Algorithmic Analysis of Termination Problems for Quantum Programs

YANGJIA LI, Institute of Software, Chinese Academy of Sciences, China
MINGSHENG YING, University of Technology Sydney, Australia, Institute of Software, Chinese Academy of Sciences, China, and Tsinghua University, China

We introduce the notion of linear ranking super-martingale (LRSM) for quantum programs (with nondeterministic choices, namely angelic and demonic choices). Several termination theorems are established showing that the existence of the LRSMs of a quantum program implies its termination. Thus, the termination problems of quantum programs is reduced to realisability and synthesis of LRSMs. We further show that the realisability and synthesis problem of LRSMs for quantum programs can be reduced to an SDP (Semi-Definite Programming) problem, which can be settled with the existing SDP solvers. The techniques developed in this paper are used to analyse the termination of several example quantum programs, including quantum random walks and quantum Bernoulli factory for random number generation. This work is essentially a generalisation of constraint-based approach to the corresponding problems for probabilistic programs developed in the recent literature by adding two novel ideas: (1) employing the fundamental Gleason's theorem in quantum mechanics to guide the choices of templates; and (2) a generalised Farkas' lemma in terms of observables (Hermitian operators) in quantum physics.

CCS Concepts: • **Theory of computation → Program analysis**;

Additional Key Words and Phrases: Quantum programming, termination, ranking function, super-martingale, SDP (Semi-Definite Programming), quantum random walk, quantum Bernoulli factory

ACM Reference Format:
Yangjia Li and Mingsheng Ying. 2018. Algorithmic Analysis of Termination Problems for Quantum Programs. *Proc. ACM Program. Lang.* 2, POPL, Article 35 (January 2018), 29 pages. https://doi.org/10.1145/3158123

1 INTRODUCTION

Quantum Programming: Quantum programming research has been extensively conducted for two decades, including design and implementation of quantum programming languages as well as their operational and denotational semantics and type systems. For example, the first quantum programming language QCL was presented by Ömer [2003], a quantum extension qGCL of Dijkstra's guarded-command language GCL was proposed by Sanders and Zuliani [2000], a model of quantum computing embedded in Haskell was developed by Sabry [2003], and the first two quantum languages QPL and QML of the functional programming paradigm were defined by Selinger [2004b] and Altenkirch and Grattage [2005], respectively. In the last few years, several more practical and scalable quantum programming languages have been defined and their compilers have been implemented, including Quipper [Green et al. 2013], Scaffold [JavadiAbhari et al. 2012],

Authors' addresses: Yangjia Li, State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, 100190, China, yangjia@ios.ac.cn; Mingsheng Ying, University of Technology Sydney, Sydney, Australia, Mingsheng.Ying@uts.edu.au, State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, 100190, China , Tsinghua University, Beijing, 100084, China.

2475-1421/2018/1-ART35
https://doi.org/10.1145/3158123

Microsoft's LIQUi|⟩ [Wecker and Svore 2014] and QWIRE [Paykin et al. 2017]. For a detailed survey, see [Gay 2006; Seinger 2004a; Ying 2016].

Verification of Quantum Programs: Various program logics and model-checking techniques have also been extended for verification of quantum programs [Baltag and Smets 2006; Brunet and Jorrand 2004; Chadha et al. 2006; Feng et al. 2007, 2013; Gay et al. 2008; Kakutani 2009; Ying et al. 2014a] . For example, a Hoare-like logic for reasoning about both partial and total correctness of quantum programs was developed and its (relative) completeness was proved by Ying [2011]. A theorem prover was implemented by Liu et al. [2016] for quantum Hoare logic based on Isabelle/HOL. The notion of invariants for quantum programs was introduced, and an SDP (Semi-Definite Programming) algorithm for generating invariants of quantum programs was presented by Ying et al. [2017].

