Under consideration for publication in Math. Struct. in Comp. Science

Quantum Weakest Preconditions

ELLIE D'HONDT [1] and PRAKASH PANANGADEN[2†]

[1] *Vrije Universiteit Brussel, Belgium*
[2] *McGill University, Montreal, Canada*

Received 24 January 2005; Revised 15 December 2005

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 immediately from the Kraus representation theorem. As an example we give the semantics of Selinger's language in terms of our weakest preconditions. We also cover some specific situations and exhibit an interesting link with stabilizers.

1. Introduction

Quantum computation is rapidly becoming a significant topic in theoretical computer science. To be sure, there still are essential technological and conceptual problems to overcome in building functional quantum computers. Nevertheless there are fundamental new insights into quantum computability (Deutsch, 1985; Deutsch and Jozsa, 1992), quantum algorithms (Grover, 1996; Shor, 1994) and into the nature of quantum mechanics itself (Peres, 1995, Part III), particularly with the emergence of quantum information theory (Nielsen and Chuang, 2000, Ch. 12).

These developments inspire one to consider the problems of programming 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 quantum algorithms are currently programmed at a very low level – comparable to classical computing 60 years ago. In the search for structure in the space of quantum algorithms one is led to consider issues like compositionality, semantics, type systems and logics; these are issues that usually arise in the context of programming languages. The present paper is situated in the nascent area of quantum programming methodology and the design and semantics of quantum programming languages. We extend the well-known paradigm of *weakest preconditions* (Hoare, 1969; Dijkstra, 1976) to the quantum context.

† Ellie D'Hondt was funded by the FWO and the VUB (Flanders) and Prakash Panangaden was funded in part by a grant from NSERC (Canada) and in part by a visiting fellowship from EPSRC (U.K.).

The influence of Dijkstra's work on weakest preconditions has been deep and pervasive and even led to textbook level expositions of the subject (Gries, 1981). 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 eventually come out of the present work.

In this paper we make two contributions: first, we develop the appropriate quantum analogue of weakest preconditions and develop the duality theory. Rather than reducing quantum computation to probabilistic computation and using well-known ideas from this setting (Kozen, 1981; Kozen, 1985), we define quantum weakest preconditions directly. It turns out that the same beautiful duality between state-transformer (forwards) and predicate-transformer (backwards) semantics that one finds in the traditional (Smyth, 1983; Plotkin, 1983) and the probabilistic settings (Kozen, 1985) appears in the quantum setting. This is related to the fact that when state transformers are specified to be completely positive maps, we can prove the existence of corresponding weakest preconditions in a very general way using a powerful mathematical result called the Kraus representation theorem (Nielsen and Chuang, 2000, Sec. 8.2.4). In fact the correspondence is very much more direct in this case than in the case of conventional or probabilistic languages.

Second, we write the detailed weakest precondition semantics for a particular quantum programming language. Quantum programming languages have started to appear recently. Perhaps the best known is the quantum flow chart language (Selinger, 2003), also referred to as QPL, which is based on the slogan "quantum data and classical control". QPL has a clean denotational semantics and a clear conceptual basis; we give an alternative weakest precondition semantics for this language. It should be noted, however, that our notion of weakest preconditions and the basic existence results are *language independent*.

The structure of this paper is as follows. In Sec. 2 the general setup, in particular quantum state transformers and quantum predicates, is laid out. Next, in Sec. 3 we define quantum weakest preconditions and healthy predicate transformers, proving their existence for arbitrary completely positive maps and observables. In Sec. 4 we summarize the basic structure of Selinger's language, and develop its weakest precondition semantics. We apply our results to specific situations such as Grover's algorithm and stabilizers in Sec. 5, and conclude with Sec. 6.

2. The quantum framework

In this section we define the main concepts on which our theory of quantum weakest preconditions is based. We first give a general overview, after which we specify concrete definitions for quantum states and state transformers in Sec. 2.1 and for quantum predicates in Sec. 2.2.

Traditionally, there are several means of developing formal semantics for programming languages. In the operational semantics for an imperative language one has a notion of *states*, typically denoted s, 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 but 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 to determine what must be true of the initial input. States are replaced by *predicates* p over the state space, together with a satisfaction relation $\models$. Language constructs are interpreted as predicate transformers. 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 in Sec. 3.4 they are *dual* to each other. The situation for deterministic languages can be found in the first column of Table 1.

In the world of probabilistic programs one sees the same duality in action, after suitably generalizing the notions of states and predicates. *Probability distributions* now play the role of states. There are, of course, states as before and, in a particular execution, there is only one state at every stage. However, 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 (Kozen, 1985) that predicates are measurable functions – or random variables, to use the probability terminology. We note that a special case of random variables are characteristic functions, which are more easily recognizable as the analogues of predicates; in fact they *are* predicates. In a probabilistic setting one has *expectation* value rather than truth: truth values now lie in $[0, 1]$ rather than in $\{0, 1\}$. Third, the pairing between measurable functions f and probability distributions μ is now given by the integral, which is the probabilistic expression of the expectation value. These measurable functions are to be viewed as *observations*, which may or may not lead to termination. The pairing between f and μ then expresses the probability with which termination is achieved when observing f. For probabilistic languages the second column of Table 1 summarizes the main concepts.

For the quantum world we again need a notion of state – or, more precisely, probability distributions over possible states – a notion of predicate, and a pairing. Our choices are very much guided by the probabilistic case, but we are *not* claiming that quantum computation can be seen as a special case of classical probabilistic computation. Instead, we take *density matrices* as the analogue of probability distributions, while for predicates we take the *observables* of the system. These are given by (a certain restricted class of) Hermitian operators. Finally, the notion of a pairing is again the expectation value, but given by the rules of quantum mechanics; that is we have $\mathrm{tr}(M\rho)$, where tr stands for the usual trace from linear algebra, ρ is a density matrix and M an observable. Throughout this paper we work with finite-dimensional Hilbert spaces and one can think of M and ρ as matrices. We discuss these concepts in more depth in Secs. 2.1 and 2.2; a summary can be found in the last column of Table 1. Note however that, just as for the probabilistic case, the pairing $\mathrm{tr}(M\rho)$ may be interpreted as the probability of termination when observation M is made in the state ρ.

Table 1. *Comparing situations.*

Deterministic	Probabilistic	Quantum
states s	probability distributions μ	density matrices ρ
predicates p	measurable functions f	observables M
satisfaction $s \models p$	expectation value $\int f d\mu$	quantum expectation value $\mathrm{tr}(M\rho)$

Why cannot one just use probabilistic predicates and the general theory of probabilistic predicate transformers in a quantum context? The following simple example – due to one of the referees – illustrates why. Suppose that we have a two-dimensional Hilbert space of states with basis vectors written $|0\rangle$ and $|1\rangle$. Two other states in this Hilbert space are $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. We use the notation $\{|\psi\rangle\}$ for the density matrix $|\psi\rangle\langle\psi|$ and write convex combinations like $\lambda\{|\psi\rangle\} + (1-\lambda)\{|\psi\rangle\}$ for the density matrix of a mixed state, i.e. an ensemble. Now consider the measurable function f defined by:

$$\begin{aligned} f(|0\rangle) &= 0 \\ f(|1\rangle) &= 0 \\ f(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)) &= 1 \\ f(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)) &= 1 \ . \end{aligned} \tag{1}$$

This function is indeed measurable but not linear and cannot correspond to any kind of physical observable or measurement. To see what happens, consider the ensemble $\rho = \frac{1}{2}\{|0\rangle\} + \frac{1}{2}\{|1\rangle\}$. When f is applied to this one obtains 0. However, when f is applied to the ensemble $\rho' = \frac{1}{2}\{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\} + \frac{1}{2}\{\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\}$ we obtain the value 1. The point is that ρ and ρ' are physically indistinguishable, and thus one cannot have a physical observable that tells these "two" ensembles apart. When developing a theory of predicates and predicate transformers one must therefore restrict to mathematical objects that are compatible with the linear structure of quantum mechanics. It is a conceptual error to think that quantum mechanics can be understood just with probabilistic constructs. We note that the work in (Butler and Hartel, 1999), which uses probabilistic predicates to analyze Grover's algorithm (Grover, 1996), avoids this conundrum because it considers only pure-state situations.

