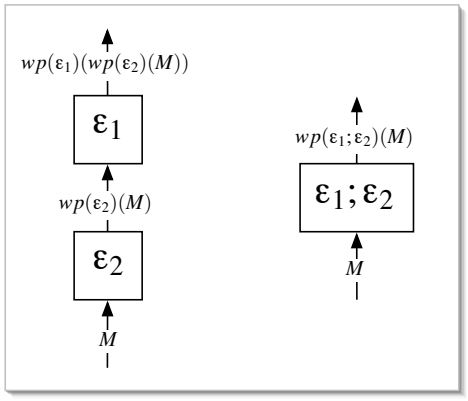

Figure 1: Sequential composition schematically.

Suppose we take the sequential composition of two effects $\mathcal{E}_1$ and $\mathcal{E}_2$, as shown in Fig.1. For the composed effect $\mathcal{E}_1;\mathcal{E}_2$, and arbitrary predicate M, we have that

$$\mathrm{Tr}(M.(\mathcal{E}_1;\mathcal{E}_2)(\rho)) = \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_1;\mathcal{E}_2)(M).\rho) \tag{29}$$

On the other hand, if we calculate weakest preconditions for both effects separately and then compose them sequentially, we obtain

$$\begin{aligned}
\mathrm{Tr}(M.(\mathcal{E}_1;\mathcal{E}_2)(\rho)) &= \mathrm{Tr}(M.\mathcal{E}_2(\mathcal{E}_1(\rho))) && (30)\\
&= \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_2)(M).\mathcal{E}_1(\rho)) && (31)\\
&= \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_1)(\mathrm{wp}(\mathcal{E}_2)(M)).\rho) && (32)\\
&= \mathrm{Tr}((\mathrm{wp}(\mathcal{E}_2);\mathrm{wp}(\mathcal{E}_1))(M).\rho) && (33)
\end{aligned}$$

Comparing Eqs.(29) and (33) we obtain that, for sequential composition, weakest predicate transformers compose as

$$\mathrm{wp}(\mathcal{E}_1;\mathcal{E}_2) = \mathrm{wp}(\mathcal{E}_2);\mathrm{wp}(\mathcal{E}_1) \tag{34}$$

Notice that the order in which predicates are to be transformed is inverse to the order in which the associated effects are applied.

5.2 Parallel composition

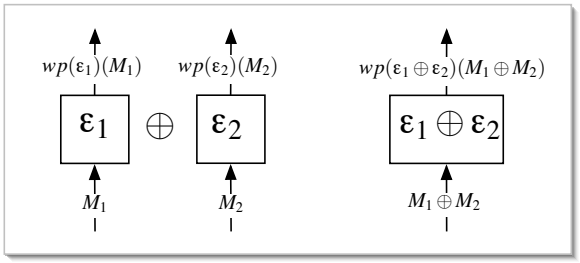

Figure 2: Parallel composition schematically.

Suppose we take the parallel composition of two effects $\mathcal{E}_1$ and $\mathcal{E}_2$, as shown in Fig.2. For the composed effect, given by $\mathcal{E}_1 \oplus \mathcal{E}_2$, we have that

$$\mathrm{Tr}((M_1 \oplus M_2).(\mathcal{E}_1 \oplus \mathcal{E}_2)(\rho_1 \oplus \rho_2)) = \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_1 \oplus \mathcal{E}_2)(M_1 \oplus M_2).(\rho_1 \oplus \rho_2)) \tag{35}$$

On the other hand, if we calculate weakest preconditions for both effects separately and then compose them in a parallel way, we obtain

$$\mathrm{Tr}((M_1 \oplus M_2).(\mathcal{E}_1 \oplus \mathcal{E}_2)(\rho_1 \oplus \rho_2)) \tag{36}$$
$$= \mathrm{Tr}(M_1.\mathcal{E}_1(\rho_1)) + \mathrm{Tr}(M_2.\mathcal{E}_2(\rho_2)) \tag{37}$$
$$= \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_1)(M_1).\rho_1) + \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_2)(M_2).\rho_2) \tag{38}$$
$$= \mathrm{Tr}((\mathrm{wp}(\mathcal{E}_1) \oplus \mathrm{wp}(\mathcal{E}_2))(M_1 \oplus M_2).(\rho_1 \oplus \rho_2)) \tag{39}$$

