arXiv:1810.11334v1 [cs.LO] 26 Oct 2018

Reasoning about Parallel Quantum Programs

Mingsheng Ying and Yangjia Li

Centre for Quantum Software and Information, University of Technology Sydney, Australia
State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China
Department of Computer Science and Technology, Tsinghua University, China
Institute of ompute Science, University of Tartu, Estonia
`Mingsheng.Ying@uts.edu.au`

Abstract. We initiate the study of parallel quantum programming by defining the operational and denotational semantics of parallel quantum programs. The *technical contributions* of this paper include: (1) find a series of useful proof rules for reasoning about correctness of parallel quantum programs; and (2) prove a strong soundness theorem of these proof rules showing that partial correctness is well maintained at each step of transitions in the operational semantics of a parallel quantum program. This is achieved by *partially* overcoming the following *conceptual challenges* that are never present in classical parallel programming: (i) the intertwining of nondeterminism caused by quantum measurements and introduced by parallelism; (ii) entanglement between component quantum programs; and (iii) combining quantum predicates in the overlap of state Hilbert spaces of component quantum programs with shared variables. It seems that a *full* solution to these challenges and developing a (relatively) complete proof system for parallel quantum programs are still far beyond the current reach.

Keywords: Quantum programming, Parallel programs, Entanglement, Interference, Operational semantics, Denotational semantics, Partial correctness, Total correctness.

1 Introduction

Quantum programming research started from several high-level quantum programming languages proposed as early as in the later 1990's and early 2000's: QCL by Ömer [32], qGCL by Sanders and Zuliani [38], QPL by Selinger [39] and QML by Altenkirch and Grattage [3]. Now it has been extensively conducted for two decades; see [40], [15], [49] for a survey. In particular, some more practical and scalable quantum programming languages have been defined and implemented in the last few years, including Quipper [18], Scaffold [1], QWIRE [36], and Microsoft's LIQUi$|\rangle$ [47] and Q# [45]. Various semantics and type theories of quantum programming languages have been extensively studied; for example, a denotational semantics of quantum lambda calculus with recursion was discovered by Hasuo and Hoshino [21] and Pagani et al. [35], an algebraic theory for equational reasoning about quantum programs was developed by Staton [43], and type systems have been established for quantum lambda-calculus [41] and QWIRE [36].

Quantum Hoare Logic: Several verification techniques for classical programs have also been extended to quantum programs [5], [7], [8], [13], [16], [24], [37]. In particular, the notion of weakest precondition for quantum programs was introduced by D'Hondt and Panangaden in [12], and then a Hoare-like logic for both partial and total correctness of quantum programs with (relative) completeness was built in [48]. Recently, a theorem prover for quantum Hoare logic was implemented based on Isabelle/HOL in [29], and an SDP (Semi-Definite Programming) algorithm for generating invariants and an SDP algorithm for termination analysis of quantum programs with ranking functions (or super-martingales) were presented in [51], [28].

Why Parallel Quantum Programming? The works mentioned above concentrate on sequential quantum programming. However, parallel programming problem for quantum computing has already arisen in the following three areas:

- Several models of parallel and distributed quantum computing were proposed more than fifteen years ago, mainly with the following two motivations: (i) use the physical resources of two or more small-capacity quantum computers to realise large-capacity quantum computing, which is out of the reach of current technology; and (ii) find quantum algorithms for solving paradigmatic parallel and distributed computing problems, faster than the known classical algorithms. For example, several parallel implementations of the quantum Fourier transform and Shor's quantum factoring algorithm were presented in [9], [30]; a formal language for defining quantum circuits in distributed quantum computing was introduced in [50]; a quantum algorithm for the leader election problem was given in [46] and a quantum protocol for the dinning philosopher problem was shown in [2].
- Experiments of the physical implementation of parallel and distributed quantum computing have been frequently reported in the recent years. In particular, the issue of instruction parallelism has already been considered in Rigetti's quantum instruction set architecture [42] and IBM Q [10].
- Motivated by the tremendous progress toward practical quantum hardware in the recent years, some authors [11] started to consider how to design an operating system for quantum computers; in particular, what new abstractions could a quantum operating system expose to the programmer? It is well-known that parallelism is a major issue in operating systems for classical computers [23]. As one can imagine, it will also be a major issue in the design and implementation of future quantum operating systems.

Aims of the Paper: This paper initiates the study of parallel quantum programming by introducing a programming language that can be used to program parallel and distributed quantum algorithms like those mentioned above. This language is the quantum **while**-language [48], [49] expanded with the construct of parallel composition. We define the operational and denotational semantics of parallel composition of quantum programs. The emphasis of this paper is to establish a series of useful proof rules for reasoning about correctness of parallel quantum programs. Several simple examples are given along the way to show the subtle difference between the classical and quantum cases. A strong soundness theorem is proved for them showing that partial correctness is well maintained at each step of the transitions in the operational semantics of a parallel quantum program, which can be seen as a quantum generalisation of Lemma 8.8 in [4] or the strong soundness theorem in Section 7.4 of [14]. We expect that the results obtained in this paper can also be used to model and reason about parallelism in quantum operating systems.

For simplicity of the presentation, in this paper we focus on partial correctness of parallel quantum programs. The techniques for reasoning about partial correctness developed in this paper can be extended to deal with total correctness of parallel quantum programs by adding ranking functions (or super-martingales) to guarantee termination.

Owicki-Gries and Lamport Method: The proof system introduced by Owicki and Gries [34] and Lamport [27] is one of the most popular methods for reasoning about classical parallel programs. Roughly speaking, it consists of the Hoare logic for sequential programs, a rule for introducing auxiliary variables recording control flows and a key rule (R.PC) for parallel composition shown in Figure 1.

$$\text{(R.PC)}\qquad \frac{\text{Proofs of }\ \{A_i\}\, P_i\, \{B_i\}\ \ (i=1,...,n) \text{ are interference free}}{\left\{\bigwedge_{i=1}^{n} A_i\right\} P_1\| \cdots \|P_n \left\{\bigwedge_{i=1}^{n} B_i\right\}}$$

Fig. 1. Proof Rule for Parallel Composition.

Naturally, a starting point for our research on reasoning about parallel quantum programs is to generalise the Owicki-Gries and Lamport method to the quantum setting. However, it is highly nontrivial to develop such a quantum generalisation, especially to find an *appropriate* quantum version of inference rule (R.PC) for parallel composition of programs, and the unique features of quantum systems render us with several challenges in parallel quantum programming that would never be present in parallel programming for classical computers.

Major Challenges in Parallel Quantum Programming:

– *Intertwined nondeterminism*: In a quantum **while**-program, nondeterminism is caused only by the involved quantum measurements, and in a classical parallel program, nondeterminism is introduced only by the parallelism. However, in a parallel quantum program, these two kinds of nondeterminism occur simultaneously, and their intertwining is hard to deal with in defining the denotational semantics of the program; in particular, when it contains loops which can have infinite computations. This problem will be settled in Section 4 by establishing a subtle confluence property for different execution paths of the parallel quantum program (see Lemmas 41 and 42 and their proofs).

– *Entanglement*: The denotational semantics achieved by solving the above challenge provides us with a basis for building an Owicki-Gries and Lamport-like proof system for parallel quantum programs. At the first glance, it seems that disjoint parallel quantum programs are easy to deal with because (i) interference freedom is automatically there, as what happens in classical disjoint parallel programs; and (ii) conjunctives $\bigwedge_{i=1}^{n} A_i$ and $\bigwedge_{i=1}^{n} B_i$ in rule (R.PC) have proper quantum counterparts, namely tensor products $\bigotimes_{i=1}^{n} A_i$ and $\bigotimes_{i=1}^{n} B_i$, respectively, when $P_1, ..., P_n$ are disjoint. But actually a difficulty that makes no sense in classical computing arises in reasoning about parallel quantum programs even in this simple case. More explicitly, entanglement is indispensable for realising the advantage of quantum computing over classical computing, but a quantum generalisation of (R.PC) is not strong enough to cope with the situation where entanglement between component programs is present. We propose two techniques to overcome this difficulty: (a) introducing an additional inference rule obtained by invoking a deep theorem about the relation between noise and entanglement from quantum physics [19] (see rule (R.S2E) in Figure 12); and (b) introducing auxiliary variables (see Subsection 4.2) based on the observation in physics that entanglement may emerge when reducing a state of a composite system to its subsystems [31].

– *Combining predicates in the overlap of state Hilbert spaces*: When we further consider parallel quantum programs with shared variables, another difficulty appears which never happens in classical computation: the Hilbert spaces $\mathcal{H}_{P_i}$ $(i = 1, ..., n)$ of quantum predicates A_i, B_i $(i = 1, ..., n)$ have overlaps. Then conjunctives $\bigwedge_{i=1}^{n} A_i$ and $\bigwedge_{i=1}^{n} B_i$ cannot be simply replaced by tensor products $\bigotimes_{i=1}^{n} A_i$ and $\bigotimes_{i=1}^{n} B_i$, respectively, because they are not well-defined in the state Hilbert space $\bigotimes_{i=1}^{n} \mathcal{H}_{P_i}$ of $P_1 \| \cdots \| P_n$. Up to now, we only have a partial solution to this difficulty. The idea is that probabilistic (convex) combinations of A_i $(i = 1, ..., n)$ and B_i $(i = 1, ..., n)$ can serve as a kind of approximations to the quantum counterparts of conjunctives $\bigwedge_{i=1}^{n} A_i, \bigwedge_{i=1}^{n} B_i$, respectively. Although a probabilistic combination is not a perfect quantum version of conjunctive, as a tensor product did in the case of disjoint parallel quantum programs, its reasonableness and usefulness can be clearly seen through its connection to local Hamiltonians in many-body quantum systems (see a detailed discussion in Remark 52). Furthermore, we can define a notion of parametrised interference freedom between the proof outlines of component quantum programs. Then a quantum variant of inference rule (R.PC) can be introduced to reason about parallel quantum programs with shared variables.

Organisation of the Paper: For convenience of the reader, we briefly review quantum Hoare logic in Section 2. The notion of proof outline is required to present inference rule (R.PC) for classical parallel programs with shared variables. A corresponding notion is needed to present the quantum generalisation(s) of rule (R.PC). Such a notion is introduced for quantum **while**-programs in Section 3. Our study of parallel quantum programming starts in Section 4 where we define the operational and denotational semantics of disjoint parallel quantum programs and present a quantum generalisation of rule (R.PC) for reasoning about them. In particular, we introduce an additional rule that enables us to deal with entanglement between component quantum programs in Subsection 4.2. Section 5 is devoted to examine parallel quantum programs with shared variables, where we introduce the notion of parameterised noninterference and present an inference rule for a parallel quantum program with its precondition (resp. postcondition) as a probabilistic combination of the preconditions (resp. postconditions) of its component programs. Section 6 is the concluding section where several unsolved problems are pointed out and their difficulties are briefly discussed.

2 Hoare Logic for Quantum While-Programs

The parallel quantum programs considered in this paper are parallel compositions of quantum **while**-programs studied in [48], [49]. In this section, we briefly review the syntax and semantics of quantum **while**-language and quantum Hoare logic from [48], [49]. They will serve as a basis of the subsequent sections.

2.1 Syntax and Semantics of Quantum while-Programs

We assume a countably infinite set Var of quantum variables. For each $q \in Var$, we write $\mathcal{H}_q$ for its state Hilbert space. For any $X \subseteq Var$, we put $\mathcal{H}_X = \bigotimes_{q\in X} \mathcal{H}_q$.

Definition 21 (Syntax [48], [49]) *The quantum **while**-programs are defined by the grammar:*

$$\begin{aligned} P ::= \ & \mathbf{skip} \mid P_1; P_2 \mid q := |0\rangle \mid \overline{q} := U[\overline{q}] && (1)\\ & \mid \mathbf{if}\ \ (\square m \cdot M[\overline{q}] = m \to P_m)\ \ \mathbf{fi} && (2)\\ & \mid \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ P\ \mathbf{od} && (3)\end{aligned}$$

Here, $q := |0\rangle$ means that quantum variable q is initialised in a basis state $|0\rangle$. $\overline{q} := U[\overline{q}]$ denotes that unitary transformation U is applied to quantum register $\overline{q}$, which is a sequence of quantum variables. In the case statement $\mathbf{if} \cdots \mathbf{fi}$, quantum measurement M is performed on the register $\overline{q}$ and then a subprogram P_m is selected for next execution according to the measurement outcome m. In the loop $\mathbf{while} \cdots \mathbf{end}$, measurement M in the loop guard has only two possible outcomes $0, 1$; if the outcome is 0 the loop terminates, and if the outcome is 1 the program executes the loop body P and enters the loop again.

For each quantum program P, we write $var(P)$ for the set of quantum variables occurring in P. Let $\mathcal{H}_P = \mathcal{H}_{var(P)}$ be the state Hilbert space of P. We write $\mathcal{D}(\mathcal{H}_P)$ for the set of partial density operators, i.e. positive operators with traces ≤ 1, in $\mathcal{H}_P$. A configuration is a pair $C = \langle P, \rho\rangle$, where P is a program or the termination symbol $\downarrow$, and $\rho \in \mathcal{D}(\mathcal{H}_P)$ denotes the state of quantum variables.

Definition 22 (Operational Semantics [48], [49]) *The operational semantics of quantum* ***while****-programs is defined as a transition relation $\rightarrow$ by the transition rules in Figure 2.*

$$\text{(Sk)}\quad \langle \mathbf{skip}, \rho\rangle \rightarrow \langle \downarrow, \rho\rangle \qquad\qquad \text{(In)}\quad \langle q := |0\rangle, \rho\rangle \rightarrow \langle \downarrow, \rho_0^q\rangle$$

$$\text{(UT)}\quad \langle \overline{q} := U[\overline{q}], \rho\rangle \rightarrow \langle \downarrow, U\rho U^\dagger\rangle \qquad\qquad \text{(SC)}\quad \frac{\langle P_1, \rho\rangle \rightarrow \langle P_1', \rho'\rangle}{\langle P_1; P_2, \rho\rangle \rightarrow \langle P_1'; P_2, \rho'\rangle}$$

$$\text{(IF)}\quad \langle \mathbf{if}\ (\square m \cdot M[\overline{q}] = m \rightarrow P_m)\ \mathbf{fi}, \rho\rangle \rightarrow \langle P_m, M_m\rho M_m^\dagger\rangle$$

$$\text{(L0)}\quad \langle \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ P\ \mathbf{od}, \rho\rangle \rightarrow \langle \downarrow, M_0\rho M_0^\dagger\rangle$$

$$\text{(L1)}\quad \langle \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ P\ \mathbf{od}, \rho\rangle \rightarrow \langle P; \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ P\ \mathbf{od}, M_1\rho M_1^\dagger\rangle$$

Fig. 2. Transition Rules for Quantum **while**-Programs. In (In), $\rho_0^q = |0\rangle_q\langle 0|\rho|0\rangle_q\langle 0| + |0\rangle_q\langle 1|\rho|1\rangle_q\langle 0|$ if $type(q) = \mathbf{Bool}$ and $\rho_0^q = \sum_{n=-\infty}^{\infty} |0\rangle_q\langle n|\rho|n\rangle_q\langle 0|$ if $type(q) = \mathbf{Int}$. In (SC), we make the convention $\downarrow; P_2 = P_2$. In (IF), m ranges over every possible outcome of measurement $M = \{M_m\}$.

Note that the transitions in rules (IF), (L0) and (L1) are essentially probabilistic; for example, for each m, the transition in (IF) happens with probability $p_m = tr(M^\dagger M_m \rho)$, and the program state ρ is changed to $\rho_m = M_m\rho M_m^\dagger / p_m$. But we choose to combine probability p_m and density operator ρ_m into a partial density operator $M_m\rho M_m^\dagger = p_m\rho_m$. This convention allows us to present the operational semantics as a non-probabilistic transition system, and it further works for the composition of a sequence of transitions because all transformations in quantum mechanics are linear. Thus, it significantly simplifies the presentation.

Definition 23 (Denotational Semantics [48], [49]) *For any quantum* ***while****-program P, its semantic function is the mapping $[\![P]\!] : \mathcal{D}(\mathcal{H}_P) \rightarrow \mathcal{D}(\mathcal{H}_P)$ defined by*

$$[\![P]\!](\rho) = \sum \{|\rho' : \langle P, \rho\rangle \rightarrow^* \langle \downarrow, \rho'\rangle|\} \tag{4}$$

for every $\rho \in \mathcal{D}(\mathcal{H}_P)$, where $\rightarrow^*$ is the reflexive and transitive closure of $\rightarrow$, and $\{|\cdot|\}$ denotes a multi-set.

Intuitively, for an input ρ, if for each $k \geq 0$, program P terminates at step k with probability q_k and outputs density operator σ_k, then with the explanation given in the paragraph before the above definition in mind it is easy to see that $[\![P]\!](\rho) = \sum_{k=0}^{\infty} q_k\sigma_k$.