2.1. *Quantum states and state transformers*

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 (Peres, 1995). 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 *probability distributions* over the states in a Hilbert space. In quantum mechanics this situation is characterized by *density matrices*, of which a good expository discussion appears in (Nielsen and Chuang, 2000, Ch. 2). Concretely, a density matrix ρ on a Hilbert space $\mathcal{H}$ is a *positive operator*, that is, for all states $|x\rangle$ in $\mathcal{H}$ one requires that $\langle x|\rho x\rangle \geq 0$, with furthermore $\mathrm{tr}\rho \leq 1$. The reason why we do not have the usual equality is that we do not assume that everything is always normalized. Hence, in order to interpret a density matrix as a probability distribution one first needs to renormalize if necessary. This is a bit of a nuisance if one wants a direct interpretation of the density matrix at every stage of the computation; however, one does recover the probabilities correctly if one starts with a normalized density matrix at the start of a computation and multiplies out everything at the end. This convention saves some notational overhead and is used by Selinger (Selinger, 2003). We denote the set of all density matrices over a Hilbert space $\mathcal{H}$ by $\mathcal{DM}(\mathcal{H})$.

As we have mentioned in the above, forward operational semantics is described by quantum state transformers. The properties of such state transformers are now well understood. A physical transformation must take a density matrix to a density matrix. Thus it seems reasonable to require that physical operations correspond to positive maps, which are linear maps that take a positive operator to a positive operator. However, 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. Indeed, this means that if we formally regard some system as part of another far away system which we do not touch (that is, to which we apply the identity transformation), then suddenly we have an unphysical transformation. A simple example is provided by the transpose operation, which is a positive map while its tensor with an identity is not. Therefore, we need the stronger requirement that physical operations are *completely* positive, a property which is defined as follows.

Definition 2.1. A map $\mathcal{E}$ is *completely positive* when it takes density matrices to density matrices, and likewise for all trivial extensions $I \otimes \mathcal{E}$.

Note that such a map may operate between distinct Hilbert spaces, that is in general we have $\mathcal{E} : \mathcal{DM}(\mathcal{H}_1) \to \mathcal{DM}(\mathcal{H}_2)$. We denote by $\mathcal{CP}(\mathcal{H}_1, \mathcal{H}_2)$ the set of all such maps, and write $\mathcal{CP}(\mathcal{H})$ for $\mathcal{CP}(\mathcal{H}, \mathcal{H})$.

We frequently rely on the Kraus representation theorem for completely positive maps.

Theorem 2.1 (Kraus Theorem). The map $\mathcal{E} : \mathcal{DM}(\mathcal{H}_1) \to \mathcal{DM}(\mathcal{H}_2)$ is a completely positive map if and only if for all $\rho \in \mathcal{DM}(\mathcal{H}_1)$ we have that

$$\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger \tag{2}$$

for some set of operators $\{E_i : \mathcal{H}_1 \to \mathcal{H}_2\}$, with $\sum_i E_i^\dagger E_i \leq I$.

The condition on the E_i ensures that trace of the density matrix never increases. Eq.(2) is also known as the *operator-sum representation.* The proof to this theorem can be found, for example, in (Nielsen and Chuang, 2000, Sec. 8.2.4). Note there is nothing in the theorem that says that the E_i are unique.

2.2. *Quantum predicates*

In this section, we 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 *Scott-continuous* in order to deal with programming language aspects such as recursion and iteration.

As argued above, quantum predicates are given by Hermitian operators. However, general Hermitian operators will not yield a satisfactory logical theory with the duality that we are looking for. We need to restrict to positive operators and - in order to obtain least upper bounds for increasing sequences - we need to bound them. More precisely, we have the following definition.

Definition 2.2. A *predicate* is a positive - hence Hermitian - operator with the maximum eigenvalue bounded by 1.

The reason for taking predicates to have the maximum eigenvalue bounded by 1 is in order to get a complete partial order (CPO); we clarify this below. Since our predicates are positive operators their eigenvalues are real and positive. We denote the set of all predicates on a Hilbert space $\mathcal{H}$ by $\mathcal{P}(\mathcal{H})$.

Proposition 2.1. Let M be a Hermitian operator. Then $0 \leq \mathrm{tr}(M\rho) \leq 1$ holds for all density matrices ρ if and only if M is positive and its eigenvalues are bounded by 1.

Proof. Note that for any element $|\psi\rangle$ of $\mathcal{H}$ we have $\mathrm{tr}(M|\psi\rangle\langle\psi|) = \langle\psi \mid M \mid \psi\rangle$. Assume that $0 \leq \mathrm{tr}(M\rho) \leq 1$ for all density matrices ρ. Choose $\rho = |\psi\rangle\langle\psi|$ where $|\psi\rangle$ is an arbitrary normalized vector. We have $0 \leq \mathrm{tr}(M|\psi\rangle\langle\psi|) = \langle\psi \mid M \mid \psi\rangle$, which says that M is positive. Now choose $|\psi\rangle$ to be a normalised eigenvector of M with eigenvalue λ, necessarily real and positive, so we have that $\mathrm{tr}(M|\psi\rangle\langle\psi|) = \langle\psi \mid M \mid \psi\rangle = \lambda\langle\psi|\psi\rangle = \lambda \leq 1$. Thus the eigenvalues are bounded by 1. The converse is obvious once we note that any density matrix is a convex combination of density matrices of the form $|\psi\rangle\langle\psi|$. $\square$

Thus we could have defined predicates as positive operators M such that for every density matrix ρ we have $0 \leq \mathrm{tr}(M\rho) \leq 1$. This exhibits the predicates as "dual" to density matrices.

We define an ordering as follows.

Definition 2.3. For matrices M and N in $\mathbb{C}^{n\times n}$ we define $M \sqsubseteq N$ if $N - M$ is positive.

This order is known in the literature as the *Löwner partial order* (Löwner, 1934). Note that this definition can be rephrased in the following way, where $\mathcal{DM}(\mathcal{H})$ denotes the set of all density matrices.

Proposition 2.2. $M \sqsubseteq N$ 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, $\mathrm{tr}((N - M).|x\rangle\langle x|x) \geq 0$. By linearity of the trace and the fact that the spectral theorem holds for all $\rho \in \mathcal{DM}(\mathcal{H})$ we obtain the desired result. For the converse, take all pure states $\rho = |x\rangle\langle x|$. Then we find that for all $x \in \mathcal{H}$ we have $\langle x|N - M|x\rangle \geq 0$, or in other words $M \sqsubseteq N$. □

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 2.3. The poset $(\mathcal{P}(\mathcal{H}), \sqsubseteq)$ is a complete partial order (CPO), i.e. it contains least upper bounds of increasing sequences. □

Taking predicates to be *bounded* Hermitian operators leads to Prop. 2.3, which guarantees the existence of fixpoints and thus allows for the formal treatment of iteration and recursion in Sec. 4.

3. Quantum weakest preconditions and duality

In this section we elaborate our theory of quantum weakest preconditions. We first give the main definitions in Sec. 3.1, after which we explore healthiness conditions in Sec. 3.2. Next, we investigate weakest precondition predicate transformers for completely positive maps in Sec. 3.3. With the latter results we obtain a duality between the forward state transformer semantics and the backward weakest precondition semantics in Sec. 3.4.

3.1. *Definitions*

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 $\mathrm{tr}(M\rho)$. Preconditions for a quantum program $\mathcal{Q}$ – described in an unspecified quantum programming language – are defined as follows. We write $\mathcal{Q}$ for the program as well as for the trace-nonincreasing completely positive map that it denotes.