Comparing Eqs.(35) and (39) we obtain that, for parallel composition, weakest precondition predicate transformers compose as

$$\mathrm{wp}(\mathcal{E}_1 \oplus \mathcal{E}_2) = \mathrm{wp}(\mathcal{E}_1) \oplus \mathrm{wp}(\mathcal{E}_2) \tag{40}$$

5.3 Block structure

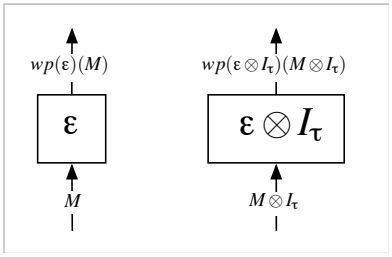

Figure 3: Block structure schematically.

Let us now study what occurs if we weaken a context with extra variables. That is to say, we want to relate weakest precondition predicate transformers for an effect $\mathcal{E}$ operating on a density matrix ρ with those related to an effect $\mathcal{E} \otimes \mathcal{I}_\tau$ operating on $\rho \otimes \tau$. Here τ is an arbitrary context extension of ρ, and $\mathcal{I}_\tau$ is the corresponding identity effect.

For the effect operating on the extended context, with corresponding predicate $M \otimes I_\tau$, we have that

$$\mathrm{Tr}((M \otimes I_\tau).(\mathcal{E} \otimes I_\tau)(\rho \otimes \tau)) = \mathrm{Tr}(\mathrm{wp}(\mathcal{E} \otimes I_\tau)(M \otimes I_\tau).(\rho \otimes \tau)) \tag{41}$$

due to Prop.3.6. On the other hand, calculating weakest preconditions with respect to the contexts ρ and τ separately, i.e. first working out the tensor products occurring in Eq.(41), we obtain

$$
\begin{aligned}
&\mathrm{Tr}((M \otimes I_\tau).(\mathcal{E} \otimes I_\tau)(\rho \otimes \tau)) \\
&= \mathrm{Tr}(M.\mathcal{E}(\rho) \otimes I_\tau.I_\tau(\tau)) \\
&= \mathrm{Tr}(M.\mathcal{E}(\rho) \otimes \tau) \\
&= \mathrm{Tr}(M.\mathcal{E}(\rho)).\mathrm{Tr}(\tau) && \text{(works if } M.\mathcal{E}(\rho) \text{ is diagonalizable (?))} \\
&= \mathrm{Tr}(\mathrm{wp}(\mathcal{E})(M).\rho).\mathrm{Tr}(\tau) && \text{(by Prop.3.6)} \\
&= \mathrm{Tr}((\mathrm{wp}(\mathcal{E})(M) \otimes I_\tau).(\rho \otimes \tau)) && (42)
\end{aligned}
$$

Comparing Eqs.(41) and (42) we obtain that, for an arbitrary context extension τ, weakest precondition predicate transformers compose as

$$\mathrm{wp}(\mathcal{E} \otimes I_\tau) = \mathrm{wp}(\mathcal{E}) \otimes I_\tau \tag{43}$$

6 Iteration and recursion

6.1 Iteration

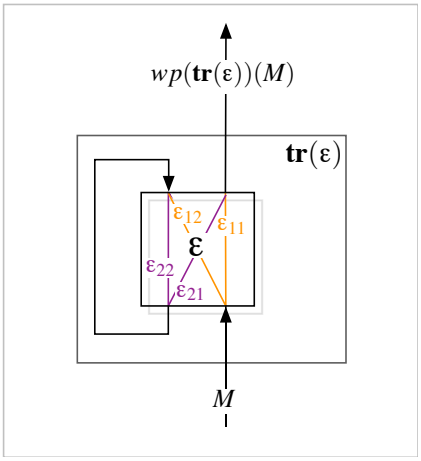

Figure 4: Iteration schematically.