2.2 Correctness

First-order logical formulas are used as the assertions about the properties of classical program states. The properties of quantum program states are described by quantum predicates introduced by D'Hondt and Panangaden in [12]. The Löwner order between operators is defined as follows: $A \sqsubseteq B$ if and only if $B - A$ is positive. Then a quantum predicate in a Hilbert space $\mathcal{H}$ is an observable (a Hermitian operator) A in $\mathcal{H}$ with $0 \sqsubseteq A \sqsubseteq I$, where 0 and I are the zero operator and the identity operator in $\mathcal{H}$, respectively.

Definition 24 (Correctness Formula, Hoare Triple [48], [49]) *A correctness formula (or a Hoare triple) is a statement of the form $\{A\}P\{B\}$, where P is a quantum **while**-program, and both A, B are quantum predicates in $\mathcal{H}_P$, called the precondition and postcondition, respectively.*

Definition 25 (Partial and Total Correctness [48], [49])
1. *The correctness formula $\{A\}P\{B\}$ is true in the sense of total correctness, written $\models_{tot} \{A\}P\{B\}$, if for all $\rho \in \mathcal{D}(\mathcal{H}_P)$ we have: $tr(A\rho) \leq tr(B[\![P]\!](\rho))$.*
2. *The correctness formula $\{A\}P\{B\}$ is true in the sense of partial correctness, written $\models_{par} \{A\}P\{B\}$, if for all $\rho \in \mathcal{D}(\mathcal{H}_P)$ we have: $tr(A\rho) \leq tr(B[\![P]\!](\rho)) + [tr(\rho) - tr([\![P]\!](\rho))]$.*

The defining inequalities of total and partial correctness can be easily understood by noting that the interpretation of $tr(A\rho)$ in physics is the expectation (i.e. average value) of observable A in state ρ, and $tr(\rho) - tr([\![P]\!](\rho))$ is indeed the probability that with input ρ program P does not terminate.

2.3 Proof System

A Hoare-like logic for quantum **while**-programs was established in [48], [49]. It includes a proof system qPD for partial correctness and a system qTD for total correctness. The axioms and inference rules of qPD are presented in Figure 3. Similar to the classical case, qTD is obtained from qPD by adding a ranking function into rule (R.LP) to guarantee termination.

The soundness and (relative) completeness of both qPD and qTD were proved in [48], [49].

Theorem 21 (Soundness and Completeness [48], [49]) *For any quantum **while**-program P, and for any quantum predicates A, B,*

$$\models_{par} \{A\}P\{B\} \Leftrightarrow \vdash_{qPD} \{A\}P\{B\}, \qquad \models_{tot} \{A\}P\{B\} \Leftrightarrow \vdash_{qTD} \{A\}P\{B\}.$$

3 Proof Outlines

The notion of proof outline was introduced in classical programming theory so that the proofs of programs can be organised in a structured way. On the other hand, a crucial premise in inference rule (R.PC) for a parallel program with

(Ax.Sk) $\{A\}\mathbf{Skip}\{A\}$ (Ax.In.B) $\{|0\rangle_q\langle 0|A|0\rangle_q\langle 0| + |1\rangle_q\langle 0|A|0\rangle_q\langle 1|\}q := |0\rangle\{A\}$

(Ax.In.I) $\left\{\sum_{n=-\infty}^{\infty} |n\rangle_q\langle 0|A|0\rangle_q\langle n|\right\} q := |0\rangle\{A\}$

(Ax.UT) $\{U^\dagger AU\}\overline{q} := U\,[\overline{q}]\,\{A\}$ (R.SC) $\dfrac{\{A\}P_1\{B\} \quad \{B\}P_2\{C\}}{\{A\}P_1;P_2\{C\}}$

(R.IF) $\dfrac{\{A_m\}P_m\{B\} \text{ for all } m}{\left\{\sum_m M_m^\dagger A_m M_m\right\} \mathbf{if}\ (\square m \cdot M[\overline{q}] = m \rightarrow P_m)\ \mathbf{fi}\{B\}}$

(R.LP) $\dfrac{\{B\}P\left\{M_0^\dagger AM_0 + M_1^\dagger BM_1\right\}}{\{M_0^\dagger AM_0 + M_1^\dagger BM_1\}\mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ P\ \mathbf{od}\{A\}}$

(R.Or) $\dfrac{A \sqsubseteq A' \quad \{A'\}P\{B'\} \quad B' \sqsubseteq B}{\{A\}P\{B\}}$

Fig. 3. Proof System qPD for Quantum **while**-Programs. In (Ax.In.B), *type*(q) = **Boolean**. In (Ax.In.I), *type*(q) = **Int**.

shared variables is interference freedom between its component programs, which is usually defined in terms of proof outlines of the component programs. So in this section, we generalise the notion of proof outline to quantum **while**-programs so that it can be used in Section 5 to present our inference rules for parallel quantum programs with shared variables.

Definition 31 *Let P be a quantum **while**-program. A proof outline for partial correctness of P is a formula $\{A\}P^*\{B\}$ formed by the formation axioms and rules in Figure 4, where P^* results from P by interspersing quantum predicates.*

The notion of proof outline for total correctness of quantum **while**-programs can be defined in a similar way; but we omit it here because in the rest of this paper we only consider partial correctness of parallel quantum programs.

We will mainly use a special form of proof outlines defined in the following:

Definition 32 *A proof outline $\{A\}P^*\{B\}$ of P is called standard if every subprogram Q of P is proceeded by exactly one quantum predicate, denoted pre(Q), in P^*.*

The following proposition shows that standard proof outline are general enough for our purpose.

Proposition 31 *1. If $\{A\}P^*\{B\}$ is a proof outline for partial correctness, then $\vdash_{qPD} \{A\}P\{B\}$.*
2. If $\vdash_{qPD} \{A\}P\{B\}$, then there is a standard proof outline $\{A\}P^*\{B\}$ for partial correctness.

(Ax.Sk$'$) $\{A\}\mathbf{Skip}\{A\}$ (Ax.In.B$'$) $\{|0\rangle_q\langle 0|A|0\rangle_q\langle 0| + |1\rangle_q\langle 0|A|0\rangle_q\langle 1|\}q := |0\rangle\{A\}$

(Ax.In.I$'$) $\left\{\sum_{n=-\infty}^{\infty} |n\rangle_q\langle 0|A|0\rangle_q\langle n|\right\} q := |0\rangle\{A\}$

(Ax.UT$'$) $\{U^\dagger AU\}\overline{q} := U\,[\overline{q}]\,\{A\}$ (R.SC$'$) $\dfrac{\{A\}P_1^*\{B\} \quad \{B\}P_2^*\{C\}}{\{A\}P_1^*;\{B\}P_2^*\{C\}}$

(R.IF$'$)
$$\frac{\{A_{m_i}\}P^*_{m_i}\{B\}\ (i=1,...,k)}{\begin{array}{l}\left\{\sum_i^k M^\dagger_{m_i}A_{m_i}M_{m_i}\right\}\ \mathbf{if}\ M[\overline{q}] = m_1 \to \{A_{m_1}\}\,P^*_{m_1}\\ \qquad\qquad\qquad\qquad ....\\ \qquad\qquad\qquad\quad \square\ M[\overline{q}] = m_k \to \{A_{m_k}\}\,P^*_{m_k}\\ \qquad\qquad \mathbf{fi}\ \{B\}\end{array}}$$

(R.LP$'$)
$$\frac{\{B\}P^*\left\{M_0^\dagger AM_0 + M_1^\dagger BM_1\right\}}{\{M_0^\dagger AM_0 + M_1^\dagger BM_1\}\ \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ \{B\}\ P^*\ \left\{M_0^\dagger AM_0 + M_1^\dagger BM_1\right\}\ \mathbf{od}\ \{A\}}$$

(R.Or$'$) $\dfrac{A \sqsubseteq A' \quad \{A'\}P^*\{B'\} \quad B' \sqsubseteq B}{\{A\}\{A'\}P\{B'\}\{B\}}$ (R.Del) $\dfrac{\{A\}P^*\{B\}}{\{A\}P^{**}\{B\}}$

Fig. 4. Formation Axioms and Rules for Partial Correctness of Quantum **while**-Programs. In (R.IF'), $\{m_1, ..., m_k\}$ is the set of all possible outcomes of measurement M. In (R.Del), P^{**} is obtained by deleting some quantum predicates from P^*.

Proof. Induction on the lengths of proof and formation.

The notion of proof outline enables us to present a strong soundness of quantum Hoare logic indicating that soundness is well maintained in each step of the proofs of quantum **while**-programs. To this end, we need an auxiliary notation defined in the following:

Definition 33 *Let P be a quantum **while**-program and T a subprogram of P. Then $at(T,P)$ is inductively defined as follows:*

1. *If $T \equiv P$, then $at(T,P) \equiv P$;*
2. *If $P \equiv P_1; P_2$, then $at(T,P) \equiv at(T,P_1); P_2$ when T is a subprogram of P_1, and $at(T,P) \equiv at(T,P_2)$ when T is a subprogram of P_2;*
3. *If $P \equiv$ **if** $(\square m \cdot M[\overline{q}] = m \to P_m)$ **fi**, then for each m, whenever T is a subprogram of P_m, $at(T,P) \equiv at(T,P_m)$;*
4. *If $P \equiv$ **while** $M[\overline{q}] = 1$ **do** P' **od** and T is a subprogram of P', then $at(T,P) \equiv at(T,P'); P$.*

Intuitively, $at(T,P)$ is defined as the program that starts at the entrance of T and terminates at the exits of P. Moreover, we define a *configuration ensemble* as a multi-set $\mathcal{A} = \{|\langle P_i, \rho_i\rangle|\}$ of configurations with $\sum_i tr(\rho_i) \le 1$, and particularly, a *terminating configuration ensemble* as $\mathcal{A} = \{|\langle\downarrow, \rho_i\rangle|\}$. For simplicity, we identify a singleton $\{|\langle P, \rho\rangle|\}$ with the configuration $\langle P, \rho\rangle$. Moreover, we need to extend the transition relation between configurations given in Definition 22 to a transition relation between configuration ensembles.

Definition 34 *The transition relation between configuration ensembles is of the form:*

$$\{|\langle P_i, \rho_i\rangle|\} \rightarrow \{|\langle Q_j, \sigma_j\rangle|\}$$

and defined by rules (Sk), (In), (UT), (SC) in Figure 2 together with the rules presented in Figure 5.

(IF′) $\langle \mathbf{if}\ (\square m \cdot M[\overline{q}] = m \rightarrow P_m)\ \mathbf{fi}, \rho\rangle \rightarrow \{|\langle P_m, M_m \rho M_m^\dagger\rangle|\}$

(L′) $\langle \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ P\ \mathbf{od}, \rho\rangle \rightarrow \{|\langle \downarrow, M_0 \rho M_0^\dagger\rangle, \langle P; \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ P\ \mathbf{od}, M_1 \rho M_1^\dagger\rangle|\}$

(MS1) $\dfrac{C \rightarrow \mathcal{B}}{\mathcal{A} \cup \{C\} \rightarrow \mathcal{A} \cup \mathcal{B}}$ (MS2) $\dfrac{\mathcal{A}_1 \rightarrow \mathcal{B}_1 \qquad \mathcal{A}_2 \rightarrow \mathcal{B}_2}{\mathcal{A}_1 \cup \mathcal{A}_2 \rightarrow \mathcal{B}_1 \cup \mathcal{B}_2}$

Fig. 5. Extended Transition Rules for Quantum **while**-Programs. In rule (MS1), $\mathcal{A}$ is a terminating configuration ensemble, $\mathcal{B}$ is a configuration ensemble, and C is a configuration. In rule (MS2), $\mathcal{A}_1, \mathcal{A}_2, \mathcal{B}_1$ and $\mathcal{B}_2$ are all configuration ensembles. Note that in both (MS1) and (MS2), $\cup$ stands for union of multi-sets.

We observe that for each possible measurement outcome m, transition rule (IF) in Figure 2 gives a transition from configuration $\langle \mathbf{if} \cdots \mathbf{fi}, \rho\rangle$. Transition rule (IF') in Figure 5 is essentially a merge of these transitions by collecting all the target configurations into a configuration ensemble. Similarly, transition rule (L') is a merge of (L0) and (L1) in Figure 2. Transition rule (MS1) is introduced simply for lifting transitions of configurations to transitions of configuration ensembles. Rule (MS2) allows us to combine several transitions from some small ensembles into a single transition from a large ensemble.

The semantic function of a quantum program can be represented in terms of the transition between configuration ensembles. For any configuration ensemble $\mathcal{A}$, we define:

$$val(\mathcal{A}) = \sum \{|\rho' : \langle \downarrow, \rho'\rangle \in \mathcal{A}|\}.$$

It is evident that if $\mathcal{A} \rightarrow \mathcal{B}$ then $val(\mathcal{A}) \sqsubseteq val(\mathcal{B})$ because $\langle \downarrow, \rho\rangle$ has no transition; that is, $\langle \downarrow, \rho\rangle \in \mathcal{A}$ implies $\langle \downarrow, \rho\rangle \in \mathcal{B}$.

Definition 35 1. *A computation of a quantum* ***while****-program P starting in a state $\rho \in \mathcal{D}(\mathcal{H}_P)$ is an infinite sequence*

$$\pi = \langle P, \rho\rangle \rightarrow \mathcal{A}_1 \rightarrow \cdots \rightarrow \mathcal{A}_n \rightarrow \cdots,$$

or a finite sequence with a terminating configuration ensemble at the end.

2. *The value of computation π is defined as follows:*

$$val(\pi) = \begin{cases} val(\mathcal{A}_n) & \text{if } \pi \text{ is finite and } \mathcal{A}_n \text{ is the last} \\ & \qquad\qquad \text{configuration ensemble,} \\ \lim_{n\rightarrow\infty} val(\mathcal{A}_n) & \text{if } \pi \text{ is infinite.} \end{cases}$$

Note that in the case of infinite π, sequence $\{val(\mathcal{A}_n)\}$ is increasing according to the Löwner order $\sqsubseteq$. On the other hand, we know that $\mathcal{D}(\mathcal{H}_P)$ with $\sqsubseteq$ is a CPO (see [49], Lemma 3.3.2). So, $\lim_{n\to\infty} val(\mathcal{A}_n)$ exists.

Lemma 31 (Determinism) *For any quantum **while**-program P and $\rho \in \mathcal{D}(\mathcal{H}_P)$, there is exactly one computation π of P starting in ρ and $[\![P]\!](\rho) = val(\pi)$.*

Proof. The uniqueness of the computation $\pi = \langle P, \rho\rangle \to \mathcal{A}_1 \to \cdots \to \mathcal{A}_n \to \cdots$ of P starting in ρ follows immediately from Definition 34. Furthermore, with Definition 23 we have:

$$[\![P]\!](\rho) = \sum\{|\rho' : \langle P,\rho\rangle \to^* \langle\downarrow,\rho'\rangle|\} = \lim_{n\to\infty}\sum\{|\rho' : \langle P,\rho\rangle \to^n \langle\downarrow,\rho'\rangle|\} = \lim_{n\to\infty} val(\mathcal{A}_n) = val(\pi).$$

Now we are ready to present the strong soundness theorem for quantum **while**-programs.

Theorem 31 (Strong Soundness for Quantum while-Programs) *Let $\{A\}P^*\{B\}$ be a standard proof outline for partial correctness of quantum **while**-program P. If $\langle P,\rho\rangle \to^* \{|\langle P_i,\rho_i\rangle|\}$, then:*

1. *for each i, $P_i \equiv at(T_i, P)$ for some subprogram T_i of P or $P_i \equiv \downarrow$; and*
2. *$tr(A\rho) \le \sum_i tr(B_i\rho_i)$, where*

$$B_i = \begin{cases} B & \text{if } P_i \equiv \downarrow, \\ pre(T_i) & \text{if } P_i \equiv at(T_i, P). \end{cases}$$

Proof. See Appendix B.

The soundness for quantum **while**-programs given in Theorem 21 can be easily derived from the above theorem. Of course, the above theorem is a generalisation of the strong soundness for classical **while**-programs (see [4], Theorem 3.3). But it is worthy to notice a difference between them: due to the branching caused by quantum measurements, in the right-hand side of the inequality in clause (2) of the above theorem, we have to take a summation over a configuration ensemble $\{|\langle P_i,\rho_i\rangle|\}$ rather than considering a single configuration $\langle P_i,\rho_i\rangle$.

4 Disjoint Parallel Quantum Programs

Now we start to deal with parallel quantum programs. As the first step, let us consider the simplest case, namely disjoint parallel quantum programs. In classical computing, the behaviour of a disjoint parallel program is simple due to noninterference between its components; in particular, only a simplified version of rule (R.PC) in Figure 1 (without noninterference condition) is needed for reasoning about them (see [4], Lemmas 7.6 and 7.7 and Rule 24 on page 255). As we will see shortly, however, two of the three major challenges pointed out in the Introduction - intertwining nondeterminism and entanglement - already appear in the case of disjoint parallel quantum programs.