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

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

We also introduce 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.

The exact syntax of the quantum program $\mathcal{Q}$ is left unspecified deliberately, as we want to state these definitions without committing to any particular framework. Of course we expect $\mathcal{Q}$ to implement at least some transformation on density matrices, in particular we may think of $\mathcal{Q}$ as implementing a completely positive map. Note however, that Def. 3.1, as well as Def. 3.2 below, does not exclude other possibilities. For example we could also investigate possibilities proposed in (Shaji and Sudarshan, 2005), where it is argued that positive but not completely positive or even not positive maps are also good candidates for describing open quantum evolutions.

This definition deserves motivation. If all density matrices were normalized then it is easy to motivate Def. 3.1: if we want the expectation value of N in the state $\mathcal{Q}(\rho)$ to be above some real number r, say, then this is guaranteed if the expectation value of M in the state ρ is above r. In the case of our unnormalized density matrices we have to do a little calculation to see that the same holds. We write the expectation value of M in a state (density matrix) ρ as $\langle M\rangle_\rho$. Now we assume that M, N and $\mathcal{Q}$ satisfy the conditions of Def. 3.1. Let ρ be any (unnormalized) density matrix and let its normalized version be $\overline{\rho} = \rho/\mathrm{tr}(\rho)$. Then we have

$$\begin{aligned} \langle M\rangle_\rho &= \mathrm{tr}(M\overline{\rho}) \\ &= \frac{1}{\mathrm{tr}(\rho)}\cdot \mathrm{tr}(M\rho) \\ &\leq \frac{1}{\mathrm{tr}(\rho)}\cdot \mathrm{tr}(N\mathcal{Q}(\rho)) \\ &= \frac{\mathrm{tr}(\mathcal{Q}(\rho))}{\mathrm{tr}(\rho)}\cdot\frac{1}{\mathrm{tr}(\mathcal{Q}(\rho))}\mathrm{tr}(N\mathcal{Q}(\rho)) \\ &= \frac{\mathrm{tr}(\mathcal{Q}(\rho))}{\mathrm{tr}(\rho)}\cdot\langle N\rangle_{\mathcal{Q}\rho} \\ &\leq \langle N\rangle_{\mathcal{Q}(\rho)}\ . \end{aligned} \tag{4}$$

Thus, even though the density matrices are not normalized and we cannot read the expectations *directly* at every intermediate stage, Def. 3.1 still has the same import as in the normalized case, as well as in the case of probabilistic predicate transformers.

From this we define weakest preconditions in the usual way.

Definition 3.2. A *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 $L\{\mathcal{Q}\}M$ implies $L \sqsubseteq \mathrm{wp}(\mathcal{Q})(M)$.

Note that *weakest* in this context is equal to *largest*; indeed, a larger predicate means that Eq.(3) holds for more initial states ρ, and thus corresponds to a weaker constraint. The weakest precondition predicate transformer for a program $\mathcal{Q}$, if it exists, is denoted $\mathrm{wp}(\mathcal{Q}) : \mathcal{P}(\mathcal{H}_2) \to \mathcal{P}(\mathcal{H}_1)$, where $\mathcal{H}_2$ and $\mathcal{H}_1$ are the output and input Hilbert spaces respectively.

3.2. *Healthiness conditions*

In analogy with (Dijkstra, 1976), we want to formulate *healthiness conditions* for quantum predicate transformers. These are important because they characterize exactly those programs that can be given a weakest precondition semantics which is dual to its forwards state transformer semantics. Moreover, healthiness conditions allow one to prove general laws for reasoning about programs. The healthiness conditions we propose for the quantum case are *linearity* and *complete positivity*, leading to the following definition.

Definition 3.3. A *healthy* predicate transformer $\alpha : \mathcal{P}(\mathcal{H}_2) \to \mathcal{P}(\mathcal{H}_1)$ is a predicate transformer that is *linear* and *completely positive*, i.e. it it takes predicates to predicates and likewise for all trivial extensions $I \otimes \alpha$. We denote the associated space of healthy predicate transformers as $\mathcal{PT}(\mathcal{H}_2, \mathcal{H}_1)$.

As we shall see in the following section these conditions all hold in the framework where quantum programs correspond to completely positive maps. Linearity is certainly a requirement in the inherently linear context of quantum mechanics, as the example given in Sec. 2 clearly shows. Just as in the probabilistic case (Morgan and McIver, 2004), linearity implies the analogues of some of the healthiness conditions for deterministic programs, namely feasibility, which means that $\mathrm{wp}(\mathcal{Q})(0) = 0$, monotonicity and continuity. These proofs are easy and are left to the reader. The requirement that predicate transformers should be completely positive on $\mathcal{P}(\mathcal{H})$, is a very natural one. Indeed, if α 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.

We equip $\mathcal{PT}(\mathcal{H}_2, \mathcal{H}_1)$ with an order structure by extending the Löwner order on predicates in the following way.

Definition 3.4. For healthy predicate transformers α and β in $\mathcal{PT}(\mathcal{H}_2, \mathcal{H}_1)$ we define $\alpha \sqsubseteq \beta$ if $\beta - \alpha$ is a healthy predicate transformer.

If $\alpha \sqsubseteq \beta$ then for all predicates $M \in \mathcal{P}(\mathcal{H}_2)$ we have that $\alpha(M) \sqsubseteq \beta(M)$, where $\alpha(M)$ and $\beta(M)$ are predicates on $\mathcal{H}_1$. Requiring only this would be the obvious extension of the Löwner order, however, since we are working in the space of healthy predicate transformers we also need to demand that $\beta - \alpha$ is completely positive. That is, for all extended predicates $M_e \in \mathcal{P}(\mathcal{H}_2 \otimes \mathcal{H})$ we have $(\alpha \otimes I_{\mathcal{H}})(M_e) \sqsubseteq (\beta \otimes I_{\mathcal{H}})(M_e)$. We then have the following result.

Proposition 3.1. The poset $(\mathcal{PT}(\mathcal{H}_2, \mathcal{H}_1), \sqsubseteq)$ is a CPO.

Proof. The proof is analogous to that of Lemma 6.4 of (Selinger, 2003). □

Note that the CPO structure as defined on predicates $\mathcal{P}(\mathcal{H})$ and associated predicate transformers $\mathcal{PT}(\mathcal{H})$ is identical to that for density matrices $\mathcal{DM}(\mathcal{H})$ and associated completely positive maps $\mathcal{CP}(\mathcal{H})$, as defined in (Selinger, 2003).

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

Proposition 3.2. The operator α is a healthy predicate transformer if and only if one has that

$$\forall M \in \mathcal{P}(\mathcal{H}). \alpha(M) = \sum_u A_u^\dagger M A_u \tag{5}$$

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

3.3. *Predicate transformers for completely positive maps*

Let us now consider the following framework: the forward semantics of a quantum program $\mathcal{Q}$ is given by a trace-nonincreasing completely positive map $\mathcal{E} \in \mathcal{CP}(\mathcal{H}_1, \mathcal{H}_2)$, which we write as $[\![\mathcal{Q}]\!] = \mathcal{E}$. 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 3.3. $\forall \mathcal{E} \in \mathcal{CP}(\mathcal{H}_1, \mathcal{H}_2)$ and $N \in \mathcal{P}(\mathcal{H})$, $\mathrm{wp}(\mathcal{E})(N)$ exists and is unique. Furthermore, we have that

$$\forall \rho . \mathrm{tr}(\mathrm{wp}(\mathcal{E})(N)\rho) = \mathrm{tr}(N\mathcal{E}(\rho)) \tag{6}$$

Proof.