Consider a flowchart which is obtained from an effect $\mathcal{E}$ by introducing a loop.

As explained in Sec.4, the semantics of the flowchart is given by taking the categorical trace over $\mathcal{E}$, denoted $\mathbf{tr}(\mathcal{E})$. For a predicate M and effect $\mathbf{tr}(\mathcal{E})$, we have that

$$\mathrm{Tr}(M.(\mathbf{tr}(\mathcal{E}))(\rho)) = \mathrm{Tr}(\mathrm{wp}(\mathbf{tr}(\mathcal{E}))(M).\rho) \tag{44}$$

On the other hand, by iterating explicitly we obtain

$$\begin{aligned}
&\mathrm{Tr}(M.(\mathbf{tr}(\mathcal{E}))(\rho)) \\
&= \mathrm{Tr}(M.(\mathcal{E}_{11} + (\mathcal{E}_{12}; \textstyle\sum_{i=o}^{\infty} \mathcal{E}_{22}^{i}; \mathcal{E}_{21}))(\rho)) && \text{(by Eq.(26))} \\
&= \mathrm{Tr}(M.\mathcal{E}_{11}(\rho)) + \mathrm{Tr}(M.(\mathcal{E}_{12}; \textstyle\sum_{i=o}^{\infty} \mathcal{E}_{22}^{i}; \mathcal{E}_{21})(\rho)) && \text{(by linearity)} \\
&= \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_{11})(M).\rho) + \mathrm{Tr}((\mathrm{wp}(\mathcal{E}_{21}); \mathrm{wp}(\textstyle\sum_{i=o}^{\infty} \mathcal{E}_{22}^{i}); \mathrm{wp}(\mathcal{E}_{12}))(M).\rho) && \text{(by Eq.(34))} \\
&= \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_{11})(M).\rho) + \mathrm{Tr}((\mathrm{wp}(\mathcal{E}_{21}); \textstyle\sum_{i=o}^{\infty} \mathrm{wp}(\mathcal{E}_{22}^{i}); \mathrm{wp}(\mathcal{E}_{12}))(M).\rho) && \text{(by Prop.3.9)} \\
&= \mathrm{Tr}(\mathrm{wp}(\mathcal{E}_{11})(M).\rho) + \mathrm{Tr}((\mathrm{wp}(\mathcal{E}_{21}); \textstyle\sum_{i=o}^{\infty} \mathrm{wp}(\mathcal{E}_{22})^{i}; \mathrm{wp}(\mathcal{E}_{12}))(M).\rho) && \text{(by Eq.(34))} \\
&= \mathrm{Tr}((\mathrm{wp}(\mathcal{E}_{11}) + \mathrm{wp}(\mathcal{E}_{21}); \textstyle\sum_{i=o}^{\infty} \mathrm{wp}(\mathcal{E}_{22})^{i}; \mathrm{wp}(\mathcal{E}_{12}))(M).\rho) && \text{(by linearity)}
\end{aligned} \tag{45}$$

Comparing Eqs.(44) and (45), we obtain that the weakest predicate transformer for an iterated effect $\mathbf{tr}(\mathcal{E})$ is obtained as

$$\mathrm{wp}(\mathbf{tr}(\mathcal{E})) = \mathrm{wp}(\mathcal{E}_{11}) + \mathrm{wp}(\mathcal{E}_{21}); \sum_{i=o}^{\infty} \mathrm{wp}(\mathcal{E}_{22})^{i}; \mathrm{wp}(\mathcal{E}_{12}) \tag{46}$$

Moreover, the existence of the limit in Eq.(46) is guaranteed due to Prop.3.4.