4.1 Syntax and Semantics

In this subsection, we define the syntax and operational and denotational semantics of disjoint parallel quantum programs. As we saw in Definitions 22 and 34, the statistical nature of quantum measurements introduces nondeterminism even in the operational semantics of quantum **while**-programs. Such nondeterminism is much more complicated in parallel quantum programs; in particular when they contain loops and thus can have infinite computations, because it is intertwined with another kind of nondeterminism, namely nondeterminism introduced in parallelism. But surprisingly, the determinism is still true for the denotational semantics of disjoint parallel quantum programs, and it further entails that disjoint parallel compositions of quantum programs can always be sequentialised.

Definition 41 (Syntax) *Disjoint parallel quantum programs are generated by the grammar given in equations (1) and (2) together with the following clause:*

$$P ::= P_1 \| \cdots \| P_n \tag{5}$$

*where $n > 1$, $P_1, ..., P_n$ are quantum **while**-programs, and $var(P_i) \cap var(P_j) = \emptyset$ for $i \neq j$.*

Program P in equation (5) is called the disjoint parallel composition of $P_1, ..., P_n$. We write: $var(P) = \bigcup_{i=1}^{n} var(P_i)$ for the set of quantum variables in P. Thus, the state Hilbert space of P is $\mathcal{H}_P = \mathcal{H}_{var(P)} = \bigotimes_{i=1}^{n} \mathcal{H}_{P_i}$.

Definition 42 (Operational Semantics) *The operational semantics of disjoint parallel quantum program is the transition relation between configuration ensembles defined by the rules used in Definition 34 together with rule (PC) in Figure 6.*

$$\text{(PC)}\quad \frac{\langle P_i, \rho\rangle \to \{|\langle P'_{ij}, \rho'_j\rangle|\}}{\langle P_1\| \cdots \|P_{i-1}\|P_i\|P_{i+1}\| \cdots \|P_n, \rho\rangle \to \{|\langle P_1\| \cdots \|P_{i-1}\|P'_{ij}\|P_{i+1}\| \cdots \|P_n, \rho'_j\rangle|\}}$$

Fig. 6. Transition Rule for (Disjoint) Parallel Quantum Programs. Here, $1 \leq i \leq n$.

Intuitively, transition rule (PC) models interleaving concurrency; more precisely, it means that for a fixed $1 \leq i \leq n$, the ith component P_i of parallel quantum programs $P \equiv P_1\| \cdots \|P_n$ performs a transition, then P can perform the same transition.

To further illustrate the transition rule (PC), we consider the following simple example . In this paper, to simplify the presentation, for a pure state $|\varphi\rangle$ and a complex number α with $|\alpha| \leq 1$, we often use the vector $\alpha|\varphi\rangle$ to denote the corresponding partial density operator $|\alpha|^2|\varphi\rangle\langle\varphi|$.

Example 41 *Let p, q, r be three qubit variables,*

$$P_1 \equiv \ p := X[p]; q := Z[q], \qquad\qquad P_2 \equiv \textbf{if } M[r] = 0 \rightarrow \textbf{skip} \qquad \square \qquad 1 \rightarrow r := H[r] \qquad \textbf{fi}$$

where X, Z are the Pauli gates, H the Hadamard gate and $M = \{M_0 = |0\rangle\langle 0|, M_1 = |1\rangle\langle 1|\}$ is the measurement in the computational basis, and let $|\psi\rangle = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)$ be the GHZ (Greenberger-Horne-Zeilinger) state. Then

$$\langle P_1\|P_2, |\psi\rangle\rangle \rightarrow_1 \langle q := Z[q]\|P_2, \frac{1}{\sqrt{2}}(|100\rangle + |011\rangle)\rangle \rightarrow_2 \begin{cases} \langle q := Z[q]\|\textbf{skip}, \frac{1}{\sqrt{2}}|100\rangle\rangle \\ \langle q := Z[q]\|r := H[r], \frac{1}{\sqrt{2}}|011\rangle\rangle \end{cases}$$

$$\rightarrow_{1,2} \begin{cases} \langle \downarrow \|\textbf{skip}, \frac{1}{\sqrt{2}}|100\rangle\rangle \\ \langle q := Z[q]\| \downarrow, \frac{1}{\sqrt{2}}|01-\rangle\rangle \end{cases} \qquad \rightarrow_{2,1} \begin{cases} \langle \downarrow, \frac{1}{\sqrt{2}}|100\rangle\rangle \\ \langle \downarrow, -\frac{1}{\sqrt{2}}|01-\rangle\rangle \end{cases}$$

is a computation of parallel program $P_1\|P_2$ starting in state $|\psi\rangle$. Here, we use $\rightarrow_i$ to indicate that the transition is made by P_i according to rule (PC), and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

It is interesting to see that at the second step of the computation in the above example, measurement M is performed by component P_2 and thus certain nondeterminism occurs; that is, two different configurations are produced according to the two different outcomes $0, 1$ of M. Then in steps 3 and 4, the following kind of interleaving appears: the two different components P_1 and P_2 executed respectively on the two different configurations that come from the same measurement M. Here, in a sense, nondeterminism caused by quantum measurements is intertwined with nondeterminism introduced by parallelism. It is worth noting that for a classical parallel program $P \equiv P_1\| \cdots \|P_n$ with P_i $(1 \leq i \leq n)$ being **while**-programs, such an interleaving never happens because nondeterminism does not occur in the execution of any component P_i.

Now we are ready to introduce the denotational semantics of disjoint parallel quantum programs. But it cannot be defined by simply mimicking Definitions 23 and 35.

Definition 43 (Denotational Semantics) *The semantic function of a disjoint parallel program P is the mapping $[\![P]\!] : \mathcal{D}(\mathcal{H}_P) \rightarrow 2^{\mathcal{D}(\mathcal{H}_P)}$ defined by*

$$[\![P]\!](\rho) = \{val(\pi) : \pi \text{ is a computation of } P \text{ starting in } \rho\},$$

for any $\rho \in \mathcal{D}(\mathcal{H}_P)$, where $val(\pi)$ is given in Definition 35.

As stated at the beginning of this subsection, the denotational semantics of a disjoint parallel quantum program is deterministic although its operational semantics may demonstrate a very complicated nondeterminism.

Lemma 41 (Determinism) *For any disjoint parallel quantum program P and $\rho \in \mathcal{D}(\mathcal{H}_P)$, $[\![P]\!](\rho)$ is a singleton.*

Proof. See Appendix C.

For a disjoint parallel quantum program P and for any $\rho \in \mathcal{D}(\mathcal{H}_P)$, if singleton $[\![P]\!](\rho) = \{\rho'\}$, then we will always identify $[\![P]\!](\rho)$ with the partial density operator ρ'.

It is well-known that every disjoint parallel composition of classical **while**-programs can be sequentialised (see [4], Lemma 7.7). This result can also be generalised to the quantum case.

Lemma 42 (Sequentialisation) *Suppose that quantum **while**-programs $P_1, \cdots, P_n$ are disjoint. Then:*

1. *For any permutation $i_1, \cdots, i_n$ of $1, \cdots, n$, $[\![P_1\| \cdots \|P_n]\!] = [\![P_{i_1}\| \cdots \|P_{i_n}]\!]$.*
2. *$[\![P_1\| \cdots \|P_n]\!] = [\![P_1; \cdots ; P_n]\!]$.*

Proof. See Appendix D.

4.2 Correctness of Disjoint Parallel Programs

Due to its determinism (Lemma 41), (partial and total) correctness of a disjoint parallel quantum program P can be defined simply using Definition 25 provided that for each input ρ, we identify the singleton $[\![P]\!](\rho) = \{\rho'\}$ with the partial density operator ρ'.

In this subsection, we derive a series of rules for proving correctness of disjoint parallel programs. Naturally, we first try to find appropriate quantum generalisations of the inference rules for classical disjoint parallel programs. But at the end of this subsection, we will see that a novel rule that has no classical counterpart is needed to cope with entanglement.

Sequentialisation Rule To warm up, let us first consider a simple inference rule. As mentioned in the previous subsection, all disjoint parallel programs in classical computing can be sequentialised with the same denotational semantics. Accordingly, they can be verified through sequentialisation ([4], Section 7.3). For quantum computing, the following sequentialisation rule is valid too:

$$\text{(R.Seq)} \qquad \frac{\{A\}\, P_1; \cdots ; P_n\, \{B\}}{\{A\}\, P_1\| \cdots \|P_n\, \{B\}}$$

Fig. 7. Sequentialisation Rule for Disjoint Parallel Programs.

Lemma 43 *The rule (R.Seq) is sound for both partial and total correctness.*

Proof. Immediate from Lemma 42.3.

We give a simple application of rule (R.Seq) for disjoint parallel quantum programs. The following simple example was given in [4] to show the necessity of introducing auxiliary variables:

$$\{x = y\}x := x + 1\|y := y + 1\{x = y\}.$$

This correctness formula for a disjoint parallel program cannot be proved by merely using the parallel composition rule (R.PC) in Fig. 1. However, it can be simply derived by rule (R.Seq). Here, we consider a quantum analog of it.

Example 42 *Let p, q be two quantum variables with the same state Hilbert space $\mathcal{H}$. For each orthonormal basis $\Phi = \{|\varphi_i\rangle\}$ of $\mathcal{H}$, we define a quantum predicate:*

$$A_\Phi = \sum_i p_i|\varphi_i\varphi_i\rangle\langle\varphi_i\varphi_i| \tag{6}$$

in $\mathcal{H}\otimes\mathcal{H}$, where $\{p_i\}$ is a probability distribution. It can be viewed as a quantum counterpart of equality $x = y$. It is interesting to note that the quantum counterpart of $x = y$ is not unique because for different bases $\Phi = \{|\varphi_i\rangle\}$, A_Φ are different. For any unitary operator U in $\mathcal{H}$, we have:

$$\models_{tot} \{A_\Phi\}p := U[p]\|q := U[q]\{A_{U(\Phi)}\} \tag{7}$$

where $A_{U(\Phi)}$ is the quantum counterpart of equality defined by orthonormal basis $U(\Phi) = \{U|\varphi_i\rangle\}$. Clearly, (7) can be proved using rule (R.Seq) together with (Ax.UT) in Figure 3.

It is worth pointing out that the quantum generalisation of a concept in a classical system usually has the flexibility arising from different choices of the basis of its state Hilbert space.

Tensor product of quantum predicates We now start to consider how the crucial rule for reasoning about parallel programs, rule (R.PC) in Figure 1, can be generalised to the quantum case. To this end, we first need to identify a quantum counterpart of conjunction $\bigwedge_{i=1}^n A_i$ (and $\bigwedge_{i=1}^n B_i$) in rule (R.PC). For disjoint parallel quantum programs, a natural choice is tensor product $\bigotimes_{i=1}^n A_i$ because it enjoys a nice physical interpretation:

$$tr\left(\left(\bigotimes_{i=1}^n A_i\right)\left(\bigotimes_{i=1}^n \rho_i\right)\right) = \prod_{i=1}^n tr\left(A_i\rho_i\right).$$

The above equation shows that the probability that a product state $\bigotimes_{i=1}^n \rho_i$ satisfies quantum predicate $\bigotimes_{i=1}^n A_i$ is the product of the probabilities that each component state ρ_i satisfies the corresponding predicate A_i. This observation motivates an inference rule for tensor product of quantum predicates presented in Figure 8. It can be seen as the simplest quantum generalisation of rule (R.PC) in Figure 1.

$$\text{(R.PC.P)} \qquad \frac{\{A_i\}\, P_i\, \{B_i\} \ \ (i=1,...,n)}{\left\{\bigotimes_{i=1}^n A_i\right\} P_1\| \cdots \|P_n \left\{\bigotimes_{i=1}^n B_i\right\}}$$

Fig. 8. Rule for Tensor Product of Quantum Predicates

Lemma 44 *The rule (R.PC.P) is sound with respect to both partial and total correctness.*

Proof. See Appendix E.

The rule (R.PC.P) can only be used to infer correctness of disjoint parallel quantum programs with respect to (tensor) product predicates. For instance, we can use (R.PC.P) to prove a very special case of (7) in Example 42 with $\{p_i\}$ being a degenerate distribution at some i_0:

$$\vdash_{tot} \{|\varphi\varphi\rangle\langle\varphi\varphi|\}\, p := U[p] \| q := U[q]\, \{|\psi\psi\rangle\langle\psi\psi|)|\}$$

where $|\varphi\rangle = |\varphi_{i_0}\rangle$ and $|\psi\rangle = U|\varphi\rangle$, but it is not strong enough to derive the entire (7).

Separable Quantum Predicates A larger family of predicates in $\bigotimes_{i=1}^n \mathcal{H}_{P_i}$ than product predicates is separable predicates defined in the following:

Definition 44 *Let A be a quantum predicate in $\bigotimes_{i=1}^n \mathcal{H}_{P_i}$. Then:*

1. *A is said to be separable if there exist $p_j \geq 0$ and quantum predicates A_{ji} in $\mathcal{H}_{P_i}$ $(i = 1, ..., n;\ j = 1, ..., m)$ such that $\sum_{j=1}^m p_j \leq 1$ and*
$$A = \sum_{j=1}^{m} p_j \left(\bigotimes_{i=1}^{n} A_{ji} \right)$$
where m is a positive integer or ∞.
2. *A is entangled if it is not separable.*

To deal with separable preconditions and postconditions, we need several auxiliary axioms and rules. For any $X \subseteq Y \subseteq Var$ and operator A in $\mathcal{H}_X = \bigotimes_{q\in X} \mathcal{H}_q$, $cl_Y(A) = A \otimes I_{\mathcal{H}_{Y\setminus X}}$ is called the cylindric extension of A in $\mathcal{H}_Y$. The needed axioms and rules are presented in Figure 9. A comparison between these axioms and rules and their counterparts for classical programs should be helpful for the reader. The appearance of axiom (Ax.Inv) is the same as axiom (INVARIANCE) in Section 3.8 of [4]. Rules (R.CC) and (R.Inv) are generalisations of rules (CONJUNCTION) and (INVARIANCE) in [4], respectively, with logical conjunction being replaced by a probabilistic (convex) combination. But rule (R.Lim) has no counterpart for classical programs.

The following lemma establishes soundness of the auxiliary rules in Figure 9.

(Ax.Inv) $\{A\}\,P\,\{A\}$ (R.Inv) $\dfrac{\{A\}P\{B\}}{\{pA+qC\}P\{pB+qC\}}$

(R.CC) $\dfrac{\{A_i\}\,P\,\{B_i\}\ \ (i=1,...,m)}{\left\{\sum_{i=1}^m p_iA_i\right\}\,P\,\left\{\sum_{i=1}^m p_iB_i\right\}}$ (R.Lim) $\dfrac{\lim_{n\to\infty} A_n = A \quad \{A_n\}\,P\,\{B_n\} \quad \lim_{n\to\infty} B_n = B}{\{A\}P\{B\}}$

Fig. 9. Auxiliary Rules. In (Ax.Inv), $var(P)\cap V=\emptyset$ and $A=cl_{V\cup var(P)}(B)$ for some $V\subseteq Var$ and for some quantum predicate B in $\mathcal{H}_V$. In (R.Inv), $p,q\geq 0$, $p+q\leq 1$, and C is a quantum predicate in $\mathcal{H}_V$ for some $V\subseteq Var$ with $V\cap var(P)=\emptyset$. In (R.CC), $p_i\geq 0$ $(i=1,...,m)$ and $\sum_{i=1}^m p_j\leq 1$. In (R.Lim), $\{A_n\}$ and $\{B_n\}$ are increasing and decreasing, respectively sequences with respect to the Löwner order.

Lemma 45 *1. The axiom (Ax.Inv) is sound for partial correctness.*
2. The rules (R.CC), (R.Inv) and (R.Lim) are sound both for partial and total correctness.

Proof. See Appendix F.

A combination of rule (R.PC.P) with the auxiliary axioms and rules (R.CC), (Ax.Inv), (R.Inv) and (R.Lim) in Figure 9 yields rule (R.PC.S) in Figure 10.

(R.PC.S) $$\frac{\{A_{ji}\}\,P_i\,\{B_{ji}\}\ \ (i=1,...,n;\ j=1,...,m)}{\left\{\sum_{j=1}^m p_j\left(\bigotimes_{i=1}^n A_{ji}\right)\right\}P_1\|\cdots\|P_n\left\{\sum_{j=1}^m p_j\left(\bigotimes_{i=1}^n B_{ji}\right)\right\}}$$

Fig. 10. Rule for Separable Quantum Predicates. Coefficients $p_j\geq 0$ and $\sum_{j=1}^m p_j\leq 1$; m is a positive integer or ∞.

Obviously, rule (R.PC.S) can reason about disjoint parallel quantum programs with separable quantum predicates; for example, correctness (7) in Example 42 can be proved using rule (R.PC.S).