To prove existence, take an arbitrary predicate N and operation $\mathcal{E}$. From the Kraus representation theorem stated in Sec. 2.1, one has that

$$\mathcal{E}(\rho) = \sum_m E_m \rho E_m^\dagger \tag{7}$$

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

$$\mathrm{tr}(N\mathcal{E}(\rho)) = \mathrm{tr}((\sum_m E_m^\dagger N E_m)\rho) \tag{8}$$

If we then take

$$M = \sum_m E_m^\dagger N E_m \tag{9}$$

in Eq.(8), we obtain

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

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

$$\forall \rho . \mathrm{tr}(M'\rho) \leq \mathrm{tr}(N\mathcal{E}(\rho)) \tag{11}$$

but because of Eq.(10) and Prop. 2.2, 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 $M = P$. □

From Eq.(9) and Prop. 3.2 we obtain the following.

Corollary 3.1. For all $\mathcal{E} \in \mathcal{CP}(\mathcal{H})$, $\mathrm{wp}(\mathcal{E}) \in \mathcal{PT}(\mathcal{H})$, i.e. it is a healthy predicate transformer.

3.4. *Duality*

In this section, we investigate the duality between the forward semantics of completely positive maps as state transformers, and the backwards semantics of healthy predicate transformers. This duality is part of a web of dualities known to mathematicians as Stone-type dualities (Johnstone, 1982), the prototype of which is the duality between boolean algebras and certain topological spaces called Stone spaces. For readers with a background in category theory we note that such a duality is captured by an adjoint equivalence mediated by a pairing, for example the satisfaction relation between states and predicates. Kozen - following suggestions of Plotkin - found such a duality in the context of probabilistic programs (Kozen, 1985). We show that such a duality exists in the quantum setting as well.

In the quantum context, we find the duality by defining an isomorphism between the set of all completely positive maps $\mathcal{CP}(\mathcal{H}_1, \mathcal{H}_2)$ and the set of all healthy predicate transformers $\mathcal{PT}(\mathcal{H}_2, \mathcal{H}_1)$. We can associate a healthy predicate transformer with every operation $\mathcal{E} \in \mathcal{CP}(\mathcal{H}_1, \mathcal{H}_2)$; this follows immediately from Prop. 3.3. Indeed, we associate with every operation $\mathcal{E}$ its weakest precondition predicate transformer $\mathrm{wp}(\mathcal{E})$. To complete the duality, we need to associate an operation $\mathcal{A} \in \mathcal{CP}(\mathcal{H}_1, \mathcal{H}_2)$ with a predicate transformer $\alpha \in \mathcal{PT}(\mathcal{H}_2, \mathcal{H}_1)$. Using the operator-sum representation for predicate transformers as given in Eq.(5), we have that

$$\begin{aligned} \mathrm{tr}(\alpha(M)\rho) &= \mathrm{tr}((\sum_u A_u^\dagger M A_u)\rho) \\ &= \mathrm{tr}(M.(\sum_u A_u \rho A_u^\dagger) \end{aligned} \tag{12}$$

If we then take

$$\mathcal{A}(\rho) = \sum_u A_u \rho A_u^\dagger \tag{13}$$

we obtain

$$\mathrm{tr}(\alpha(M)\rho) = \mathrm{tr}(M\mathcal{A}(\rho)) \tag{14}$$

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

To see this as a duality more clearly, we use the notation $\rho \models_r M$ defined in Sec. 3. Then we have

$$\frac{\mathcal{E}(\rho) \models_r M}{\rho \models_r \mathrm{wp}(\mathcal{E})M}. \tag{15}$$

It is straightforward to see that we have an order isomorphism between the domain of predicate transformers $\mathcal{PT}(\mathcal{H}_2, \mathcal{H}_1)$ and the domain of state transformers $\mathcal{CP}(\mathcal{H}_1, \mathcal{H}_2)$, and this for arbitrary Hilbert spaces $\mathcal{H}_1$ and $\mathcal{H}_2$. As an aside we note that because of this and the fact that maps in $\mathcal{PT}(\mathcal{H}_2, \mathcal{H}_1)$ are Scott-continuous, we immediately obtain that healthy predicate transformers are Scott-continuous as well.

4. Weakest precondition semantics for QPL

The quantum flow chart language or Quantum Programming Language (QPL), is a typed programming language for quantum computation with a formal semantics, which is built upon the idea of quantum data and classical control (Selinger, 2003). It is very different from previously defined quantum programming languages, which do not have a formal semantics and are imperative rather than functional. Syntactically, programs in QPL are represented either by flow charts or by QPL terms. The basic language constructs are allocating or discarding bits or qubits, assignment, branching, merge, measurement and unitary transformation. One can then build more complex programs from these atomic flow chart components through context extension, vertical and horizontal composition, iteration and recursion.

At each moment the denotation of the system, called a *state* in (Selinger, 2003), is given by a tuple of density matrices. The tuple dimension originates from classical bits present in the program, while tuple entries represent the state of all available qubits as density matrices. Each member of the tuple corresponds to a particular instantiation of the classical variables in lexicographical order; this is otherwise interpreted as a classical control path. Concretely, a state for a typing context containing n bits and m qubits is given by a 2^n-tuple $(\rho_0, \ldots, \rho_{2^n-1})$ of density matrices in $\mathcal{DM}(\mathbb{C}^{2^m})$. Program transformations are given by tuples of trace-decreasing completely positive maps which act on states – these are called superoperators in (Selinger, 2003). Note that positivity on tuples is defined such that it holds for each entry, while the trace of a tuple is defined as the sum of the traces of its entries.

The formal semantics of QPL is developed within the category $\mathbf{Q}$, which has signatures (which define tuples of complex finite-dimensional vector spaces) as its objects and superoperators as its morphisms. This category is equipped with a CPO-structure, composition, a coproduct $\oplus$ and a tensor product $\otimes$, all of which are Scott-continuous, and a monoidal trace $\mathbf{Tr}$. The latter is just the categorical trace for the co-pairing map $\oplus$; as per (Selinger, 2003) we use the term monoidal to avoid confusion with the categorical trace for the tensor product, i.e. the matrix trace tr. The coproduct $\oplus$ denotes concatenation of signatures. Note that, unlike the very similar situation of finite-dimensional vector spaces, it is *not* a product, as the diagonal map $\Delta : A \to A \oplus A$ does not respect matrix traces and hence is not a superoperator. All basic flow chart components are morphisms of this category. For example,the semantics of measurement of one qubit q is defined as

$$[\![\text{measure } q]\!] : \mathbf{qbit} \to \mathbf{qbit} \oplus \mathbf{qbit} : \rho \to (\mathcal{E}_0 \oplus \mathcal{E}_1)(\rho) = P_0\rho P_0 \oplus P_1\rho P_1 \ , \tag{16}$$

where $P_\psi = |\psi\rangle\langle\psi|$. Context extension is modeled by specific $\oplus$ or $\otimes$ operations on the state. Vertical and horizontal composition correspond to composition and coproducts of morphisms respectively, while iteration is interpreted via the monoidal trace. Specifically, suppose that an operation $\mathcal{E} : \sigma \oplus \tau \to \sigma' \oplus \tau$, where σ, 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 operation obtained from $\mathcal{E}$ by iterating over τ is then given by the *monoidal trace of* $\mathcal{E}$, defined as

$$\mathbf{Tr}(\mathcal{E}) = \mathcal{E}_{11} + \sum_{i=0}^{\infty} \mathcal{E}_{21}; \mathcal{E}_{22}^{i}; \mathcal{E}_{12} \ . \tag{17}$$

The existence of this limit is ensured by the CPO structure on superoperators (Selinger, 2003).

QPL also allows recursively defined operations $\mathcal{E} = F(\mathcal{E})$, where F is a flow chart. 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. Concretely,

$$\begin{aligned} \mathcal{E} &= \sqcup_i F_i \qquad \text{with } F_0 = 0 \text{ and } F_{i+1} = \Phi_F(F_i) & (18)\\ &= \sqcup_i \Phi_F^i(0) \ , & (19)\end{aligned}$$

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 CPO structure.