6.2 Recursion

Consider an effect which is defined recursively, i.e. an effect $\mathcal{E}$ satisfying the equation $\mathcal{E} = F(\mathcal{E})$, where F is a flowchart. The required fixed point solution to this recursive equation is given by Eqs.(27) and (28). Using Prop.3.6 for such a recursive flowchart, we obtain

$$\mathrm{Tr}(M.\mathcal{E}(\rho)) = \mathrm{Tr}(\mathrm{wp}(\mathcal{E})(M).\rho) \tag{47}$$

On the other hand, if we work out the weakest precondition relations using Eq.(27), we obtain

$$\begin{aligned}\mathrm{Tr}(M.\mathcal{E}(\rho)) &= \mathrm{Tr}(M.(\sqcup_i F_i)(\rho)) \\ &= \mathrm{Tr}(\mathrm{wp}(\sqcup_i F_i)(M).\rho) \qquad \text{(by Prop.3.6)} \\ &= \mathrm{Tr}((\sqcup_i \mathrm{wp}(F_i))(M).\rho) \qquad \text{(by Prop.3.9)} \qquad (48) \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (49)\end{aligned}$$

Comparing Eqs.(47) and (49), we obtain that the weakest precondition predicate transformer for a recursively defined effect $\mathcal{E} = F(\mathcal{E})$ is obtained as

$$\mathrm{wp}(\mathcal{E}) = \sqcup_i \mathrm{wp}(F_i) = \sqcup_i \mathrm{wp}(\Phi_F^i(0)) \tag{50}$$

The existence of the least upper bound in Eq.(50) is guaranteed due to Prop.3.4. This result depends of course on the concrete recursive specification considered. Specifically, one needs to determine Φ_F in order to determine the weakest precondition predicate transformer corresponding to an effect $\mathcal{E}$, defined recursively as $\mathcal{E} = F(\mathcal{E})$.

7 Examples

In this section we look at some specific situations and their weakest precondition predicate transformers.

7.1 Unitary transformation

Suppose we transform a density matrix unitarily, such that the associated effect $\mathcal{E}_U$ is given by

$$\mathcal{E}_U = U\rho U^\dagger \tag{51}$$

Hence we have a degenerate operator-sum representation. The weakest precondition with respect to $\mathcal{E}_U$ for an arbitrary predicate M is then given by

$$\mathrm{wp}(\mathcal{E}_U)(M) = U^\dagger M U \tag{52}$$

7.2 Projection

Let us now look at the situation where we carry out a projection, such that we have an effect $\mathcal{E}_P$ acting on density matrices as follows

$$\mathcal{E}_P = P\rho P^\dagger \tag{53}$$

The weakest precondition with respect to $\mathcal{E}_P$ for an arbitrary predicate M is then given by

$$\mathrm{wp}(\mathcal{E}_P)(M) = P^\dagger M P \tag{54}$$

Now suppose the predicate M is itself a projection operator, denoted P_1. Then there are two possibilities:

1. If $[P, P_1] = 0$, then $\mathrm{wp}(\mathcal{E}_P)(P_1) = PP_1 = P_1P$, which is again a projector.
2. If $[P, P_1] \neq 0$, then $\mathrm{wp}(\mathcal{E}_P)(P_1) = P^\dagger P_1 P$, which is not even a projector. A specific instance is shown in Fig.5.

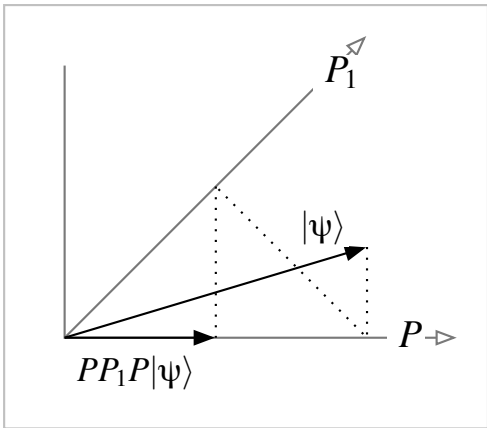

Figure 5: Two projectors, with a 45° angle between them, acting on a state $|\psi\rangle$.