Entangled Quantum Predicates It is well-understood that entanglement is an indispensable physical resource that makes quantum computers outperform classical computers. Obviously, inference rule (R.PC.S) is unable to prove any correctness of the form $\{A\}P_1\|\cdots\|P_n\{B\}$ for a parallel quantum program $P_1\|\cdots\|P_n$ where A or B is an entangled predicate, as shown in the following:

Example 43 *We consider a variant of Example 42. For each orthonormal basis $\Phi=\{|\varphi_i\rangle\}$ of $\mathcal{H}$, we write: $\beta_\Phi=\sum_i|\varphi_i\varphi_i\rangle$ for the (unnormalised) maximally entangled state in $\mathcal{H}\otimes\mathcal{H}$. Then $E_\Phi=|\beta_\Phi\rangle\langle\beta_\Phi|$ can be seen as another quantum counterpart of equality $x=y$ (different from A_Φ defined by equation (6)). Obviously,*

$$\models_{tot}\{|\beta_\Phi\rangle\langle\beta_\Phi|\}\,p:=U[p]\|q:=U[q]\left\{|\beta_{U(\Phi)}\rangle\langle\beta_{U(\Phi)}|\right\}; \quad (8)$$

that is, if the input is maximally entangled, so is the output. Indeed, we can prove correctness formula (8) by using rules (R.Seq) and (Ax.UT), but (8) cannot be derived by directly using rule (R.PC.S).

Auxiliary Variables Interestingly, (R.PC.S) itself is not strong enough to prove correctness (8) with entangled precondition and postcondition, but a combination of it and several rules for introducing auxiliary variables can. For any $X, Y \subseteq \mathit{Var}$ with $X \cap Y = \emptyset$, the partial trace tr_Y is a mapping from operators in $\mathcal{H}_{X\cup Y}$ to operators in $\mathcal{H}_X$ defined by $tr_Y(|\varphi\rangle\langle\psi| \otimes |\varphi'\rangle\langle\psi'|) = \langle\psi'|\varphi'\rangle \cdot |\varphi\rangle\langle\psi|$ for every $|\varphi\rangle, |\psi\rangle$ in $\mathcal{H}_X$ and $|\varphi'\rangle, |\psi'\rangle$ in $\mathcal{H}_Y$, together with linearity. With this notation, the needed rules are presented in Figure 11. It is interesting to note

$$\text{(R.TI)}\quad \frac{\{A\}P\{B\}}{\{tr_W A\}\,P\,\{B\}} \qquad\qquad \text{(R.SO)}\quad \frac{\{A\}P\{B\}}{\{\mathcal{E}^*(A)\}\,P\,\{\mathcal{E}^*(B)\}}$$

Fig. 11. Auxiliary Rules. In (R.TI), $V, W \subseteq \mathit{Var}$, $V \cap W = \emptyset$, A, B are quantum predicates in $\mathcal{H}_{V\cup W}$ and $\mathcal{H}_V$, respectively, and $\mathit{var}(P) \subseteq V$. In (R.SO), $\mathcal{E}$ is a super-operator in $\mathcal{H}_V$ for some $V \subseteq \mathit{Var}$ with $V \cap \mathit{var}(P) = \emptyset$.

that rule (R.TI) is a quantum generalisation of two rules (DISJUNCTION) and ($\exists$-INTRODUCTION) in Section 3.8 of [4], where partial trace is considered as a quantum counterpart of logical disjunction and existence quantifier. Rule (R.SO) is a quantum generalisation of rule (SUBSTITUTION) there, with the substitution $\overline{z} := \overline{t}$ being replaced by a super-operator $\mathcal{E}$.

The following lemma establishes soundness of the rules in Figure 11.

Lemma 46 *The rules (R.TI) and (R.SO) are sound both for partial and total correctness.*

Proof. See Appendix F.

Now let us see how the rules in Figure 11 can be used to prove correctness (8) in Example 43. The key idea is to introduce two auxiliary variables p', q' with the same state space $\mathcal{H}$. First, by (Ax.UT) we have:

$$\vdash_{tot} \left\{(E_\Phi)_{pp'}\right\} p := U[p]\,\{|\alpha\rangle_{pp'}\langle\alpha|\}\,, \qquad \vdash_{tot} \left\{(E_\Phi)_{qq'}\right\} q := U[q]\,\{|\alpha\rangle_{qq'}\langle\alpha|\} \tag{9}$$

where we use subscripts p, q, p', q' to indicate the corresponding subsystems, and $|\alpha\rangle = \sum_i (U|i\rangle)\,|i\rangle$. Now applying rule (R.PC.S) to (9) yields:

$$\vdash_{tot} \left\{(E_\Phi)_{pp'} \otimes (E_\Phi)_{qq'}\right\} p := U[p] \| q := U[q]\,\{|\alpha\rangle_{pp'}\langle\alpha| \otimes |\alpha\rangle_{qq'}\langle\alpha|\} \tag{10}$$

Finally, we define superoperator:

$$\mathcal{E}(\rho) = \sum_i \left(|\beta\rangle_{p'q'}\langle i|\right) \rho \left(|i\rangle_{p'q'}\langle\beta|\right)$$

for all mixed states ρ of p' and q', and obtain (8) by applying rule (R.SO) to (10) because

$$E_\Phi \otimes I_{p'q'} = \mathcal{E}^* \left((E_\Phi)_{pp'} \otimes (E_\Phi)_{qq'} \right) = \sum_i (|i\rangle_{p'q'}\langle\beta|) \left((E_\Phi)_{pp'} \otimes (E_\Phi)_{qq'} \right) (|\beta\rangle_{p'q'}\langle i|),$$

$$E_{U(\Phi)} \otimes I_{p'q'} = \mathcal{E}^* \left(|\alpha\rangle_{pp'}\langle\alpha| \otimes |\alpha\rangle_{qq'}\langle\alpha| \right) = \sum_i (|i\rangle_{p'q'}\langle\beta|) \left(|\alpha\rangle_{pp'}\langle\alpha| \otimes |\alpha\rangle_{qq'}\langle\alpha| \right) (|\beta\rangle_{p'q'}\langle i|).$$

Although the strategy of introducing auxiliary variables works for the above example, we do not know whether it can deal with all entangled preconditions and postconditions or not.

Transferring Separable Predicates to Entangled Predicates Fortunately, a deep result in the theoretical analysis of NMR (Nuclear Magnetic Resonance) quantum computing provides us with another solution. It was discovered in [52], [6] that all mixed states of n qubits in a sufficiently small neighbourhood of the maximally mixed state are separable. The interpretation of this result in physics is that entanglement cannot exist in the presence of too much noise. The result was generalised in [19] to the case of any quantum systems with finite-dimensional state Hilbert spaces. Recall that the Hilbert-Schmidt norm (or 2-norm) of operator A is defined as follows: $\|A\|_2 = \sqrt{tr(A^\dagger A)}$. In particular, if $A = (A_{ij})$ is a matrix, then $\|A\|_2 = \sqrt{\sum_{i,j} |A_{ij}|^2}$.

Theorem 41 (Gurvits and Barnum [19]) *Let $\mathcal{H}_1, \cdots, \mathcal{H}_n$ be finite-dimensional Hilbert spaces, and let A be a positive operator in $\bigotimes_{i=1}^n \mathcal{H}_n$. If*

$$\|A - I\|_2 \leq \frac{1}{2^{n/2-1}}$$

where I is the identity operator in $\bigotimes_{i=1}^n \mathcal{H}_n$, then A is separable.

The following corollary can be easily derived from the above theorem.

Corollary 41 *For any two positive operators A, B in $\bigotimes_{i=1}^n \mathcal{H}_i$, there exists $0 < \epsilon \leq 1$ such that both $(1-\epsilon)I + \epsilon A$ and $(1-\epsilon)I + \epsilon B$ are separable.*

Proof. Let $C = (1-\epsilon)I + \epsilon A$ and $D = (1-\epsilon)I + \epsilon B$. Then $\|C - I\|_2 = \epsilon\|A - I\|_2$ and $\|D - I\|_2 = \epsilon\|B - I\|_2$. So, by Theorem 41 it suffices to take

$$\epsilon \leq \frac{1}{2^{n/2-1}\max[\|A - I\|_2, \|B - I\|_2]}.$$

Motivated by Corollary 41, we introduce a new inference rule (R.S2E) in Figure 12.

The idea behind rule (R.S2E) is that in order to prove correctness $\{A\}P_1\|\cdots\|P_n\{B\}$ for entangled predicates A and B, we find a parameter $\epsilon > 0$ such that $(1-\epsilon)I + \epsilon A$ and $(1-\epsilon)I + \epsilon B$ are separable, and then we can prove:

$$\{(1-\epsilon)I + \epsilon A\}P_1\|\cdots\|P_n\{(1-\epsilon)I + \epsilon B\}$$

(R.S2E) $$\frac{\{(1-\epsilon)I+\epsilon A\}\,P\,\{(1-\epsilon)I+\epsilon B\}}{\{A\}\,P\,\{B\}}$$

Fig. 12. Rule for Transforming Separable Predicates to Entangled Predicates. Here, $0<\epsilon\leq 1$.

by using rule (R.PC.S). It is worth pointing out that Corollary 41 warrants that we can choose the same parameter ϵ in the precondition and postcondition.

Example 44 *For $k=0,1$, consider the quantum program* $\mathbf{while}_k$ *given in Example 4.2. We write: $|\Phi\rangle=\frac{1}{\sqrt{2}}(|01\rangle+|10\rangle)$ for a maximally entangled state of a 2-qubit system. Then it holds that*

$$\models_{par}\left\{I_4-\frac{1}{2}|10\rangle\langle 10|\right\}\mathbf{while}_0\parallel\mathbf{while}_1\{|\Phi\rangle\langle\Phi|\},\tag{11}$$

where I_4 is the 4×4 unit matrix. The correctness formula (11) has entangled precondition and postcondition, and thus cannot be proved by only using rule (R.PC.S). Here, we show that it can be proved by combining rule (R.S2E) with (R.PC.S). In fact, one can first verify that

$$\models_{par}\left\{I_2-|\alpha|^2|1-k\rangle\langle 1-k|\right\}\mathbf{while}_k\{|\psi\rangle\langle\psi|\}\tag{12}$$

for $k=0,1$ and any state $|\psi\rangle=\alpha|k\rangle+\beta|1-k\rangle$, where I_2 is the 2×2 unit matrix. Moreover, we write:

$$|\curvearrowright\rangle=\frac{1}{\sqrt{2}}(|0\rangle+\mathrm{i}|1\rangle),\ |\curvearrowleft\rangle=\frac{1}{\sqrt{2}}(|0\rangle-\mathrm{i}|1\rangle).$$

Then we have the following decomposition of separable operator:

$$\left(1-\frac{2}{3}\right)I_4+\frac{2}{3}|\Phi\rangle\langle\Phi|=\frac{1}{3}(|01\rangle\langle 01|+|10\rangle\langle 10|+|++\rangle\langle ++|+|--\rangle\langle --|$$
$$+|\curvearrowright\curvearrowright\rangle\langle\curvearrowright\curvearrowright|+|\curvearrowleft\curvearrowleft\rangle\langle\curvearrowleft\curvearrowleft|),$$

and it is derived that

$$\left\{I_4-\frac{1}{3}|10\rangle\langle 10|\right\}\mathbf{while}_0\parallel\mathbf{while}_1\left\{(1-\frac{2}{3})I_4+\frac{2}{3}|\Phi\rangle\langle\Phi|\right\}\tag{13}$$

by applying (12) for $|\psi\rangle=|0\rangle,|1\rangle,|+\rangle,|-\rangle,|\curvearrowright\rangle,|\curvearrowleft\rangle$ and $i=0,1$, respectively, and applying rule (R.PC.S). Finally, correctness (11) is obtained by applying rule (R.S2E) to (13) with $\epsilon=\frac{2}{3}$.

To conclude this section, we present the soundness of inference rule R.S2E).

Lemma 47 *The rule (R.S2E) is sound for both partial and total correctness.*

Proof. See Appendix G.

5 Parallel Quantum Programs with Shared Variables

Disjoint parallel quantum programs were considered in the last section. In this section, we deal with a class of more general parallel quantum programs, namely parallel quantum programs with shared variables. Our main aim is to find an appropriate quantum extension of inference rule (R.PC) in Figure 1. Then the third major challenge pointed out in the Introduction - combining quantum predicates in the overlap of state Hilbert spaces - will emerge in the case of shared variables.

5.1 Syntax and Semantics

In this subsection, we define the syntax and operational and denotational semantics of parallel quantum programs with shared variables. Superficially, they are straightforward generalisations of the corresponding notions in classical programming. But as we already saw in Subsection 4.1, nondeterminism induced by quantum measurements and its intertwining with parallelism; in particular when some infinite computations of loops are involved, make the semantics much harder to deal with than in the classical case.

Definition 51 *1. Component quantum programs are generated by the grammar given in Eqs. (1), (2) and (3) together with the following clause for atomic regions:*

$$P ::= \langle P_0 \rangle$$

where P_0 is loop-free and contains no further atomic regions; that is, it is generated only by Eqs. (1) and (2).

2. Parallel quantum programs (with shared variables) are generated by the grammar given in Eqs. (1), (2) and (3) together with the following clause for parallel composition:

$$P ::= P_1 \| \cdots \| P_n$$

where $n > 1$, and $P_1, ..., P_n$ are component quantum programs.

The syntax of parallel quantum programs defined above is similar to that of classical parallel programs. Furthermore, a *normal* subprogram of program P is a subprogram of P that does not occur within any atomic region of P.

The set of quantum variables in a parallel quantum program is defined as follows: $var(\langle P \rangle) = var(P)$, and if $P \equiv P_1 \| \cdots \| P_n$ then $var(P) = \bigcup_{i=1}^{n} var(P_i)$. Furthermore, the state Hilbert space of a parallel quantum program P is $\mathcal{H}_P = \mathcal{H}_{var(P)}$. It is worth pointing out that in general for a parallel quantum program $P \equiv P_1 \| \cdots \| P_n$ with shared variables, $\mathcal{H}_P \neq \bigotimes_{i=1}^{n} \mathcal{H}_{P_i}$ because it is not required that $var(P_1), \cdots, var(P_n)$ are disjoint.

Definition 52 *The operational semantics of parallel quantum programs is defined by the transitions rules in Figures 2 and 6 and rule (AR) in Figure 13 for atomic regions:*

$$\text{(AR)} \qquad \frac{\langle P,\rho\rangle \rightarrow^* \langle\downarrow,\rho'\rangle}{\langle\langle P\rangle,\rho\rangle \rightarrow \langle\downarrow,\rho'\rangle}$$

Fig. 13. Transition Rule for Atomic Regions

The rule (AR) means that any terminating computation of P is reduced to a single-step computation of atomic region $\langle P\rangle$. Such a reduction guarantees that a computation of $\langle P\rangle$ may not be interfered with by other components in a parallel composition.

Based on the operational semantics defined above, the denotational semantics of parallel quantum programs with shared variables can be defined in the same way as Definition 43. In particular, it follows from rule (AR) that $[\![\langle P\rangle]\!](\rho) = [\![P]\!](\rho)$ for any input ρ.

The following example shows the difference between the behaviours of a quantum program and its atomic version in parallel with another quantum program involving a quantum measurement on a shared variable.

Example 51 *Let p, q be qubit variables and*

$$P_1 \equiv p := H[p]; p := H[p], \qquad P_1' \equiv \langle P_1\rangle, \qquad P_2 \equiv \mathbf{if}\ \begin{array}{l} M[p] = 0 \rightarrow \mathbf{skip} \\ \square \qquad\qquad 1 \rightarrow q := X[q] \\ \mathbf{fi} \end{array}$$

where H, X are the Hadamard and Pauli gates, respectively and M the measurement in the computational basis. Consider the EPR (Einstein-Podolsky-Rosen) pair $|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ as an input, where the first qubit is p and the second is q.

1. *One of the computations of parallel composition $P_1'\|P_2$ is*

$$\pi_1 = \langle P_1'\|P_2, |\psi\rangle\rangle \rightarrow_1 \langle\downarrow\|P_2, |\psi\rangle\rangle \rightarrow_2 \left\{\langle\downarrow\|\mathbf{skip}, \frac{1}{\sqrt{2}}|00\rangle\rangle, \langle\downarrow\|q := X[q], \frac{1}{\sqrt{2}}|11\rangle\rangle\right\}$$
$$\rightarrow_2 \left\{\langle\downarrow, \frac{1}{\sqrt{2}}|00\rangle\rangle, \langle\downarrow, \frac{1}{\sqrt{2}}|10\rangle\rangle\right\}$$

Indeed, for all other computations π of $P_1'\|P_2$ starting in $|\psi\rangle$, we have:

$$val(\pi) = val(\pi_1) = \frac{1}{2}(|00\rangle\langle 00| + |10\rangle\langle 10|) \stackrel{\triangle}{=} \rho_1,$$

and thus $[\![P_1'\|P_2]\!](|\psi\rangle) = \{\rho_1\}$.