In what follows we derive a weakest precondition semantics for QPL. Note that in order to to this, our predicates need to operate on tuples of density matrices. We do this by writing expressions of the type $M_1 \oplus M_2$ where M_1 and M_2 are predicates in the sense of Def. 2.2. This works since $\oplus$ is in fact defined on arbitrary linear maps. We frequently write $\mathrm{wp}(\mathcal{Q})$ instead of $\mathrm{wp}([\![\mathcal{Q}]\!])$; by this we mean that we use the forward semantics of $\mathcal{Q}$, which is given by a tuple of completely positive maps, to derive the weakest precondition predicate transformer for $\mathcal{Q}$ according to the results in Sec. 3.3.

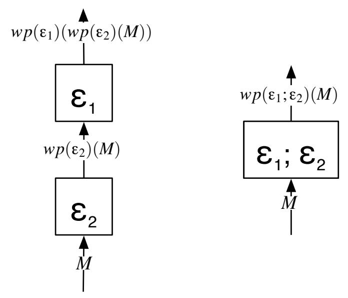
Basic flow charts. In our approach we uniformly consider all basic flow charts to be operations in the operator-sum representation as in Eq.(7). As such Prop. 3.3 already provides a weakest precondition semantics for these atomic flow charts. Note, however, that predicates need to be defined in accordance with the type of the tuple exiting a basic flow chart. As a concrete example, we mention measurement, for which the forward semantics is specified in Eq.(16). We find that for all predicates $M_1 \oplus M_2$ we have

$$\begin{aligned} \mathrm{wp}(\mathbf{measure}\ q)(M_1 \oplus M_2) &= \mathrm{wp}(\mathcal{E}_0 \oplus \mathcal{E}_1)(M_1 \oplus M_2)\\ &= \mathrm{wp}(\mathcal{E}_0)(M_1) + \mathrm{wp}(\mathcal{E}_1)(M_2) \qquad (20)\\ &= P_0 M_1 P_0 + P_1 M_2 P_1 \ . \end{aligned}$$

We now turn towards weakest precondition relations for composition techniques of QPL.

Sequential composition. Suppose we take the sequential composition of two operations $\mathcal{E}_1$ and $\mathcal{E}_2$, as shown in Fig. 1. For the composed operation $\mathcal{E}_1; \mathcal{E}_2$ and for all predicates M we have that

$$\mathrm{tr}(M.(\mathcal{E}_1; \mathcal{E}_2)(\rho)) = \mathrm{tr}(\mathrm{wp}(\mathcal{E}_1; \mathcal{E}_2)(M).\rho) \ . \tag{21}$$



$wp(\varepsilon_1)(wp(\varepsilon_2)(M))$
ε_1
$wp(\varepsilon_2)(M)$
ε_2
M
$wp(\varepsilon_1;\varepsilon_2)(M)$
$\varepsilon_1; \varepsilon_2$
M

Fig. 1. Sequential composition schematically.

If we calculate weakest preconditions for both operations separately and then compose them sequentially, we obtain

$$\begin{aligned} \mathrm{tr}(M.(\mathcal{E}_1;\mathcal{E}_2)(\rho)) &= \mathrm{tr}(M.\mathcal{E}_2(\mathcal{E}_1(\rho))) \\ &= \mathrm{tr}(\mathrm{wp}(\mathcal{E}_2)(M).\mathcal{E}_1(\rho)) \\ &= \mathrm{tr}(\mathrm{wp}(\mathcal{E}_1)(\mathrm{wp}(\mathcal{E}_2)(M)).\rho) \\ &= \mathrm{tr}((\mathrm{wp}(\mathcal{E}_2);\mathrm{wp}(\mathcal{E}_1))(M).\rho) \ . \end{aligned} \tag{22}$$

Hence by Eqs.(21) and (22) we obtain that weakest predicate transformers compose sequentially as follows,

$$\mathrm{wp}(\mathcal{E}_1;\mathcal{E}_2) = \mathrm{wp}(\mathcal{E}_2);\mathrm{wp}(\mathcal{E}_1) \ . \tag{23}$$

This is the same rule as one finds for sequential composition in classical programming languages (Dijkstra, 1976).

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

$$\mathrm{tr}((M_1 \oplus M_2).(\mathcal{E}_1 \oplus \mathcal{E}_2)(\rho_1 \oplus \rho_2)) = \mathrm{tr}(\mathrm{wp}(\mathcal{E}_1 \oplus \mathcal{E}_2)(M_1 \oplus M_2).(\rho_1 \oplus \rho_2)) \ . \tag{24}$$

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

$$\begin{aligned} \mathrm{tr}((M_1 \oplus M_2).(\mathcal{E}_1 \oplus \mathcal{E}_2)(\rho_1 \oplus \rho_2)) &= \mathrm{tr}(M_1.\mathcal{E}_1(\rho_1) \oplus M_2.\mathcal{E}_2(\rho_2)) \\ &= \mathrm{tr}(M_1.\mathcal{E}_1(\rho_1)) + \mathrm{tr}(M_2.\mathcal{E}_2(\rho_2)) \\ &= \mathrm{tr}(\mathrm{wp}(\mathcal{E}_1)(M_1).\rho_1) + \mathrm{tr}(\mathrm{wp}(\mathcal{E}_2)(M_2).\rho_2) \\ &= \mathrm{tr}((\mathrm{wp}(\mathcal{E}_1)(M_1) \oplus \mathrm{wp}(\mathcal{E}_2)(M_2)).(\rho_1 \oplus \rho_2)) \\ &= \mathrm{tr}((\mathrm{wp}(\mathcal{E}_1) \oplus \mathrm{wp}(\mathcal{E}_2))(M_1 \oplus M_2).(\rho_1 \oplus \rho_2)) \ . \end{aligned} \tag{25}$$

Comparing Eqs.(24) and (25) we obtain that for parallel composition weakest precondition predicate transformers compose as follows,

$$\mathrm{wp}(\mathcal{E}_1 \oplus \mathcal{E}_2) = \mathrm{wp}(\mathcal{E}_1) \oplus \mathrm{wp}(\mathcal{E}_2) \tag{26}$$

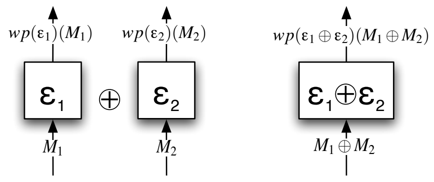


Fig. 2. Parallel composition schematically.

Context extension. Let us now study what occurs if we weaken a context with dummy classical or quantum variables. Suppose first that we have a QPL program $\mathcal{Q}$ with denotation $\mathcal{E}$. We first modify $\mathcal{Q}$ by picking a fresh classical variable b and adding it to $\mathcal{Q}$'s context; denote the resulting program $\mathcal{Q}_b$. The forward semantics of the latter is given by $\mathcal{E} \oplus \mathcal{E}$ (Selinger, 2003), and hence by Eq.(26) we find that

$$\mathrm{wp}(\mathcal{Q}_b) = \mathrm{wp}(\mathcal{Q}) \oplus \mathrm{wp}(\mathcal{Q}) \ . \tag{27}$$

Suppose next that we add a fresh qubit q to $\mathcal{Q}$'s context, and write $\mathcal{Q}_q$ for the resulting program. The forward semantics of $\mathcal{Q}_q$ is given by

$$[\![\mathcal{Q}_q]\!] \left(\begin{array}{c|c} \rho_1 & \rho_2 \\ \hline \rho_3 & \rho_4 \end{array} \right) = \left(\begin{array}{c|c} \mathcal{E}(\rho_1) & \mathcal{E}(\rho_2) \\ \hline \mathcal{E}(\rho_3) & \mathcal{E}(\rho_4) \end{array} \right) \ , \tag{28}$$

which we write more concisely as

$$[\![\mathcal{Q}_q]\!] = \left(\begin{array}{c|c} \mathcal{E} & \mathcal{E} \\ \hline \mathcal{E} & \mathcal{E} \end{array} \right) . \tag{29}$$