7.3 Grover's algorithm

We now look at a less trivial example, namely that of Grover's algorithm, also known as the database search algorithm [Gro96]. Suppose we transform a density matrix unitarily, such that the associated effect $\mathcal{E}_U$ is given by

$$\mathcal{E}_G = G^n \rho (G^n)^\dagger \tag{55}$$

where the Grover operator G is given by

$$G = (2|\psi\rangle\langle\psi| - I)O \tag{56}$$

Here O is a quantum oracle, which labels solutions to the search problem, while $(2|\psi\rangle\langle\psi| - I)$ is the *inversion about mean* operation. Geometrically, the Grover operator is a rotation in the two-dimensional space spanned by the states

$$|\alpha\rangle = \frac{1}{\sqrt{N-K}} \sum_{x \notin \{solutions\}} |x\rangle \tag{57}$$

$$|\beta\rangle = \frac{1}{\sqrt{K}} \sum_{x \in \{solutions\}} |x\rangle \tag{58}$$

where N is the dimension of the state space and K is the number of solutions to the search problem in question. More specifically, G can be decomposed as

$$G = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \quad \text{with } \sin\theta = \frac{2\sqrt{K(N-K)}}{N} \tag{59}$$

Applying (52), we obtain as weakest precondition with respect to $\mathcal{E}_G$ for an arbitrary predicate M

$$\mathrm{wp}(\mathcal{E}_G)(M) = (G^n)^\dagger M G^n \tag{60}$$

Now suppose we take M to be a projector on the solution space, that is

$$M = |\beta\rangle\langle\beta| \tag{61}$$

Using (59), we see that $(G^n)^\dagger|\beta\rangle$ corresponds to n rotations over an angle of $-\theta$ in the state space spanned by $|\alpha\rangle$ and $|\beta\rangle$. In other words, through repeated applications of $\mathrm{wp}(\mathcal{E}_G)$ to $|\beta\rangle\langle\beta|$, one can approximate arbitrary preconditions[3]. This property is the dual of the one that states that Grover's algorithm is successful for arbitrary initial states, after a suitable number of iterations [BBB$^+$98].

[3]To decrease the error, one can always enlarge the state space, thus obtaining a smaller rotation angle θ.

8 Conclusion

In this article, we have developed the predicate transformer and weakest precondition formalism for quantum computation. We have done this by first noting that the quantum analogue to predicates are expectation values of quantum measurements, given by the expression $\mathrm{Tr}(M\rho)$. Then we have defined the concept of weakest preconditions within this framework, proving that a weakest precondition exists for arbitrary completely positive maps and observables. We have also worked out the weakest precondition semantics for the Quantum Programming Language (QPL) developed in [Sel03]. QPL is the first model for quantum computation with a denotational semantics, and as such the first serious attempt to design a quantum programming language intended for programming quantum algorithms compositionally.

One can now investigate how properties transform during a quantum computation, as for example (dis)entanglement, amplitude distributions and so forth. These are exactly the properties that lie at the basis of quantum computation. However, currently it is not at all clear how one should exploit them systematically when designing quantum algorithms. Therefore, we feel that this work can contribute significantly in the research towards the quantum programming paradigm in general. What is important to note is that we can write useful specifications - as in the Grover search algorithm - in terms of our predicates.

With this development in place one can envisage a goal-directed programming methodology for quantum computation. Of course one needs more experience with quantum programming idioms and the field is not yet ready to produce a "quantum" Science of Programming[4]. It seems to us that in the field of communication protocols, such as those based on teleportation, we now have a good stock of ideas and examples which could be used as the basis of methodologies in this context. We are actively thinking about this.

The most closely related work - apart from Selinger's work on his programming language - is the work by Sanders and Zuliani [SZ00] which develops a guarded command language used for developing quantum algorithms. This is a very interesting paper and works seriously towards developing a methodology for quantum algorithms. However, they use probability and nondeterminism to capture probabilistic aspects of quantum algorithms. Ours is an *intrinsically quantum* framework. The notion of weakest precondition that we develop here is not related to anything in their framework. There are other works [BS04] - as yet unpublished - in which a quantum dynamic logic is being developed. Clearly such work will be related though they use a different notion of pairing. Also the work in [Eda04]

[4] a classic text by Gries [Gri81] on developing imperative programs.

is related and merits further investigation. Edalat uses the interval domain of reals rather than the reals as the values of the entries in his density matrices. This seems a good way to deal with uncertainity in the values.