2. $P_1\|P_2$ has a computation starting in $|\psi\rangle$ that is quite different from π_1:

$$\pi_2 = \langle P_1\|P_2, |\psi\rangle\rangle \rightarrow_1 \left\{\langle p := H[p]\|P_2, \frac{1}{\sqrt{2}}(|+0\rangle + |-1\rangle)\rangle\right\}$$

$$\rightarrow_2 \begin{cases}\langle p := H[p]\|\mathbf{skip}, \frac{1}{2}(|00\rangle + |01\rangle)\rangle, \\ \langle p := H[p]\|q := X[q], \frac{1}{2}(|10\rangle - |11\rangle)\rangle\end{cases} \rightarrow_2 \begin{cases}\langle p := H[p]\|\downarrow, \frac{1}{2}(|00\rangle + |01\rangle)\rangle, \\ \langle p := H[p]\|\downarrow, \frac{1}{2}(|11\rangle - |10\rangle)\rangle\end{cases}$$

$$\rightarrow_1 \left\{\langle\downarrow, \frac{1}{2}(|+0\rangle + |+1\rangle)\rangle, \langle\downarrow, \frac{1}{2}(|-1\rangle - |-0\rangle)\rangle\right\}$$

We have:

$$\begin{aligned} val(\pi_1) \neq val(\pi_2) = \frac{1}{4}(&|00\rangle\langle 00| + |00\rangle\langle 11| + |01\rangle\langle 01| + |01\rangle\langle 10| \\ &+ |10\rangle\langle 01| + |10\rangle\langle 10| + |11\rangle\langle 00| + |11\rangle\langle 11|) \stackrel{\triangle}{=} \rho_2 \end{aligned}$$

and $[\![P_1\|P_2]\!](|\psi\rangle) = \{\rho_1, \rho_2\}$.

The above example indicates that the determinism of the denotational semantics of disjoint parallel quantum programs (Lemma 41) is no longer true for parallel quantum programs with shared variables.

5.2 Correctness of Parallel Quantum Programs

First of all, Example 51 shows that for a parallel quantum program P with shared variables and an input ρ, $[\![P]\!](\rho)$ may have more than one element. Therefore, the definition of correctness of quantum **while**-programs (Definition 25) is not directly applicable to parallel quantum programs with shared variables. But a simple modification of it works.

Definition 53 (Partial and Total Correctness) *Let P be a parallel quantum program (with shared variables) and A, B quantum predicates in $\mathcal{H}_P$. Then the correctness formula $\{A\}P\{B\}$ is true in the sense of total correctness (resp. partial correctness), written $\models_{tot} \{A\}P\{B\}$ (resp. $\models_{par} \{A\}P\{B\}$), if for each input $\rho \in \mathcal{D}(\mathcal{H}_P)$, it holds that*

$$tr(A\rho) \leq tr(B\rho') \ (resp.\ tr(A\rho) \leq tr(B\rho') + [tr(\rho) - tr(\rho')])$$

for all $\rho' \in [\![P]\!](\rho)$.

In this subsection, we introduce some rules for reasoning about correctness of parallel quantum programs with shared variables.

Correctness of Component Quantum Programs We first consider component quantum programs. The proof techniques for classical component programs can be generalised to the quantum case without any difficulty. More precisely, partial correctness of component quantum programs can be verified with the

$$\text{(R.At)} \qquad \frac{\{A\}P\{B\}}{\{A\}\langle P\rangle\{B\}}$$

Fig. 14. Rule for Atomic Regions.

proof system qPD for quantum **while**-programs plus the rule (R.AT) in Figure 14 for atomic regions.

Proof outlines for partial correctness of component quantum programs are generated by the rules in Figure 4 together with the rule (R.At') in Figure 15. A proof outline of a component program P is standard if every normal subprogram Q is preceded by exactly one quantum predicate $pre(Q)$. The notation $at(T,P)$ is defined in the same way as in Definition 33, but only for normal subprograms T of P. The strong soundness theorem for quantum **while**-programs (Theorem 31) can be easily generalised to the case of component quantum programs.

$$\text{(R.At}'\text{)} \qquad \frac{\{A\}P^*\{B\}}{\{A\}\langle P\rangle\{B\}}$$

Fig. 15. Rule for Atomic Regions.

Interference Freedom Now we consider how can we reason about correctness of parallel quantum programs with shared variables. The following example shows non-compositionality in the sense that correctness of a parallel quantum program is not solely determined by correctness of its component programs.

Example 52 *Let q be a quantum variable of type* **Bool** *(Boolean) or* **Int** *(Integers). Consider the following two programs:*

$$P_1 \equiv q := U[q], \qquad P_1' \equiv q := V[q]; q := W[q]$$

where U, V, W are unitary operators in $\mathcal{H}_q$ such that $U = WV$. It is obvious that P_1 and P_1' are equivalent in the following sense: for any quantum predicates A, B in $\mathcal{H}_q$,

$$\models_{par} \{A\}P_1\{B\} \Leftrightarrow \models_{par} \{A\}P_1'\{B\}.$$

Now let us further consider their parallel composition with the simple initialisation program:

$$P_2 \equiv q := |0\rangle.$$

We show that $P_1\|P_2$ and $P_1'\|P_2$ are not equivalent; that is,

$$\models_{par} \{A\}P_1\|P_2\{B\} \Leftrightarrow \models_{par} \{A\}P_1'\|P_2\{B\}$$

is not always true. Let us define the deformation index of unitary operator U as

$$D(U)=\inf_{\rho}\frac{\langle 0|U\rho U^{\dagger}|0\rangle}{\langle 0|\rho|0\rangle}.$$

Then we have:

$$\models_{par}\{\lambda\cdot|0\rangle\langle 0|\}P_1\|P_2\{|0\rangle\langle 0|\}\text{ if and only if }\lambda\leq\min\left[D(U),|\langle 0|U|0\rangle|^2\right];\tag{14}$$

$$\models_{par}\{\lambda\cdot|0\rangle\langle 0|\}P_1'\|P_2\{|0\rangle\langle 0|\}\text{ if and only if }\lambda\leq\min\left[D(U),D(V)\cdot|\langle 0|W|0\rangle|^2,|\langle 0|U|0\rangle|^2\right].\tag{15}$$

It is easy to see that the partial correctness in (14) is true but the one in (15) is false when q is a qubit, $\lambda=1$, $U=I$ (the identity) and $V=W=H$ is the Hadamard gate.

As in the case of classical parallel programs, we have to take into account interference between the component programs of a parallel quantum program. The above example further motivates us to introduce a parameterised notion of interference freedom in order to deal with the subtler case of quantum programs. Let us first consider interference between a quantum predicate and a proof outline.

Definition 54 *Let $0\leq\lambda\leq 1$, and let A be a quantum predicate and $\{B\}P^*\{C\}$ a standard proof outline for partial correctness of quantum component program P. We say that A is λ-interference free with $\{B\}P^*\{C\}$ if:*

- *for any atomic region, normal initialisation or unitary transformation Q in P, it holds that*

$$\models_{par}\{\lambda A+(1-\lambda)pre(Q)\}Q\{\lambda A+(1-\lambda)post(Q)\}\tag{16}$$

  where $post(Q)$ is the quantum predicate immediately after Q in $\{B\}P^*\{C\}$;
- *for any normal case statement $Q\equiv\mathbf{if}\ (\square\ M[q]=m\rightarrow Q_m)\ \mathbf{fi}$ in P, it holds that*

$$\models_{par}\{\lambda A+(1-\lambda)pre(Q)\}\ \mathbf{if}\ \ (\square M[q]=m\rightarrow\ \{\lambda A+(1-\lambda)post_m(Q)\}\ Q_m)\ \mathbf{fi}\ \{\lambda A+(1-\lambda)post(Q)\}\tag{17}$$

  where $post_m(Q)$ is the quantum predicate immediately after the mth branch of Q in $\{B\}P^*\{C\}$.

Remark 51 *The reader might be wondering about why $post(Q)$ and $post_m(Q)$ appear in equations (16) and (17). This looks very different from the classical case. When defining interference freedom of A with $\{B\}P^*\{C\}$ for a classical program P, we only require that*

$$\models_{par}\{A\wedge pre(Q)\}Q\{A\}$$

for each basic statement Q in P (see [4], Definition 8.1). Actually, the difference between the classical and quantum cases is not as big as what we think at the first glance. In the classical case, $\models_{par} \{A \wedge pre(Q)\}Q\{A\}$ can be combined with $\models_{par} \{pre(Q)\}Q\{post(Q)\}$, which holds automatically, to yield:

$$\models_{par} \{A \wedge pre(Q)\}Q\{A \wedge post(Q)\}. \tag{18}$$

If conjunctive $\wedge$ in equation (18) is replaced by a convex combination (with probabilities λ and $1-\lambda$), then we obtain equations (16) and (17).

The above defined can be straightforwardly generalised to the notion of interference freedom between a family of proof outlines.

Definition 55 *Let $\{A_i\}P_i^*\{B_i\}$ be a standard proof outline for partial correctness of quantum component program P_i for each $1 \leq i \leq n$.*

1. *If $\Lambda = \{\lambda_{ij}\}_{i\neq j}$ is a family of real numbers in the unit interval, then we say that $\{A_i\}P^*\{B_i\}$ $(i = 1, ..., n)$ are Λ-interference free whenever for any $i \neq j$, each quantum predicate C in $\{A_i\}P_i^*\{B_i\}$ is λ_{ij}-interference free with $\{A_j\}P_j^*\{B_j\}$.*
2. *In particular, $\{A_i\}P^*\{B_i\}$ $(i = 1, ..., n)$ are said to be λ-interference free if they are Λ-interference free for $\Lambda = \{\lambda_{ij}\}_{i\neq j}$ with $\lambda_{ij} \equiv \lambda$ (the same parameter) for all $i \neq j$.*

Inference Rule for Parallel Composition of Quantum Programs with Shared Variables With the notion of interference freedom introduced above, we are able to present a quantum extension (R.PC.L) of inference rule (R.PC) for parallelism with shared variables.

$$\text{(R.PC.L)} \qquad \frac{\text{Standard proof outlines } \{A_i\}\, P_i^*\, \{B_i\}\, (i = 1, ..., n) \text{ are } \Lambda\text{−interference free}}{\{\sum_{i=1}^n p_i A_i\} P_1 \| \cdots \| P_n \{\sum_{i=1}^n p_i B_i\}}$$

Fig. 16. Rule for Parallel Quantum Programs with Shared Variables. $\{p_i\}_{i=1}^n$ is a probability distribution, and $\Lambda = \{\lambda_{ij}\}_{i\neq j}$ satisfies: $\sum_{i\neq j} \frac{p_i}{\lambda_{ij}} \leq 1$ for every j.

Example 53 *Let q_0, q_1, r be three qubit variables and P_i a quantum program with variables q_i and r:*

$$P_i \equiv q_i := |+\rangle; r := |i\rangle; q_i, r := C_X[q_i, r]$$

for $i = 0, 1$, where C_X is the control-NOT gate. Note that P_0 and P_1 have a shared variable r. We are going to use rule (R.PC.L) to show that

$$\left\{\frac{I_r}{2}\right\} P_0 \parallel P_1 \{|0\rangle_r\langle 0|\} \tag{19}$$

where I_r is the identity operator in the state Hilbert space of r. First, we have the proof outline of P_i:

$$\left\{\frac{I_{q_i,r}}{2}\right\} q_i := |+\rangle;\ \{|i\rangle_{q_i}\langle i| \otimes I_r\}\ \ r := |i\rangle;\ \{|00\rangle\langle 00| + |11\rangle\langle 11|\}\ q_i, r := C_X[q_i, r];\ \{I_{q_i} \otimes |0\rangle\langle 0|\}$$

where I_{q_i} is the identity operator in the state space of qubit q_i and $I_{q_i,r}$ the identity operator in the state space of two qubits q_i and r, for $i = 0, 1$, respectively. Moreover, one can easily verify that these two proof outlines are 0.5*-interference free. Then (19) is derived from (R.PC.L) with $p_0 = p_1 = 0.5$.*

Remark 52 *In classical computing, as proved in [33], rule (R.PC) together with a rule for auxiliary variables and Hoare logic for sequential programs gives rise to a (relative) complete logical system for reasoning about parallel programs with shared variables. Obviously, it is not the case for rule (R.PC.L) in parallel quantum programming because not every (largely entangled) precondition (resp. postcondition) of $P_1\|\cdots\|P_n$ can be written in the form of $\sum_{i=1}^n p_iA_i$ (resp. $\sum_{i=1}^n p_iQ_i$). On the other hand, we believe that (R.PC.L) is strong enough to derive a large class of useful correctness properties of parallel quantum programs with shared variables. The reason is that in many-body physics, an overwhelming majority of systems of physics interest can be described by local Hamiltonian: $H = \sum_j H_j$, where each H_j is k-local, meaning that it acts over at most k components of the system. It is clear that rule (R.PC.L) can be used to prove correctness of parallel quantum programs with their preconditions and postconditions being local Hamiltonians.*

The strong soundness of inference rule (R.PC.L) (combined with the other rules introduced in this paper) is presented in the following:

Theorem 51 (Strong Soundness for Parallel Quantum Programs with Convex Combination *Let $\{A_i\}P_i^*\{B_i\}$ be a standard proof outline for partial correctness of quantum component program P_i $(i = 1, ..., n)$ and*

$$\langle P_1\|\cdots\|P_n, \rho\rangle \to^* \{|\langle P_{1s}\|\cdots\|P_{ns}, \rho_s\rangle|\}.$$

Then:

1. *for each $1 \leq i \leq n$ and for every s, $P_{is} \equiv at(T_{is}, P_i)$ for some normal subprogram T_{is} of P_i or $P_{is} \equiv \downarrow$; and*
2. *for any probability distribution $\{p_i\}_{i=1}^n$, if $\{A_i\}P_i^*\{B_i\}$ $(i = 1, ..., n)$ are Λ-interference free for some $\Lambda = \{\lambda_{ij}\}_{i\neq j}$ satisfying*

$$\sum_{i\neq j}\frac{p_i}{\lambda_{ij}} \leq 1 \text{ for } j = 1, ..., n; \tag{20}$$

in particular, if they are λ-interference free for some $\lambda \geq 1 - \min_{i=1}^n p_i$, then we have:

$$tr\left[\left(\sum_{i=1}^n p_iA_i\right)\rho\right] \leq \sum_s tr\left[\left(\sum_{i=1}^n p_iB_{is}\right)\rho_s\right]$$

where

$$B_{is} = \begin{cases} B_i & \text{if } P_{is} \equiv \downarrow, \\ pre(T_{is}) & \text{if } P_{is} \equiv at(T_{is}, P_i). \end{cases}$$

Proof. We prove the conclusion by induction on the length l of transition sequence:

$$\langle P_1 \| \cdots \| P_n, \rho\rangle \rightarrow^l \{|\langle P_{1s} \| \cdots \| P_{ns}, \rho_s\rangle|\}.$$

The conclusion is obviously true in the induction basis case of $l = 0$. Now we assume that

$$\langle P_1 \| \cdots \| P_n, \rho\rangle \rightarrow^l \{|\langle P'_{1k} \| \cdots \| P'_{nk}, \rho'_k\rangle|\} \rightarrow \{|\langle P_{1s} \| \cdots \| P_{ns}, \rho_s\rangle|\}$$

and the last step is a transition performed by the rth component ($1 \leq r \leq n$):

$$\langle P'_{rk}, \rho'_k\rangle \rightarrow \left\{|\langle Q^{(h)}_{rk}, \rho^{(h)}_k\rangle|\right\} \tag{21}$$

for each k. Then

$$\{|\langle P_{1s} \| \cdots \| P_{ns}, \rho_s\rangle|\} = \bigcup_k \left\{|\langle P'_{1k} \| \cdots \| P'_{(r-1)k} \| Q^{(h)}_{rk} \| P'_{(r+1)k} \| \cdots \| P'_{nk}, \rho^{(h)}_k\rangle|\right\}. \tag{22}$$

By the induction hypothesis for the first l steps, we obtain:

$$tr\left[\left(\sum_{i=1}^n p_i A_i\right)\rho\right] \leq \sum_k tr\left[\left(\sum_{i=1}^n p_i B'_{ik}\right)\rho'_k\right] \tag{23}$$

where

$$B'_{ik} = \begin{cases} B_i & \text{if } P'_{ik} \equiv \downarrow, \\ pre(T'_{ik}) & \text{if } P'_{ik} \equiv at(T'_{ik}, P_i). \end{cases}$$

We set

$$B^{(h)}_{rk} = \begin{cases} B_r & \text{if } Q^{(h)}_{rk} \equiv \downarrow, \\ pre(S^{(h)}_{rk}) & \text{if } Q^{(h)}_{rk} \equiv at(S^{(h)}_{rk}, P_r). \end{cases}$$

Then for each k, by an argument similar to the case of Theorem 31 on transition (21), we can prove that

$$tr\left(B'_{rk}\rho'_k\right) \leq \sum_h tr\left(B^{(h)}_{rk}\rho^{(h)}_k\right). \tag{24}$$