Accordingly, we find that

$$\mathrm{wp}(\mathcal{Q}_q) = \left(\begin{array}{c|c} \mathrm{wp}(\mathcal{E}) & \mathrm{wp}(\mathcal{E}) \\ \hline \mathrm{wp}(\mathcal{E}) & \mathrm{wp}(\mathcal{E}) \end{array} \right) . \tag{30}$$

Iteration. Consider a flow chart which is obtained from a program $\mathcal{Q}$ by introducing a loop. As explained in the above, the semantics of the flow chart is given by the monoidal trace $\mathbf{Tr}(\mathcal{E})$, where $\mathcal{E}$ is the semantics of the flow chart obtained from $\mathcal{Q}$ by removing the loop. For a predicate M we have that

$$\mathrm{tr}(M.(\mathbf{Tr}(\mathcal{E}))(\rho)) = \mathrm{tr}(\mathrm{wp}(\mathbf{Tr}(\mathcal{E}))(M).\rho) \tag{31}$$

By iterating explicitly and using Eqs.(17) and (23) we obtain

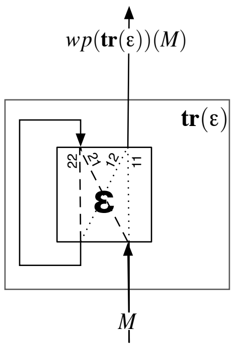


Fig. 3. Iteration schematically.

$$\begin{aligned}
&\mathrm{tr}(M.(\mathbf{Tr}(\mathcal{E}))(\rho)) \\
&= \mathrm{tr}(M.(\mathcal{E}_{11} + \sum_{i=0}^{\infty} \mathcal{E}_{21}; \mathcal{E}_{22}^{i}; \mathcal{E}_{12})(\rho)) \\
&= \mathrm{tr}(M.\mathcal{E}_{11}(\rho)) + \sum_{i=0}^{\infty} \mathrm{tr}(M.(\mathcal{E}_{21}; \mathcal{E}_{22}^{i}; \mathcal{E}_{12})(\rho)) \\
&= \mathrm{tr}(\mathrm{wp}(\mathcal{E}_{11})(M).\rho) + \sum_{i=0}^{\infty} \mathrm{tr}((\mathrm{wp}(\mathcal{E}_{12}); \mathrm{wp}(\mathcal{E}_{22})^{i}; \mathrm{wp}(\mathcal{E}_{21}))(M).\rho) \\
&= \mathrm{tr}((\mathrm{wp}(\mathcal{E}_{11}) + \sum_{i=0}^{\infty} \mathrm{wp}(\mathcal{E}_{12}); \mathrm{wp}(\mathcal{E}_{22})^{i}; \mathrm{wp}(\mathcal{E}_{21}))(M).\rho)\ .
\end{aligned} \tag{32}$$

Comparing Eqs.(31) and (32) we obtain that

$$\mathrm{wp}(\mathbf{Tr}(\mathcal{E})) = \mathrm{wp}(\mathcal{E}_{11}) + \sum_{i=0}^{\infty} \mathrm{wp}(\mathcal{E}_{12}); \mathrm{wp}(\mathcal{E}_{22})^{i}; \mathrm{wp}(\mathcal{E}_{21})\ . \tag{33}$$

Moreover, the existence of the limit in Eq.(33) is guaranteed due to Prop. 3.1.

Recursion. Consider an operation which is defined recursively, i.e. an operation $\mathcal{E}$ satisfying the equation $\mathcal{E} = F(\mathcal{E})$, where F is a flow chart. The required fixed point solution to this recursive equation is given by Eqs.(18) and (19). If we work out the weakest precondition relations using Eq.(18) and the fact that weakest precondition predicate transformers are Scott-continuous we obtain

$$\begin{aligned}
\mathrm{tr}(M.\mathcal{E}(\rho)) &= \mathrm{tr}(M.(\sqcup_i F_i)(\rho)) \\
&= \mathrm{tr}(\mathrm{wp}(\sqcup_i F_i)(M).\rho) \\
&= \mathrm{tr}((\sqcup_i \mathrm{wp}(F_i))(M).\rho)\ .
\end{aligned} \tag{34}$$

Combining this result with Prop. 3.3 we find that the weakest precondition predicate transformer for a recursively defined operation $\mathcal{E} = F(\mathcal{E})$ is obtained as

$$\mathrm{wp}(\mathcal{E}) = \sqcup_i \mathrm{wp}(F_i) = \sqcup_i \mathrm{wp}(\Phi_F^i(0))\ . \tag{35}$$

The existence of the least upper bound in Eq.(35) is guaranteed by Prop. 3.1. 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 operation $\mathcal{E}$, defined recursively as $\mathcal{E} = F(\mathcal{E})$.

5. Applications

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

5.1. *Grover's algorithm*

We first look into Grover's algorithm, also known as the database search algorithm (Grover, 1996). The algorithm is parameterized by the number of qubits n and is specified in QPL as follows, where we write N for 2^n.

$$\begin{aligned} \mathit{Grover}(N) ::= {} & \textbf{new qint}_\mathbf{n}\ q := \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \ ; \\ & \textbf{new int}\ k := C\ ; \\ & \textbf{while}\ \ k > 0\ \ \textbf{do}\ \{ \\ & \qquad q \mathrel{*}= G\ ; \\ & \qquad k := k - 1\ ; \\ & \} \\ & \textbf{measure}\ q \end{aligned} \tag{36}$$

Note that we assume the presence of product types of quantum integers $qint_n$ – qubit registers of size n – and integers int, which were elaborated in (Selinger, 2003), and also the presence of integer operations.

The Grover operator G is given by

$$G = O; IAM\ , \tag{37}$$

where O is a quantum oracle, which labels solutions to the search problem, and IAM is the *inversion about mean* operation, specifically $IAM = \frac{2}{N} \sum_{i,j=0}^{N-1} |i\rangle\langle j| - I$.

Supposing the solution to the search problem is given by s, then the relevant postcondition for Grover is given by $\bigoplus_{i=0}^{N-1} |s\rangle\langle s|$, in particular we wish to obtain

$$\mathrm{tr}(\bigoplus_{i=0}^{N-1} |s\rangle\langle s|\rho_{f_i}) = 1\ , \tag{38}$$

where $\bigoplus_{i=0}^{N-1} \rho_{f_i}$ is the final state of the algorithm, and the tuple summation is present due to measurement branching.

We work our way backwards through the algorithm using Eq.(23) in order to find the weakest precondition corresponding to the postcondition $\bigoplus_{i=0}^{N-1} |s\rangle\langle s|$. First we derive the weakest precondition for the measurement in the last step of the algorithm. We do this according to a generalization of Eq.(20) for N-valued measurements, as follows.

$$\begin{aligned}\mathrm{wp}(\mathbf{measure}\ q)(\bigoplus_{i=0}^{N-1}|s\rangle\langle s|) &= \mathrm{wp}(\mathcal{E}_0 \oplus \ldots \oplus \mathcal{E}_{N-1})(\bigoplus_{i=0}^{N-1}|s\rangle\langle s|) \\ &= \mathrm{wp}(\mathcal{E}_0)(|s\rangle\langle s|) + \cdots + \mathrm{wp}(\mathcal{E}_{N-1})(|s\rangle\langle s|) \\ &= P_0|s\rangle\langle s|P_0 + \cdots + P_{N-1}|s\rangle\langle s|P_{N-1} \\ &= |s\rangle\langle s|\ .\end{aligned} \tag{39}$$

Note that, since the remainder of the algorithm consists of unitary evolution, all relevant preconditions continue to be pure state projectors. In this case Eq.(38) holds only if the output state equals the predicate, that is if $\rho_f = |s\rangle\langle s|$, so that pure state preconditions are at the same time the states required for the algorithm to satisfy Eq.(38) after termination.