There is a large literature on probabilistic predicate transformers including several papers from the probabilistic systems group at Oxford. A forthcoming book [MM04] gives an expository account of their work. We hope to develop an analogous theory of refinement and work out laws and healthiness conditions in the purely quantum setting as they did for the probabilistic setting. Of course that is a major enterprise.

One pleasant aspect of the present work is that it is language independent; though we have used it to give the semantics of QPL the weakest precondition formalism stands on its own. We can therefore apply it to other computational models that are appearing and for which language ideas have not yet emerged.

Acknowledgements

It is a pleasure to thank Samson Abramsky, Bob Coecke, Elham Kashefi and Peter Selinger for helpful discussions. This research was funded by the FWO (Flanders) for the first author; for the second authos in part by a grant from NSERC (Canada) and in part by a visiting fellowship from EPSRC (U.K.).

References

[BBB+98] David Biron, Ofer Biham, Eli Biham, Markus Grassl, and Daniel A. Lidar. Generalized Grover search algorithm for arbitrary initial amplitude distribution. *Quantum Computing and Quantum Communications*, LNCS 1509:140–147, 1998.

[BS04] A. Baltag and S. Smets. Quantum dynamic logic. 2004.

[Deu85] David Deutsch. Quantum theory, the church-turing principle and the universal quantum computer. *Proc. R. Soc. Lond.*, pages A400: 97–117, 1985.

[Dij76] E. W. Dijkstra. *A Discipline of Programming*. Prentice-Hall, 1976.

[DJ92] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond. A*, pages 439–553, 1992.

[Eda04] Abbas Edalat. An extension of gleason's theorem for quantum computation. *International Journal of Theoretical Physics*, 2004. (to appear).

[Gri81] D. Gries. *The Science of Programming*. Springer-Verlag, 1981.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *ACM Symposium on Theory of Computing*, pages 212–219, 1996.

[Gro97] Lov K. Grover. Quantum mechanics helps in searching for a needle in a haystack. arXive e-print quant-ph/9706033, 1997.

[Joh82] Peter Johnstone. *Stone Spaces*, volume 3 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 1982.

[Koz81] D. Kozen. Semantics of probabilistic programs. *Journal of Computer and Systems Sciences*, 22:328–350, 1981.

[Koz85] D. Kozen. A probabilistic PDL. *Journal of Computer and Systems Sciences*, 30(2):162–178, 1985.

[Löw34] K. Löwner. Uber monotone matrixfunktionen. *Mathematische Zeitschrift*, 38:177–216, 1934.

[MM04] C. Morgan and A. McIver. *Abstraction, refinement and proof for probabilistic systems*. Springer-Verlag, 2004. To appear.

[NC00] Michael Nielsen and Isaac Chuang. *Quantum computation and quantum information*. Cambridge university press, 2000.

[Per95] Asher Peres. *Quantum Theory: Concepts and Methods*. Kluwer Academic Publishers, 1995.

[Plo83] G. D. Plotkin. Lecture notes on domain theory. 1983.

[Sel03] Peter Selinger. Towards a quantum programming language. *Mathematical Structures in Computer Science*, 2003. to appear.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *IEEE Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Computing*, (26):1484–1509, 1997.

[Smy83] M. Smyth. Powerdomains and predicate transformers. In J. Diaz, editor, *Proceedings of the International Colloquium On Automata Languages And Programming*, pages 662–676. Springer-Verlag, 1983. Lecture Notes In Computer Science 154.

[SZ00] J. W. Sanders and P. Zuliani. Quantum programming. In *Mathematics of Program Construction*, number 1837, pages 80–99. Springer-Verlag, 2000.