On the other hand, $\{A_i\}P_i^*\{B_i\}$ $(i = 1, ..., n)$ are Λ-interference free. Then for every $i \neq r$, it follows from transition (21) that

$$tr\left[\left(\lambda_{ir} B'_{ik} + (1 - \lambda_{ik}) B'_{rk}\right)\rho_k\right] \leq \sum_h tr\left[\left(\lambda_{ir} B'_{ik} + (1 - \lambda_{ir}) B^{(h)}_{rk}\right)\rho^{(h)}_k\right]. \tag{25}$$

Note that $\sum_{i=1}^{n} p_i = 1$, thus condition (20) implies:

$$p_r - \sum_{i\neq r} \frac{p_i\left(1-\lambda_{ir}\right)}{\lambda_{ir}} \geq 0.$$

Therefore, we have:

$$tr\left[\left(\sum_{i=1}^{n} p_i A_i\right)\rho\right] \leq \sum_k tr\left[\left(\sum_{i=1}^{n} p_i B'_{ik}\right)\rho'_k\right] \tag{26}$$

$$= \sum_k tr\left[\left(\sum_{i\neq r}^{n} p_i B'_{ik} + p_r B'_{rk}\right)\rho'_k\right] \tag{27}$$

$$= \sum_k tr\left\{\left[\sum_{i\neq r}^{n} \frac{p_i}{\lambda_{ir}}\left(\lambda_{ir}B'_{ik} + \left(1-\lambda_{ir}\right)B'_{rk}\right) + \left(p_r - \sum_{i\neq r}\frac{p_i\left(1-\lambda_{ir}\right)}{\lambda_{ir}}\right)B'_{rk}\right]\rho'_k\right\} \tag{28}$$

$$\leq \sum_k \left\{\sum_{i\neq r}^{n} \frac{p_i}{\lambda_{ir}} \sum_h tr\left[\left(\lambda_{ir}B'_{ik} + \left(1-\lambda_{ir}\right)B_{rk}^{(h)}\right)\rho_k^{(h)}\right] + \left(p_r - \sum_{i\neq r}\frac{p_i\left(1-\lambda_{ir}\right)}{\lambda_{ir}}\right)\sum_h B_{rk}^{(h)}\rho_k^{(h)}\right\} \tag{29}$$

$$= \sum_{k,h} tr\left[\left(\sum_{i\neq r} p_i B'_{ik} + p_r B_{rk}^{(h)}\right)\rho_k^{(h)}\right] \tag{30}$$

$$= \sum_s tr\left[\left(\sum_{i=1}^{n} p_i B_{is}\right)\rho_s\right]. \tag{31}$$

Here, (26) comes from equation (23), the first and second part of (29) from (25), (24), respectively, and (31) from (22).

6 Conclusion

This paper initiates the study of parallel quantum programming; more explicitly, it defines operational and denotational semantics of parallel quantum programs and presents several useful inference rules for reasoning about correctness of parallel quantum programs. However, this is certainly merely one of the first steps toward a comprehensive theory of parallel quantum programming and leaves a series of fundamental problems unsolved. Perhaps, the most important and difficult open problem is to develop a (relatively) *complete* logical system for verification of parallel quantum programs. Based on the results already obtained in this paper, this problem consists of at least the following subproblems:

1. *Auxiliary Variables*: As is well-known in the theory of classical parallel programming (see [4], Chapters 7 and 8, and [14], Chapter 7), to achieve a

(relatively) complete logical system for reasoning about parallel programs, one must introduce auxiliary variables to record the control flow of a program, which, at the same time, should not influence the control flow inside the program. We presented two rules in Subsection 4.2 for introducing auxiliary variables, but they were used to deal with entanglement and cannot record control flows. It seems that auxiliary variables recording control flows are also needed in parallel quantum programming. At this moment, however, we do not have a clear idea about how such auxiliary variables can be defined in the quantum setting.

2. *Infinite-Dimensional State Hilbert Spaces*: In Section 4, we noticed that rule (R.PC.P) and its extension (R.PC.S) are not sufficient to derive correctness of disjoint parallel quantum programs with entangled preconditions and postconditions, and rule (R.S2E) was then introduced to transform entangled predicates into separable ones. It seems that Theorem 41 and Corollary 41 can guarantee that (R.PC.P), (R.PC.S) and (R.S2E) together with certain quantum extension of the rule for auxiliary variables recording control flows and Hoare logic for sequential quantum programs may form a (relatively) complete logical system for verification of disjoint parallel quantum programs with finite-dimensional state Hilbert spaces. But validity of Theorem 41 and Corollary 41 is still unknown in the case of infinite-dimensional state spaces, which will be crucial in developing a logical system for parallel quantum programs with infinite data types like integers and reals.

3. *Parallel Quantum Programs with Shared Variables and Quantum Marginal Problem*: The case of parallel quantum programs with shared variables is much more difficult. As pointed out in Section 5, inference rule (R.PC.L) can be used to prove some useful correctness properties of such quantum programs, but it is far from being the rule for parallel composition in a (relatively) complete logical system for these quantum programs. A possible candidate for the rule that we are seeking is based on the notions of join and margin of operators: let $\mathcal{H} = \bigotimes_{i=1}^{n} \mathcal{H}_i$ and $\mathcal{J}$ be a family of subsets of $\{1, ..., n\}$. For each $J \in \mathcal{J}$, given a positive operator A_J in $\mathcal{H}_J = \bigotimes_{j\in J} \mathcal{H}_j$. If positive operator A in $\mathcal{H}$ satisfies: $A_J = tr_{J^c} A$ for every $J \in \mathcal{J}$, where $J^c = \{1, ..., n\} \setminus J$, then A is called a join of $\{A_J\}_{J\in\mathcal{J}}$, and each A_J is called the margin of A in $\mathcal{H}_J$. With the notion of join, we can conceive that the inference rule needed for parallel composition of quantum programs with shared variables should be some variant of rule (R.PC.J) given in Figure 17. Then a problem for the future research is: whether such a rule

(R.PC.J) $$\frac{\begin{array}{c}\text{Standard proof outlines } \{A_i\}\, P_i^*\, \{B_i\}\ (i = 1, ..., n) \text{ are } \Lambda\text{−interference free} \\ A \text{ is a join of } \{A_i\}, \text{and } B \text{ is a join of } \{B_i\}\end{array}}{\{A\}P_1\| \cdots \|P_n\{B\}}$$

Fig. 17. Rule for Parallel Quantum Programs with Shared Variables.

together with a rule for auxiliary variables recording control flows and Hoare logic for sequential quantum programs can form a (relatively) complete logical system for parallel quantum logic with shared variables? Furthermore, implementing an automatic tool for verification of parallel quantum programs based on (a variant of) rule (R.PC.J) will rely on a breakthrough in finding an algorithmic solution to the following long-standing open problem (listed in [44] as one of the ten most prominent mathematical challenges in quantum chemistry; see also [26]) — *Quantum Marginal Problem*: given a family $\mathcal{J}$ of subsets of $\{1, ..., n\}$, and for each $J \in \mathcal{J}$, given a density operator (mixed state) ρ_J in $\mathcal{H}_J$. Is there a join (global state) of $\{\rho_j\}_{J\in\mathcal{J}}$ in $\mathcal{H}$?

References

1. A. J. Abhari, A. Faruque, M. Dousti, L. Svec, O. Catu, A. Chakrabati, C.-F. Chiang, S. Vanderwilt, J. Black, F. Chong, M. Martonosi, M. Suchara, K. Brown, M. Pedram and T. Brun, *Scaffold: Quantum Programming Language*, Technical Report TR-934-12, Dept. of Computer Science, Princeton University, 2012.
2. D. Aharonov, M. Ganz and L. Magnin, Dining philosophers, leader election and ring size problems, in the quantum setting, *arXiv*:1707.01187, 2017.
3. T. Altenkirch and J. Grattage, A functional quantum programming language, In: *Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science (LICS)*, 2005, pp. 249-258.
4. K. R. Apt, F. S. de Boer and E. -R. Olderog, *Verification of Sequential and Concurrent Programs*, Springer, London 2009.
5. A. Baltag and S. Smets, LQP: The dynamic logic of quantum information, *Mathematical Structures in Computer Science*, 16(2006)491-525.
6. S. L. Braunstein, C. M. Caves, R. Jozsa, N. Linden, S. Popescu and R. Schack, Separability of very noisy mixed states and implications for NMR quantum computing, *Physical Review Letters*, 83(1999)1054.
7. O. Brunet and P. Jorrand, Dynamic quantum logic for quantum programs, *International Journal of Quantum Information*, 2(2004)45-54.
8. R. Chadha, P. Mateus and A. Sernadas, Reasoning about imperative quantum programs, *Electronic Notes in Theoretical Computer Science*, 158(2006)19-39.
9. R. Cleve and J. Watrous, Fast parallel circuits for the quantum Fourier transform, In: Proceedings of the 41st IEEE Annual Symposium on Foundations of Computer Science (FOCS), 2000, pp. 526-536.
10. A. W. Cross, L. S. Bishop, J. A. Smolin and J. M. Gambetta, Open quantum assembly language, *arXiv*: 1707.03429v2, 2017.
11. H. Corrigan-Gibbs, D. J. Wu and D. Boneh, Quantum operating systems, In: Proceedings of the 16th Workshop on Hot Topics in Operating Systems (HotOS), 2017, pp. 76-81.
12. E. D'Hondt and P. Panangaden, Quantum weakest preconditions, *Mathematical Structures in Computer Science*, 16(2006)429-451.
13. Y. Feng, R. Y. Duan, Z. F. Ji and M. S. Ying, Proof rules for the correctness of quantum programs, *Theoretical Computer Science*, 386(2007)151-166.
14. N. Francez, *Program Verification*, Addison-Wesley, 1992.
15. S. Gay, Quantum programming languages: survey and bibliography, *Mathematical Structures in Computer Science*, 16(2006)581-600.
16. S. Gay, R. Nagarajan, and N. Panaikolaou. QMC: A model checker for quantum systems, In: *Proceedings of the 20th International Conference on Computer Aided Verification (CAV)*, 2008, Springer LNCS 5123, pp. 543-547.
17. G. A. Gorelick, *A complete axiomatic system for proving assertions about recursive and non-recursive programs*, Technical Report, Department of Computer Science, University of Toronto, 1975.
18. A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger and B. Valiron, Quipper: A scalable quantum programming language, *Proceedings of the 34th ACM Conference on Programming Language Design and Implementation (PLDI)*, 2013, pp. 333-342.
19. L. Gurvits and H. Barnum, Separable balls around the maximally mixed multipartite quantum states, *Physical Review A* 68(2003) art. no. 042312.
20. D. Harel, *First-Order Dynamic Logic*, LNCS 68, Springer, 1979.

21. I. Hasuo and N. Hoshino, Semantics of higher-order quantum computation via Geometry of Interaction, In: *Proceedings of the 26th IEEE Symposium on Logic in Computer Science (LICS)*, 2011, 237-246.
22. R. Horodecki, P. Horodecki, M. Horodecki and K. Horodecki, Quantum entanglement, *Reviews of Modern Physics*, 81(2009)865-942.
23. M. F. Kaashoek. Parallelism and operating systems, In: *Proceedings of the SOSP History Day*, pp. 10:1-35, 2015.
24. Y. Kakutani, A logic for formal verification of quantum programs, in: *Proceedings of the 13th Asian Computing Science Conference (ASIAN)*, 2009, Springer LNCS 5913, pp. 79-93.
25. J. Kempe, A. Kitaev and O. Regev, The complexity of the local Hamiltonian problem, *SIAM Journal on Computing*, 35(2006)1070-1097.
26. A. A. Klyachko, Quantum marginal problem and N-representability, *Journal of Physics: Conference Series*, 36(2006)72-86.
27. L. Lamport, Proving the correctness of multiprocess programs, *IEEE Transactions on Software Engineering*, 3(1977)125-143.
28. Y. J. Li and M. S. Ying, Algorithmic analysis of termination problems for quantum programs, In: *Proceedings of the 45th ACM Symposium on Principles of Programming Languages (POPL)*, 2018.
29. Y. J. Li, T. Liu, S. L. Wang, N. J. Zhan and M. S. Ying, A theorem prover for quantum Hoare logic and its applications, *http://arxiv.org/pdf/1601.03835.pdf*
30. C. Moore and M. Nilsson, Parallel quantum computation and quantum codes, *SIAM Journal on Computing* 31(2001) 799-815.
31. M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, 2000.
32. B. Ömer, *Structured Quantum Programming*, Ph.D thesis, Technical University of Vienna, 2003.
33. S. Owicki, A consistent and complete deductive system for the verification of parallel programs, In: *Proceedings of the 8th ACM Symposium on Theory of Computation (STOC)*, 1976, pp. 73-86.
34. S. Owicki and D. Gries, An axiomatic proof technique for parallel programs I, *Acta Informatica*, 6(1976) 319-340.
35. M. Pagani, P. Selinger and B. Valiron, Applying quantitative semantics to higher-order quantum computing, In: *Proceedings of the 41st ACM Symposium on Principles of Programming Languages (POPL)*, 2014, pp. 647-658.
36. J. Paykin, R. Rand and S. Zdancewic, QWIRE: a core language for quantum circuits, In: *Proceedings of the 44th ACM Symposium on Principles of Programming Languages (POPL)*, 2017, pp. 846-858.
37. R. Rand, J. Paykin and S. Zdancewic, QWIRE practice: Formal verification of quantum circuits in Coq, In: *Proceedings of the 14th International Conference on Quantum Physics and Logic (QPL)*, 2017.
38. J. W. Sanders and P. Zuliani, Quantum programming, In: *Proceedings of the 5th International Conference on Mathematics of Program Construction (MPC)*, 2000, Springer LNCS 1837, Springer pp. 88-99.
39. P. Selinger, Towards a quantum programming language, *Mathematical Structures in Computer Science* 14, (2004) 527-586.
40. P. Selinger, A brief survey of quantum programming languages, In: *Proc. of 7th International Symposium on Functional and Logic Programming*, 2004, Springer LNCS 2998, pp. 1-6.

41. P. Selinger and B. Valiron, Quantum lambda calculus, In: S. Gay and I. Mackie (eds.), *Semantic Techniques in Quantum Computation*, Cambridge University Press 2009, pp. 135-172.
42. R. S. Smith, M. J. Curtis and W. J. Zeng, A practical quantum instruction set architecture, *arXiv*: 1608.03355v2, 2017.
43. S. Staton, Algebraic effects, linearity, and quantum programming languages, In: *Proceedings of 42nd ACM Symposium on Principles of Programming Languages (POPL)*, 2015, pp. 395-406.
44. F. H. Stillinger, et. al., *Mathematical Challenges from Theoretical/Computational Chemistry*, National Academy Press, 1995.
45. K. Svore, A. Geller, M. Troyer, J. Azariah, C. Granade, B. Heim, V. Kliuchnikov, M. Mykhailova, A. Paz and M. Roetteler, Q#: Enabling scalable quantum computing and development with a high-level DSL, In: *Proceedings of the Real World Domain Specific Languages Workshop 2018*, art. no. 8.
46. S. Tani, H. Kobayashi and K. Matsumoto, Exact quantum algorithms for the leader election problem, *ACM Transactions on Computation Theory* 4(2012) art. no. 1.
47. D. Wecker and K. M. Svore, LIQUi$|\rangle$: A software design architecture and domain-specific language for quantum computing, http://research.microsoft.com/pubs/209634/1402.4467.pdf.
48. M. S. Ying, Floyd-Hoare logic for quantum programs, *ACM Transactions on Programming Languages and Systems*, 33(2011) art no. 19, pp. 1-49.
49. M. S. Ying, *Foundations of Quantum Programming*, Morgan-Kaufmann, 2016.
50. M. S. Ying and Y. Feng, An algebraic language for distributed quantum computing, *IEEE Transactions on Computers*, 58(2009)728-743.
51. M. S. Ying, S. G. Ying and X. D. Wu, Invariants of quantum programs: characterisations and generation, In: *Proceedings of 44th ACM Symposium on Principles of Programming Languages (POPL)*, 2017, pp. 818-832.
52. K. Życzkowski, P. Horodecki, A. Sanpera and M. Lewenstein, Volume of the set of separable states, *Physical Review A*, 58(1998)883-892.

A Basic Properties of Operators in Hilbert Spaces

For convenience of the reader, we review some notions of operators and super-operators and their basic properties that will be used in the remaining parts of this Appendix.

Lemma A1 *Let A, B be observables (i.e. Hermitian operators) in Hilbert space $\mathcal{H}$. Then $A \sqsubseteq B$ if and only if for all density operators in $\mathcal{H}$:*

$$tr(A\rho) \leq tr(B\rho).$$

Definition A1 *Let $\mathcal{E}$ be a quantum operation (i.e. super-operator) in Hilbert space $\mathcal{H}$ with the Kraus representation $\mathcal{E} = \sum_i E_i \circ E_i^\dagger$. Then its (Schrödinger-Heisenberg) dual is the super-operator $\mathcal{E}^*$ defined by*

$$\mathcal{E}^*(A) = \sum_i E_i^\dagger A E_i$$

for any observable A in $\mathcal{H}$.