We now focus in the while loop in the algorithm. Geometrically, the Grover operator is a rotation in the two-dimensional space (Nielsen and Chuang, 2000, Sec. 6.1.3) spanned by the states $|s\rangle$ and

$$|\alpha\rangle = \frac{1}{\sqrt{N-1}}\sum_{x\neq s}|x\rangle \tag{40}$$

More specifically, G can be decomposed as

$$G = \begin{pmatrix}\cos\theta & -\sin\theta \\ \sin\theta & \cos\theta\end{pmatrix} \quad \text{with } \sin\theta = \frac{2\sqrt{N-1}}{N}\ . \tag{41}$$

Applying again Eq.(23), we obtain as weakest precondition with respect to the while loop the following,

$$\mathrm{wp}(\mathbf{while}\ \ k>0\ \ \mathbf{do}\ q \mathrel{*}= G)(|s\rangle\langle s|) = (G^k)|s\rangle\langle s|G^k\ , \tag{42}$$

where we omit explicit weakest precondition reasoning for the purely classical command $k := k-1$. Using Eq.(41), we see that $(G^C)^\dagger|s\rangle$ corresponds to C rotations over an angle of $-\theta$ in the state space spanned by $|\alpha\rangle$ and $|s\rangle$. By choosing $C = \arccos\frac{1}{\sqrt{N}}$ (Nielsen and Chuang, 2000, Sec 6.1.3), one rotates the postcondition $|s\rangle\langle s|$ towards the precondition $|\psi_i\rangle\langle\psi_i|$, where $|\psi_i\rangle$ is the initial state of the algorithm, i.e. the equal superposition state, which lies in the space spanned by the states $|\alpha\rangle$ and $|s\rangle$. In other words, using Eq.(6) and Eq.(38) we obtain that for all ρ_i

$$\begin{aligned}&\mathrm{tr}(\mathrm{wp}(\mathit{Grover})(|s\rangle\langle s|)\rho_i) = \mathrm{tr}(|s\rangle\langle s|\mathit{Grover}(\rho_i)) \\ \iff\ &\mathrm{tr}(|\psi_i\rangle\langle\psi_i|\rho_i) = 1\ .\end{aligned} \tag{43}$$

That is, Eq.(38) holds if and only if $\rho_i = |\psi_i\rangle\langle\psi_i|$ which is the case by construction of the algorithm. Hence we have established the correctness of the algorithm via our backwards semantics.

We note that an alternative derivation for Grover's algorithm based on probabilistic weakest preconditions has been reported in (Butler and Hartel, 1999). However, the use of probabilistic notions only works there because Grover's algorithm is considered for pure states only. The mathematical structures underlying their analysis is that of probabilistic weakest preconditions, which are in fact not suited at all for a generalized quantum setting, as we have stressed in Sec. 2. In our setting we could reason about mixed state solutions to Grover and compare them with the pure state solution elaborated in the above. Also, while it may seem at first sight that in (Butler and Hartel, 1999) the value of C is derived via the backward semantics this is in fact not the case. Instead, a recurrence relation for amplitudes occurring in each Grover iteration is solved; these amplitudes are found by applying the Grover iteration backwards, just as we did. We chose to adhere to the interpretation of G as a rotation in a two-dimensional state space in order to find C; we could just as well have adhered to the derivation in (Butler and Hartel, 1999). While their proof is an ingenious alternative to that in (Nielsen and Chuang, 2000), it is not based on the theory of probabilistic weakest preconditions.

5.2. *Tossing a coin*

As a second application, we derive the weakest precondition for the flow chart implementing a fair coin toss (Selinger, 2003, Example 4.1). In QPL terms the flow chart is specified as follows, where r is an input qubit register of unspecified length.

$$\begin{aligned} coin(r) ::= {} & \textbf{new qbit } q := \mathbf{0}; \\ & q \mathrel{*}= H; \\ & \textbf{measure } q; \\ & \textbf{discard } q \end{aligned} \tag{44}$$

An arbitrary postcondition for this program is of the form $M_1 \oplus M_2$, where M_1 and M_2 are both predicates over $\mathcal{P}(\mathbb{C}^{2^n})$ and n is the number of qubits in the register r. We derive the corresponding weakest precondition by flowing backwards through the program, starting with the discard operation. The latter induces the following quantum operation, where I_N is the $(N \times N)$ identity map with $N = 2^n$ as before, 0 denotes the $(N \times N)$ zero block matrix, and ρ is a density matrix in $\mathcal{DM}(\mathbb{C}^{2^{n+1}})$,

$$[\![\textbf{discard } q]\!](\rho) = \left(\begin{array}{c|c} I_N & 0 \end{array}\right) \rho \left(\begin{array}{c} I_N \\ \hline 0 \end{array}\right) + \left(\begin{array}{c|c} 0 & I_N \end{array}\right) \rho \left(\begin{array}{c} 0 \\ \hline I_N \end{array}\right) . \tag{45}$$

This leads to the following weakest precondition,

$$\mathrm{wp}(\textbf{discard } q)(M_1 \oplus M_2) = \left(\begin{array}{c|c} M_1 & 0 \\ \hline 0 & M_1 \end{array}\right) \oplus \left(\begin{array}{c|c} M_2 & 0 \\ \hline 0 & M_2 \end{array}\right) . \tag{46}$$

Next, we have the measurement step. We just give the result here, as this type of derivation was already encountered in the Grover example above.

$$\mathrm{wp}(\textbf{measure } q)[\left(\begin{array}{c|c} M_1 & 0 \\ \hline 0 & M_1 \end{array}\right) \oplus \left(\begin{array}{c|c} M_2 & 0 \\ \hline 0 & M_2 \end{array}\right)] = \left(\begin{array}{c|c} M_1 & 0 \\ \hline 0 & M_2 \end{array}\right) . \tag{47}$$

The Hadamard transformation is straightforward and leads to

$$\mathrm{wp}(q \mathrel{*}= H)\left(\begin{array}{c|c} M_1 & 0 \\ \hline 0 & M_2 \end{array}\right) = \frac{1}{2}\left(\begin{array}{c|c} M_1+M_2 & M_1+M_2 \\ \hline M_1+M_2 & M_1+M_2 \end{array}\right) . \tag{48}$$

Finally we move through the first command in the coin toss program, namely the addition of a new qubit. The forward semantics of this command is as follows, where ρ is a density matrix in $\mathcal{DM}(\mathbb{C}^{2^n})$,

$$[\![\mathbf{new\ qbit}\ q := \mathbf{0}]\!](\rho) = \left(\begin{array}{c} I_N \\ \hline 0 \end{array}\right) \rho \left(\begin{array}{c|c} I_N & 0 \end{array}\right) . \tag{49}$$

Hence, we obtain the following,

$$\mathrm{wp}(\mathbf{new\ qbit}\ q := \mathbf{0})(\frac{1}{2}\left(\begin{array}{c|c} M_1+M_2 & M_1+M_2 \\ \hline M_1+M_2 & M_1+M_2 \end{array}\right)) = \frac{1}{2}(M_1+M_2) . \tag{50}$$

Wrapping all individual steps of the coin toss program up into one weakest precondition predicate transformation according to Eq.(23) we obtain

$$\mathrm{wp}(\mathit{coin}(r))(M_1 \oplus M_2) = \frac{1}{2}(M_1+M_2) . \tag{51}$$

5.3. *Stabilizers are predicates*

The stabilizer formalism is an alternative description of quantum states (Gottesman, 1999). Instead of describing states as vectors in a suitable Hilbert space, they are described by a set of operators which leave the state invariant. Concretely, for an n-qubit system these operators are taken from the Pauli group G_n, i.e. the group of n-fold tensor products of the Pauli matrices with factors $\pm 1, \pm i$ in front. Note that if we allow all positive operators instead one obtains the more familiar density matrix formalism. Of course not all states can be described in this way. Formally, a *stabilizer state* is a simultaneous eigenvector of an abelian subgroup of the Pauli group with eigenvalue 1. This subgroup is then called the *stabilizer* S of this state, and usually represented by its generators. Surprisingly, some forms of entanglement, such as graph states for example (Raussendorf et al., 2003), as well as all Clifford group operations, can be described efficiently via stabilizers – a celebrated result known as the *Gottesman-Knill theorem* (Nielsen and Chuang, 2000, Sec. 10.5.4). This is because for an n-qubit stabilizer state its stabilizer S has $n-1$ generators (as opposed to 2^n amplitudes in the state formalism). A nice overview of stabilizer theory can be found in (Nielsen and Chuang, 2000, Ch. 10).