Lemma A2 *For any quantum operation $\mathcal{E}$, observable A and density operator ρ in $\mathcal{H}$, we have:*

$$tr(A\mathcal{E}(\rho)) = tr(\mathcal{E}^*(A)\rho).$$

In particular, it holds that

$$tr(\mathcal{E}(\rho)) = tr(\mathcal{E}^*(I)\rho),$$

where I is the identity operator in $\mathcal{H}$.

Lemma A3 *1. If A_1, A_2 are positive operators in $\mathcal{H}_1$ and $\mathcal{H}_2$, respectively, then $A_1 \otimes A_2$ is a positive operator in $\mathcal{H}_1 \otimes \mathcal{H}_2$.*
2. For any operators A_1, B_1 in $\mathcal{H}_1$ and A_2, B_2 in $\mathcal{H}_2$, $A_1 \sqsubseteq B_1$ and $A_2 \sqsubseteq B_2$ implies $A_1 \otimes A_2 \sqsubseteq B_1 \otimes B_2$.

B Proof of Theorem 3.1

Suppose that

$$\langle P, \rho\rangle \rightarrow^n \{|\langle P_i, \rho_i\rangle|\}.$$

We proceed by induction on the length n of computation.

► Induction basis: For $n = 0$, $\{|\langle P_i, \rho_i\rangle|\}$ is a singleton $\{|\langle P_1, \rho_1\rangle|\}$ with $P_1 \equiv P$ and $\rho_1 \equiv \rho$. Then we can choose $T_1 \equiv P$ and it holds that $P_1 \equiv at(T_1, P)$. Note that in the proof outline $\{A\}P^*\{B\}$, we have $A \sqsubseteq pre(P) = B_1$. Thus,

$$tr(A\rho) \leq tr(B_1\rho_1) = \sum_i tr(B_i\rho_i).$$

▶ Induction step: Now we assume that

$$\langle P, \rho\rangle \rightarrow^{n-1} \mathcal{A} \rightarrow \mathcal{A}'$$

and the conclusion is true for length $n-1$. Here, we only consider the simple case where the last step is derived by rule (MS1) with $\mathcal{A} = \{|\langle P_i, \rho_i\rangle|\}$ and $\mathcal{A}_{\downarrow} = \{\langle P, \rho\rangle \in \mathcal{A} : P \not\equiv \downarrow\}$ being a singleton $\{|\langle P_{i_0}, \rho_{i_0}\rangle|\}$. (A general case with $\mathcal{A}_{\downarrow}$ having more than one element follows from the fact that rule (MS2) preserves the inequality in clause (2) of Theorem 3.1.) Then we can assume that

$$\mathcal{A}' = \{|\langle P_i, \rho_i\rangle | i \neq i_0|\} \cup \{|\langle Q_j, \sigma_j\rangle|\}$$

where $\langle P_{i_0}, \rho_{i_0}\rangle \rightarrow \{|\langle Q_j, \sigma_j\rangle|\}$ is derived by one of the rules used in Definition 34 except (MS1) and (MS2). Thus, we need to consider the following cases:

Case 1. The last step uses rule (IF$'$). Then P_{i_0} can be written in the following form:

$$P_{i_0} \equiv \textbf{ if } \ (\square m \cdot M[\overline{q}] = m \rightarrow R_m) \ \textbf{fi},$$

and for each j, $Q_j \equiv R_m \equiv at(R_m, P)$ and $\sigma_j = M_m \rho_{i_0} M_m^{\dagger}$ for some m. On the other hand, a segment of the proof outline $\{A\}P^*\{B\}$ must be derived by the following inference:

$$\frac{\{A_m\}R_m^*\{C\} \text{ for every } m}{\left\{\sum_m M_m^{\dagger} A_m M_m\right\} \textbf{ if } \ (\square m \cdot M[\overline{q}] = m \rightarrow \{A_m\}\, R_m^*) \ \textbf{fi}\{C\}}$$

and $B_{i_0} = pre\,(P_{i_0}) \sqsubseteq \sum_m M_m^{\dagger} A_m M_m$, $A_m = pre\,(R_m)$. Therefore,

$$\begin{aligned} tr\,(B_{i_0}\rho_{i_0}) &\leq tr\left(\sum_m M_m^{\dagger} A_m M_m \rho_{i_0}\right) \\ &= \sum_m tr\left(M_m^{\dagger} A_m M_m \rho_{i_0}\right) \\ &= \sum_m tr\left(A_m M_m \rho_{i_0} M_m^{\dagger}\right) \\ &= \sum_j tr\,(pre\,(Q_j)\,\sigma_j)\,. \end{aligned}$$

By the induction hypothesis, we obtain:

$$\begin{aligned} tr(A\rho) &\leq \sum_{i \neq i_0} tr\,(B_i\rho_i) + tr\,(B_{i_0}\rho_{i_0}) \\ &\leq \sum_{i \neq i_0} tr\,(B_i\rho_i) + \sum_j tr\,(pre\,(Q_j)\,\sigma_j)\,. \end{aligned}$$

So, the conclusion is true in this case.

Case 2. The last step uses rule (L$'$). Then P_{i_0} must be in the following form:

$$P_{i_0} \equiv \textbf{ while } M[\overline{q}] = 1 \textbf{ do } R \textbf{ od}$$

and $\{|\langle Q_j, \sigma_j\rangle|\} = \{|\langle Q_0, \sigma_0\rangle, \langle Q_1, \sigma_1\rangle|\}$ with $Q_0 \equiv \mathbf{skip}, \sigma_0 = M_0\rho_{i_0}M_0^\dagger, Q_1 \equiv R; P_{i_0}$ and $\sigma_1 = M_1\rho_{i_0}M_1^\dagger$. A segment of $\{A\}P^*\{B\}$ must be derived by the following inference:

$$\frac{\{D\}R^*\{M_0CM_0^\dagger + M_1DM_1^\dagger\}}{\begin{array}{l}\{M_0CM_0^\dagger + M_1DM_1^\dagger\}\ \mathbf{while}\ M[\overline{q}] = 0\ \mathbf{do}\ \{C\}\ \mathbf{skip}\ \{C\}\\ \qquad\qquad\qquad\qquad\qquad\qquad\quad = 1\ \mathbf{do}\ R^*\{M_0CM_0^\dagger + M_1DM_1^\dagger\}\\ \qquad\qquad\qquad\qquad \mathbf{od}\ \{C\}\end{array}}$$

and $B_{i_0} \sqsubseteq M_0CM_0^\dagger + M_1DM_1^\dagger$. Then $Q_0 \equiv at(\mathbf{skip}, P), pre(Q_0) = C, Q_1 \equiv at(R, P)$ and $pre(Q_1) = D$. It follows that

$$\begin{aligned} tr\left(B_{i_0}\rho_{i_0}\right) &\le tr\left[\left(M_0CM_0^\dagger + M_1DM_1^\dagger\right)\rho_{i_0}\right] \\ &= tr\left(M_0CM_0^\dagger\rho_{i_0}\right) + tr\left(M_1DM_1^\dagger\rho_{i_0}\right) \\ &= tr\left(CM_0^\dagger\rho_{i_0}M_0\right) + tr\left(DM_1^\dagger\rho_{i_0}M_1\right) \\ &= tr\left(pre(Q_0)\sigma_0\right) + tr\left(pre(Q_1)\sigma_1\right). \end{aligned}$$

Furthermore, by the induction hypothesis, we have:

$$\begin{aligned} tr(A\rho) &\le \sum_{i\neq i_0} tr\left(B_i\rho_i\right) + tr\left(B_{i_0}\rho_{i_0}\right) \\ &\le \sum_{i\neq i_0} tr\left(B_i\rho_i\right) + \sum_j tr\left(pre\left(Q_j\right)\sigma_j\right). \end{aligned}$$

Thus, the conclusion is true in this case.

Case 3. The last step uses rule (Sk), (In) or (UT). Similar but easier.

C Proof of Lemma 4.1

We first prove a diamond property for disjoint parallel quantum programs.

Lemma C1 *The operational semantics of disjoint parallel quantum programs enjoys the diamond property:*

– *if $\mathcal{A} \to \mathcal{A}_1$, $\mathcal{A} \to \mathcal{A}_2$ and $\mathcal{A}_1 \neq \mathcal{A}_2$, then $val(\mathcal{A}_1) = val(\mathcal{A}_2)$ and there exists $\mathcal{B}$ such that $\mathcal{A}_1 \to \mathcal{B}$ and $\mathcal{A}_2 \to \mathcal{B}$.*

Proof. We only need to prove for a singleton $\mathcal{A} = \langle P, \rho\rangle$, since this property is preserved by rules (MS1) and (MS2). Assume that P is a parallel composition of n programs $P_1\|\cdots\|P_n$. Suppose that $\mathcal{A} \to \mathcal{A}_1$ results from a transition of the ith component P_i, that is, $\langle P_i, \rho\rangle \to \{|\langle Q_{ik}, \sigma_k\rangle|\}$ and

$$\mathcal{A}_1 = \{|\langle P_1\|\cdots\|P_{i-1}\|Q_{ik}\|P_{i+1}\|\cdots\|P_n, \sigma_k\rangle|\}.$$

Also, suppose that $\mathcal{A} \to \mathcal{A}_2$ results from a transition of the jth component P_j: $\langle P_j, \rho\rangle \to \{|\langle S_{jl}, \theta_l\rangle|\}$ $(j \neq i)$ and

$$\mathcal{A}_2 = \{|\langle P_1\| \cdots \|P_{j-1}\|S_{jl}\|P_{j+1}\| \cdots \|P_n, \theta_l\rangle|\}.$$

If $\mathcal{A}_1$ contains a terminating configuration, i.e. $P_1\| \cdots \|P_{i-1}\|Q_{ik}\|P_{i+1}\| \cdots \|P_n \equiv\downarrow$ for some k, then $P_m \equiv\downarrow$ for all $m \neq i$ and particularly for $m = j$, a contradiction. So, $val(\mathcal{A}_1) = 0$. Similarly, $val(\mathcal{A}_2) = 0$, and thus $val(\mathcal{A}_1) = val(\mathcal{A}_2)$.

Now we constrct $\mathcal{B}$ such that $\mathcal{A}_1 \to \mathcal{B}$ and $\mathcal{A}_2 \to \mathcal{B}$. To this end, we make the transition of the jth component in $\mathcal{A}_1$ and the transition of the ith component in $\mathcal{A}_2$. After that, an element in $\mathcal{A}_1$ becomes some configuration(s) of the form

$$\langle P_1\| \cdots \|P_{i-1}\|Q_{ik}\|P_{i+1}\| \cdots \|P_{j-1}\|R_{jl}\|P_{j+1}\| \cdots \|P_n, \delta_{kl}\rangle \qquad (32)$$

where δ_{kl} is obtained from applying to σ_k the operators that have generated θ_l. And an element in $\mathcal{A}_2$ becomes some configuration(s) of the form

$$\langle P_1\| \cdots \|P_{i-1}\|Q_{ik}\|P_{i+1}\| \cdots \|P_{j-1}\|R_{jl}\|P_{j+1}\| \cdots \|P_n, \eta_{kl}\rangle \qquad (33)$$

where η_{kl} is obtained from applying to θ_l the operators that have generated σ_k. We use $\mathcal{B}_1', \mathcal{B}_2'$ to denote the sets of elements given in equations (32), (33), respectively. Note that $var\,(P_i) \cap var\,(P_j) = \emptyset$. So, an operator in $\mathcal{H}_{P_i}$ and an operator in $\mathcal{H}_{P_j}$ always commute. Therefore, $\mathcal{B}_1' = \mathcal{B}_2'$, and we complete the proof by setting $\mathcal{B} = \mathcal{B}_1' (= \mathcal{B}_2')$.

A confluence property follows from the above diamond property.

Lemma C2 *If $\mathcal{C} \to \mathcal{A}_1 \to \cdots \to \mathcal{A}_k \to \cdots$ and $\mathcal{C} \to \mathcal{B}_1 \to \cdots \to \mathcal{B}_k \to \cdots$, then there are $\mathcal{C}_1, \cdots, \mathcal{C}_k \cdots$ such that*

$$\mathcal{C} \to^* \mathcal{C}_1 \to^* \cdots \to^* \mathcal{C}_k \to^* \cdots$$

and $\mathcal{A}_k \to^* \mathcal{C}_k$, $\mathcal{B}_k \to^* \mathcal{C}_k$ and $val(\mathcal{A}_k) = val(\mathcal{B}_k)$ for every k.

Proof. We construct configuration ensembles $\mathcal{A}_{k,i}$ and $\mathcal{B}_{k,j}$ such that

$$\mathcal{A}_k = \mathcal{A}_{k,0} \to \mathcal{A}_{k,1} \to \cdots \to \mathcal{A}_{k,k-1} \to \mathcal{A}_{k,k} = \mathcal{C}_k,$$

$$\mathcal{B}_k = \mathcal{B}_{k,0} \to \mathcal{B}_{k,1} \to \cdots \to \mathcal{B}_{k,k-1} \to \mathcal{B}_{k,k} = \mathcal{C}_k,$$

and $\mathcal{A}_{k,i} \to \mathcal{A}_{k+1,i}$ and $\mathcal{A}_{k,i} \to \mathcal{A}_{k+1,i}$, for every k and i.

We construct by induction on k. For the case of $k = 1$, it follows immediately from the diamond property (Lemma C1) that $\mathcal{C}_1 = \mathcal{A}_{1,1} = \mathcal{B}_{1,1}$ exists, and $val(\mathcal{A}_{1,0}) = val(\mathcal{A}_1) = val(\mathcal{B}_1) = val(\mathcal{B}_{1,0})$. Now assume that the construction is achieved for k. Then, we successively construct $\mathcal{A}_{k+1,i}$ for $i = 1, \cdots, k$ such that $\mathcal{A}_{k,i} \to \mathcal{A}_{k+1,i}$ and $\mathcal{A}_{k+1,i-1} \to \mathcal{A}_{k+1,i}$ from $\mathcal{A}_{k,i-1} \to \mathcal{A}_{k,i}$ and $\mathcal{A}_{k,i-1} \to \mathcal{A}_{k+1,i-1}$ by applying Lemma C1. Similarly, we construct $\mathcal{B}_{k+1,i}$ such that $\mathcal{B}_{k,i} \to \mathcal{B}_{k+1,i}$ and $\mathcal{B}_{k+1,i-1} \to \mathcal{B}_{k+1,i}$ for $i = 1, \cdots, k$. Then $\mathcal{C}_{k+1} = \mathcal{A}_{k+1,k+1} = \mathcal{B}_{k+1,k+1}$ satisfying $\mathcal{A}_{k+1,k} \to \mathcal{C}_{k+1}$ and $\mathcal{B}_{k+1,k} \to \mathcal{C}_{k+1}$ is constructed from $\mathcal{C}_k = \mathcal{A}_{k,k} \to \mathcal{A}_{k+1,k}$ and $\mathcal{C}_k = \mathcal{B}_{k,k} \to \mathcal{B}_{k+1,k}$.

To prove $val(\mathcal{A}_k) = val(\mathcal{B}_k)$, it suffices to show that $val(\mathcal{A}_{k,i}) = val(\mathcal{B}_{k,i})$ for every k and i, which can be easily achieved by induction on k and i from the construction.

Now ***we are ready to prove*** Lemma 41. For any two computations $\pi_1 = \langle P, \rho\rangle \rightarrow \mathcal{A}_1 \rightarrow \cdots \rightarrow \mathcal{A}_k \rightarrow \cdots$ and $\pi_2 = \langle P, \rho\rangle \rightarrow \mathcal{B}_1 \rightarrow \cdots \rightarrow \mathcal{B}_k \rightarrow \cdots$, we prove that $val(\pi_1) = val(\pi_2)$. If both π_1 and π_2 are infinite, then by Lemma C2 $val(\mathcal{A}_k) = val(\mathcal{B}_k)$ for every k, and thus

$$val(\pi_1) = \lim_{k\rightarrow\infty} val(\mathcal{A}_k) = \lim_{k\rightarrow\infty} val(\mathcal{B}_k) = val(\pi_2).$$

So we can assume that the shorter one between π_1 and π_2, say, π_1 is finite. Let $\mathcal{A}_n$ be the last configuration ensemble of π_1, then $\mathcal{B}_n$ must be a terminating configuration ensemble too, and thus $val(\pi_1) = val(\mathcal{A}_n) = val(\mathcal{B}_n) = val(\pi_2)$. Otherwise, $\mathcal{B}_n$ is not terminating, then $\mathcal{B}_n \neq \mathcal{A}_n$, and using Lemma C2, π_1 can be extended beyond $\mathcal{A}_n$ by some $\mathcal{C}_n$, a contradiction.

D Proof of Lemma 4.2

1. For any input ρ, by Definition 42 we see that π is a computation of $P_1\|\cdots\|P_n$ starting in ρ if and only if it is a computation of $P_{i_1}\|\cdots\|P_{i_n}$ starting in ρ. Then by Definition 43 it follows that

$$[\![P_1\|\cdots\|P_n]\!](\rho) = [\![P_{i_1}\|\cdots\|P_{i_n}]\!](\rho).$$

2. From Lemma 31, there is a unique computation

$$\pi = \langle P_1;\cdots;P_n, \rho\rangle \rightarrow \mathcal{A}_1 \rightarrow \cdots \rightarrow \mathcal{A}_k \rightarrow \cdots$$

of $P_1;\cdots;P_n$ starting in ρ. We note that each configuration in $\mathcal{A}_k$ must be of the form $\langle Q;P_l;\cdots;P_n,\sigma\rangle$ for some $1 \leq l \leq n$. Then we can replace it by the configuration

$$\langle\downarrow\|\cdots\|\downarrow\|Q\|P_l\|\cdots\|P_n,\sigma\rangle$$

and thus obtain configuration ensemble $\mathcal{A}'_k$. It is easy to see that

$$\pi' = \langle P_1\|\cdots\|P_n, \rho\rangle \rightarrow \cdots \mathcal{A}'_1 \rightarrow \cdots \rightarrow \mathcal{A}'_k \rightarrow \cdots$$

is a computation of $P_1\|\cdots\|P_n$ and $val(\pi') = val(\pi)$. By Lemma 41 (Determinism) we know that $[\![P_1\|\cdots\|P_n]\!](\rho)$ is a singleton, then

$$[\![P_1;\cdots;P_n]\!](\rho) = val(\pi') = [\![P_1\|\cdots\|P_n]\!](\rho).$$

E Proof of Lemma 4.4

Before proving Lemma 44, we present a very useful technical lemma which restates the defining inequalities for total and partial correctness (see Definition 25) in a form of Löwner order.

Lemma E1 *1.* $\models_{tot} \{A\}P\{B\}$ *if and only if* $A \sqsubseteq [\![P]\!]^*(B)$.

2. $\models_{par} \{A\}P\{B\}$ *if and only if*

$$A \sqsubseteq [\![P]\!]^*(B) + (I - [\![P]\!]^*(I)),$$

where I is the identity operator in $\mathcal{H}_P$.

Proof. Immediate from Lemma 4.1.2 in [49] and Definition A1 and Lemma A2.

The conditions for total and partial correctness given in the above lemma are often easier to manipulate than their defining inequalities in Definition 25 because the universal quantifier over density operator ρ is eliminated.

Now ***we can prove Lemma 44***. For each i, since

$$var(P_i) \cap \left(\bigcup_{j\neq i} var(P_i)\right) = \emptyset,$$

super-operator $[\![P_i]\!]$ can be written in the following Kraus form:

$$[\![P_i]\!](\rho) = \sum_k (E_{ik} \otimes I_{\overline{i}})\, \rho \left(E_{ik}^\dagger \otimes I_{\overline{i}}\right)$$

for any $\rho \in \bigotimes_{i=1}^n \mathcal{H}_{P_i}$, where $I_{\overline{i}}$ is the identity operator in $\bigotimes_{j\neq i} \mathcal{H}_j$. Then by Proposition 3.3.1(IV) in [49] and the Sequentialisation Lemma we obtain:

$$\begin{aligned}[\![P_1\|\cdots\|P_n]\!](\rho) &= [\![P_1;\cdots;P_n]\!](\rho)\\ &= [\![P_n]\!]\left(\cdots[\![P_2]\!]\left([\![P_1]\!](\rho)\right)\cdots\right)\\ &= \sum_{k_1,\ldots,k_n}\left(\bigotimes_{i=1}^n E_{ik_i}\right)\rho\left(\bigotimes_{i=1}^n E_{ik_i}^\dagger\right).\end{aligned}$$

Consequently, it holds that for any observable B_i in $\mathcal{H}_{P_i}$ $(i = 1, ..., n)$,

$$\begin{aligned}[\![P_1\|\cdots\|P_n]\!]^*\left(\bigotimes_{i=1}^n B_i\right) &= \sum_{k_1,\ldots,k_n}\left(\bigotimes_{i=1}^n E_{ik_i}^\dagger\right)\left(\bigotimes_{i=1}^n B_i\right)\left(\bigotimes_{i=1}^n E_{ik_i}\right)\\ &= \sum_{k_1,\ldots,k_n}\bigotimes_{i=1}^n\left(E_{ik_i}^\dagger B_i E_{ik_i}\right)\\ &= \bigotimes_{i=1}^n\left(\sum_{k_i} E_{ik_i}^\dagger B_i E_{ik_i}\right)\\ &= \bigotimes_{i=1}^n [\![P_i]\!]^*(B_i).\end{aligned} \tag{34}$$

Now assume that $\models_{par} \{A_i\}\, P_i\, \{B_i\}$ $(i = 1, ..., n)$. Then with Lemma E1 we have:

$$A_i \sqsubseteq [\![P_i]\!]^*(B_i) + (I_i - [\![P_i]\!]^*(I_i)) \ \ (i = 1, ..., n)$$

where I_i is the identity operator in $\mathcal{H}_{P_i}$. To simplify the presentation, we write F_i for $[\![P_i]\!]^*(I_i)$. Note that $B_i \sqsubseteq I_i$ and

$$[\![P_i]\!]^*(B_i) \sqsubseteq [\![P_i]\!]^*(I_i) = F_i.$$

Then using Lemma A3, we obtain:

$$\begin{aligned}
\bigotimes_{i=1}^{n} A_i \sqsubseteq{}& \bigotimes_{i=1}^{n} \left[[\![P_i]\!]^*(B_i) + (I_i - [\![P_i]\!]^*(I_i))\right] \\
&= \bigotimes_{i=1}^{n} [\![P_i]\!]^*(B_i) + \sum_{1\le i_1<\cdots<i_k\le n\ (k\ge 1)} \left[\left(\bigotimes_{i\notin\{i_1,\dots,i_k\}} [\![P_i]\!]^*(B_i)\right) \otimes \left(\bigotimes_{i\in\{i_1,\dots,i_k\}} (I_i - F_i)\right)\right] \\
&\sqsubseteq \bigotimes_{i=1}^{n} [\![P_i]\!]^*(B_i) + \sum_{1\le i_1<\cdots<i_k\le n\ (k\ge 1)} \left[\left(\bigotimes_{i\notin\{i_1,\dots,i_k\}} F_i\right) \otimes \left(\bigotimes_{i\in\{i_1,\dots,i_k\}} (I_i - F_i)\right)\right] \\
&= \bigotimes_{i=1}^{n} [\![P_i]\!]^*(B_i) + \left(\bigotimes_{i=1}^{n} I_i - \bigotimes_{i=1}^{n} F_i\right).
\end{aligned}$$

It follows from equation (34) that

$$\bigotimes_{i=1}^{n} F_i = [\![P_1 \| \cdots \| P_n]\!]^* \left(\bigotimes_{i=1}^{n} I_i\right)$$

and thus

$$\bigotimes_{i=1}^{n} A_i \sqsubseteq [\![P_1 \| \cdots \| P_n]\!]^* \left(\bigotimes_{i=1}^{n} B_i\right) + \left[I - [\![P_1 \| \cdots \| P_n]\!]^*(I)\right].$$

Therefore, with Lemma E1 we assert that

$$\models_{par} \left\{\bigotimes_{i=1}^{n} A_i\right\} P_1 \| \cdots \| P_n \left\{\bigotimes_{i=1}^{n}\right\}.$$

F Proof of Lemmas 4.5 and 4.6

1. We first prove that axiom (Ax.Inv) is sound for partial correctness. Since $var(P) \cap V = \emptyset$, $[\![P]\!]$ can be seen as a super-operator $\mathcal{E}$ in $\mathcal{H}_{V^c}$. Then when considering $[\![P]\!]$ as a super-operator in $\mathcal{H}$, we have:

$$\begin{aligned}
[\![P]\!]^*(A) + [I - [\![P]\!]^*(I)] &= B \otimes \mathcal{E}^*(I_{V^c}) + [I - I_V \otimes \mathcal{E}^*(I_{V^c})] \\
&= B \otimes \mathcal{E}^*(I_{V^c}) + I_V \otimes [I_{V^c} - \mathcal{E}^*(I_{V^c})] \\
&\sqsupseteq B \otimes \mathcal{E}^*(I_{V^c}) + B \otimes [I_{V^c} - \mathcal{E}^*(I_{V^c})] \\
&= B \otimes I_{V^c} = A
\end{aligned}$$

because $B \sqsubseteq I_{V^c}$ and $\mathcal{E}^*(I_{V^c}) \sqsubseteq I_{V^c}$, where I stands for the identity operator in $\mathcal{H}$.

2. Now we prove the soundness of rule (R.TI) for partial correctness. Assume that $\models_{par} \{A\}P\{B\}$. Then it holds that

$$A \sqsubseteq [\![P]\!]^*(B \otimes I_W) + (I - [\![P]\!]^*(I))$$

where I is the identity operator in $\mathcal{H}_{V\cup W}$. Note that $var(P) \subseteq V$. Then we have:

$$tr_W([\![P]\!]^*(B \otimes I_W)) = [\![P]\!]^*(B)$$

where the occurrences of $[\![P]\!]$ in the left-hand side and right-hand side are seen as a super-operator in $\mathcal{H}_{V\cup W}$ and one in $\mathcal{H}_V$, respectively. Similarly, we have:

$$tr_W([\![P]\!]^*(I)) = [\![P]\!]^*(I_V).$$

Therefore, it follows that

$$\begin{aligned} tr_W A &\sqsubseteq tr_W[\![P]\!]^*(B \otimes I_W) + tr_W(I - [\![P]\!]^*(I)) \\ &= [\![P]\!]^*(B) + (I_V - [\![P]\!]^*(I_V)) \end{aligned}$$

and $\models_{par} \{tr_W A\}\, P\{B\}$.

3. We prove the soundness of (R.CC) for partial correctness. If for every $i = 1, ..., m$, $\models_{par} \{A_i\}P\{B_i\}$, then by Lemma E1 we have:

$$A_i \sqsubseteq [\![P]\!]^*(B_i) + [I - [\![P]\!]^*(I)].$$

Consequently, it holds that

$$\begin{aligned} \sum_{i=1}^m p_i A_i &\sqsubseteq \sum_{i=1}^m p_j([\![P]\!]^*(A_j) + [I - [\![P]\!]^*(I)]) \\ &= [\![P]\!]^*\left(\sum_{i=1}^m p_i B_i\right) + [I - [\![P]\!]^*(I)] \end{aligned}$$

because $[\![P]\!]^*(\cdot)$ is linear, and $I - [\![P]\!]^*(I)$ is positive. Thus, we have:

$$\models_{par} \left\{\sum_{i=1}^m p_i A_i\right\} P \left\{\sum_{i=1}^m p_i B_i\right\}.$$

4. We prove the soundness of (R.Inv) for partial correctness. From $\models_{par} \{A\}P\{B\}$, i.e.

$$A \sqsubseteq [\![P]\!]^*(B) + (I - [\![P]\!]^*(I)),$$

we derive that

$$\begin{aligned} pA + qC &\sqsubseteq p[\![P]\!]^*(B) + p(I - [\![P]\!]^*(I)) \\ &= (p[\![P]\!]^*(B) + qC) + p(I - [\![P]\!]^*(I)) \\ &\sqsubseteq (p[\![P]\!]^*(B) + qC) + (I - [\![P]\!]^*(I)) \\ &= [\![P]\!]^*(pB + qC) + (I - [\![P]\!]^*(I)) \end{aligned}$$

because $p \leq 1$ and $V \cap var(P) = \emptyset$ implies $[\![P]\!]^*(C) = C$. Therefore, we have $\models_{par} \{pA + qC\}P\{pB + qC\}$.

5. We prove that rule (R.SO) is sound for partial correctness. Suppose that $\models_{par} \{A\}P\{B\}$, i.e.

$$A \sqsubseteq [\![P]\!]^*(B) + (I - [\![P]\!]^*(I)).$$

Note that $\mathcal{E}$ is a super-operator in $\mathcal{H}_V$ and $V \cap var(P) = \emptyset$. Then $\mathcal{E}^*$ and $[\![P]\!]^*$ commutes, i.e.

$$\mathcal{E}^* \circ [\![P]\!]^* = [\![P]\!]^* \circ \mathcal{E}^*.$$

Moreover, it holds that $\mathcal{E}^*(I) \sqsubseteq I$ and thus

$$(\mathcal{I} - [\![P]\!]^*)\,(\mathcal{E}^*(I)) \sqsubseteq (\mathcal{I} - [\![P]\!]^*)\,(I)$$

where $\mathcal{I}$ is the identity super-operator. Therefore, we obtain:

$$\begin{aligned}\mathcal{E}^*(A) &\sqsubseteq \mathcal{E}^*\left[[\![P]\!]^*(B) + (I - [\![P]\!]^*(I))\right] \\ &= \mathcal{E}^*\left([\![P]\!]^*(B)\right) + \left[\mathcal{E}^*(I) - \mathcal{E}^*\left([\![P]\!]^*(I)\right)\right] \\ &= [\![P]\!]^*\left(\mathcal{E}^*(B)\right) + (\mathcal{I} - [\![P]\!]^*)\left(\mathcal{E}^*(I)\right) \\ &\sqsubseteq [\![P]\!]^*\left(\mathcal{E}^*(B)\right) + (\mathcal{I} - [\![P]\!]^*)\,(I) \\ &= [\![P]\!]^*\left(\mathcal{E}^*(B)\right) + (I - [\![P]\!]^*(I))\end{aligned}$$

and $\models_{par} \{\mathcal{E}^*(A)\}P\{\mathcal{E}^*(B)\}$.

6. Finally, we prove the soundness of (R.Lim) for partial correctness. With the assumption that $\{A_n\}$ is increasing and $\{B_n\}$ decreasing accordiung to the Löwner order, the existence of $\lim_{n\to\infty} A_n$ and $\lim_{n\to\infty} B_n$ is guaranteed by Lemma 4.1.3 in [49]. Assume that $\models_{par} \{A_n\}P\{B_n\}$. Then

$$A_n \sqsubseteq [\![P]\!]^*(B_n) + [I - [\![P]\!]^*(I)]$$

and the continuity of super-operator $[\![P]\!]^*$ yields:

$$\begin{aligned}A = \lim_{n\to\infty} A_n &\sqsubseteq \lim_{n\to\infty} \left([\![P]\!]^*(B_n) + [I - [\![P]\!]^*(I)]\right) \\ &= [\![P]\!]^*(\lim_{n\to\infty} B_n) + [I - [\![P]\!]^*(I)] \\ &= [\![P]\!]^*(B) + [I - [\![P]\!]^*(I)]\,.\end{aligned}$$

Thus, $\models_{par} \{A\}P\{B\}$.

G Proof of Lemma 4.7

1. We first consider the total correctness. Assume that

$$\models_{tot} \{(1-\epsilon)I + \epsilon A\}\,P\,\{(1-\epsilon)I + \epsilon B\}\,.$$

Then by Lemma E1 we obtain:

$$\begin{aligned}(1-\epsilon)I + \epsilon A &\sqsubseteq [\![P]\!]^*((1-\epsilon)I + \epsilon P) \\ &= (1-\epsilon)[\![P]\!]^*(I) + \epsilon[\![P]\!]^*(B)\end{aligned} \tag{35}$$

because $[\![P]\!]$ is linear. Note that for any super-operator $\mathcal{E} = \sum_i E_i \circ E_i^\dagger$, it holds that

$$\mathcal{E}^*(I) = \sum_i E_i^\dagger E_i \sqsubseteq I.$$

Therefore, we have:

$$(1-\epsilon)[\![P]\!]^*(I) \sqsubseteq (1-\epsilon)I$$

since $\epsilon \leq 1$. Consequently, it follows from equation (35) that $\epsilon A \sqsubseteq \epsilon[\![P]\!]^*(B)$ and $A \sqsubseteq [\![P]\!]^*(B)$ because $\epsilon > 0$. So, we otain $\models_{tot} \{A\}P\{B\}$.

2. Now we consider the partial correctness. Let

$$\models_{par} \{(1-\epsilon)I + \epsilon A\}\, P\, \{(1-\epsilon)I + \epsilon B\}\,.$$

Then it follows from Lemma E1 that

$$\begin{aligned}(1-\epsilon)I + \epsilon A &\sqsubseteq [\![P]\!]^*((1-\epsilon)I + \epsilon P) + [I - [\![P]\!]^*(I)] \\ &= (1-\epsilon)[\![P]\!]^*(I) + \epsilon[\![P]\!]^*(B) + [I - [\![P]\!]^*(I)]\,.\end{aligned}$$

Since $\epsilon > 0$, a routine calculation yields:

$$A \sqsubseteq [\![P]\!]^*(B) + [I - [\![P]\!]^*(I)]\,,$$

and thus $\models_{par} \{A\}P\{B\}$.