Stabilizers, which are unitaries, fit well within the setting of weakest preconditions, because when restricting ourselves to pure states, they *are* in fact quantum predicates. This follows from the following theorem.

Proposition 5.1. Given a pure state $\rho = |\psi\rangle\langle\psi|$ and a unitary U we have that

$$\mathrm{tr}(U\rho) = 1 \iff U|\psi\rangle = |\psi\rangle \tag{52}$$

Proof. For the left to right direction, we have that

$$\begin{aligned} &\mathrm{tr}(U|\psi\rangle\langle\psi|) = \langle\psi \mid U \mid \psi\rangle = 1 \\ &\Rightarrow (\langle\psi| - \langle\psi|U^\dagger)(|\psi\rangle - U|\psi\rangle) = 0 \\ &\Rightarrow |\psi\rangle - U|\psi\rangle = 0 \\ &\Rightarrow |\psi\rangle = U|\psi\rangle \ . \end{aligned} \tag{53}$$

The other direction is obvious. □

For example, consider the creation of a Bell state $|B\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$ by applying $U = CNOT.(H \otimes I)$ to $|00\rangle$. The stabilizer of $|B\rangle$ is generated by Z_1Z_2 and X_1X_2. Hence by the above result we have $\mathrm{tr}(Z_1Z_2\mathcal{E}_U(|\psi\rangle\langle\psi|)) = \mathrm{tr}(X_1X_2\mathcal{E}_U(|\psi\rangle\langle\psi|)) = 1$, where $|\psi\rangle$ is the initial state of the algorithm and $\mathcal{E}_U(\rho) = U\rho U^\dagger$ for all ρ. Applying Eq.(9), we obtain as weakest preconditions $\mathrm{wp}(\mathcal{E}_U)(Z_1Z_2) = Z_2$ and $\mathrm{wp}(\mathcal{E}_U)(X_1X_2) = Z_1$. By Prop. 3.3 we thus also have $\mathrm{tr}(Z_1|\psi\rangle\langle\psi|) = \mathrm{tr}(Z_2|\psi\rangle\langle\psi|) = 1$. But then by the above result Z_1 and Z_2 are stabilizers of $|\psi\rangle$. Hence $|\psi\rangle = |00\rangle$, as required.

6. Conclusions

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 (Selinger, 2003). 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.

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. It is likely that in the field of communication protocols, such as those based on teleportation, we have a good stock of ideas and examples which could be used as the basis of methodologies in this context.

The most closely related work - apart from Selinger's work on his programming language - is the work by Sanders and Zuliani (Sanders and Zuliani, 2000) 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 (Baltag and Smets, 2004) - 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 (Edalat, 2004) 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 uncertainty 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 (Morgan and McIver, 2004) gives an expository account of their work. We emphasize again that the theory of probabilistic predicate transformers does not capture the proper notions appropriate for the quantum setting. Linearity and complete positivity are essential aspects of the theory of quantum predicate transformers. If one tries to work with probabilistic predicates alone one will not be able to express healthiness conditions that capture the physically allowable transformations, as the example presented in Sec. 3 illustrates.

One might worry that the predicates are too restricted. There are many "observables" in physics that are not positive; for example, the z-component of angular momentum, written J_z, for a spin $\frac{1}{2}$ system takes on the values $\pm\frac{1}{2}$. However, for reasoning about the evolution of J_z one can work instead with the operator $\frac{1}{2}[I + J_z]$ which has eigenvalues $\frac{1}{4}$ and $\frac{3}{4}$ and so is a predicate. Of course one cannot do this for unbounded operators like the energy, but this will not be a handicap for quantum computation.

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, for example the one-way model (Raussendorf and Briegel, 2001; Raussendorf et al., 2003) for which language ideas are just emerging (Danos et al., 2004).

Acknowledgements

It is a pleasure to thank Samson Abramsky, Bob Coecke, Elham Kashefi and Peter Selinger for helpful discussions. Comments by the referees were very helpful.

References

Baltag, A. and Smets, S. (2004). Quantum dynamic logic. In Selinger, P., editor, *Proceedings of the 2nd Workshop on Quantum Programming Languages (QPL04)*, Turku, Finland. Turku Centre for Computer Science, TUCS General Publication No 33.

Butler, M. J. and Hartel, P. H. (1999). Reasoning about Grover's quantum search algorithm using probabilistic wp. *ACM transactions on programming languages and systems*, 21(3):417–430.

Danos, V., Kashefi, E., and Panangaden, P. (2004). The measurement calculus. quant-ph/0412135.

Deutsch, D. (1985). Quantum theory, the Church-Turing principle and the universal quantum computer. *Proc. R. Soc. Lond.*, pages A400: 97–117.

Deutsch, D. and Jozsa, R. (1992). Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond. A*, pages 439–553.

Dijkstra, E. W. (1976). *A Discipline of Programming.* Prentice-Hall.

Edalat, A. (2004). An extension of Gleason's theorem for quantum computation. *Int. J. of Theor. Phys.* (to appear).

Gottesman, D. (1999). The Heisenberg representation of quantum computers. In Corney, S. P., Delbourgo, R., and Jarvis, P. D., editors, *Proceedings of the XXII International Colloquium on Group Theoretical Methods in Physics*, pages 32–43. International Press.

Gries, D. (1981). *The Science of Programming.* Springer-Verlag.

Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In *ACM Symposium on Theory of Computing*, pages 212–219.

Hoare, C. (1969). An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580.

Johnstone, P. (1982). *Stone Spaces*, volume 3 of *Cambridge Studies in Advanced Mathematics.* Cambridge University Press.

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

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

Löwner, K. (1934). Uber monotone matrixfunktionen. *Mathematische Zeitschrift*, 38:177–216.

Morgan, C. and McIver, A. (2004). *Abstraction, refinement and proof for probabilistic systems.* Springer-Verlag.

Nielsen, M. A. and Chuang, I. (2000). *Quantum computation and quantum information.* Cambridge university press.

Peres, A. (1995). *Quantum Theory: Concepts and Methods.* Kluwer Academic Publishers.

Plotkin, G. D. (1983). Lecture notes on domain theory.

Raussendorf, R. and Briegel, H. J. (2001). A one-way quantum computer. *Phys. Rev. Lett.*, 86(22):5188–5191.

Raussendorf, R., Browne, D. E., and Briegel, H. J. (2003). Measurement-based quantum computation on cluster states. *Phys. Rev. A*, 68(2):022312.

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

Selinger, P. (2003). Towards a quantum programming language. *Math. Struct. in Comp. Science.*

Shaji, A. and Sudarshan, E. (2005). Who's afraid of not completely positive maps? *Phys. Lett. A*, 341:48–54.

Shor, P. W. (1994). Algorithms for quantum computation: Discrete logarithms and factoring. In *IEEE Symposium on Foundations of Computer Science*, pages 124–134.

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