Termination Analysis of Quantum Programs: The termination problems of quantum programs have also received a fair amount of attention. Expected running time of quantum programs written in Scaffold was considered and its implications to the compiler ScaffCC of Scaffold was discussed by JavadiAbhari et al. [2015]. Termination of quantum **while**-loops with a unitary transformation as the loop body was studied by Ying and Feng [2010] using Jordan decomposition of complex matrices, and several major results in [Ying and Feng 2010] were extended by Ying et al. [2013a] to quantum loops where the loop body can be a general super-operator. Furthermore, termination of nondeterministic and concurrent quantum programs was examined by Li et al. [2015] and Yu and Ying [2012] as a reachability problem of quantum Markov systems.

Synthesis of Ranking Functions: The idea of using ranking function in proving termination of *classical* programs is traced back to the seminal paper [Floyd 1967]. Recently, this idea has been generalised to *probabilistic* programs. For example, a ranking function of the Lyapunov style was employed by Bournez and Garnier [2005] to prove finite termination of probabilistic programs without nondeterministic choices. The notion of ranking super-martingale was introduced by Chakarov and Sankaranarayanan [2013] and Fioriti and Hermanns [2015] for proving almost-sure and finite termination of probabilistic programs without and with nondeterminism, respectively.

One of the main techniques for synthesis of ranking functions and super-martingales is the *constraint-based* approach, of which the basic idea is to use a *template* of ranking functions or super-martingales that is constrained by the desired termination properties and then to find a solution to the constraint system. This approach was originally proposed by Colón et al. [2001, 2003] and Podelski and Rybalchenko [2004] for invariant generation and synthesis of ranking function for classical programs; for a series of successful examples, see [Rybalchenko 2010]. In the last few years, it has been extended by Chakarov and Sankaranarayanan [2013] and Chatterjee et al. [2016] to synthesis of ranking super-martingales of probabilistic programs.

Contributions of This Paper: The aim of this paper is to extend the constraint-based approach to synthesis of ranking functions for termination analysis of quantum programs (with angelic and demonic choices). Our contributions includes:

- The notion of ranking function for quantum programs was already introduced in quantum Hoare logic [Ying 2011] in order to guarantee termination for total correctness of quantum programs. In this paper, we define the notion of *ranking super-martingale (LRSM)* for quantum programs as a generalisation of ranking functions. Then we establish two *termination theorems* showing that the existence of the LRSMs of a quantum program guarantees its termination.
- As mentioned above, the constrain-based approach was already used in [Ying et al. 2017] where an SDP (Semi-Definite Programming) algorithm was found for generating invariants of quantum programs. In this paper, we prove that the *realisabilty and synthesis problem*

of LRSMs for quantum programs with angelic and demonic choices can be reduced to an SDP problem. The novelty of this work are: (i) A fundamental theorem, namely Gleason's theorem [Dvurečenskij 1993; Gleason 1957] from quantum mechanics, is employed to guide the choice of templates of LRSMs; and (ii) The reduction of synthesising LRSMs of *quantum programs with angelic choices* requires a new generalisation of Farkas's lemma in the form of observables in quantum physics (modelled as Hermitian operators), which does not appear in the previous literature.

- As applications of the techniques developed in this paper, we present several case studies of the termination analysis of example quantum programs, including quantum random walks and quantum Bernoulli factory [Dale et al. 2015] for randomness generation and processing, by using the existing SDP solvers.

Organisation of The Paper: We assume that the reader is familiar with the basics of quantum theory, which can be found in previous literature on quantum programming like [Selinger 2004b] and the recent book [Ying 2016]. In Section 2, we define the quantum programming language used in the paper. The notion of quantum game structure is introduced in Section 3 as a semantic model of quantum programs. In Section 4, we define the termination problems for quantum programs. In Section 5, we introduce the notion of LRSM and prove two termination theorems. The reduction of realisability and synthesis problem of LRSMs for quantum programs to SDP is presented in Section 6. The case studies are given in Section 7. A brief conclusion is drawn in Section 8.

2 QUANTUM PROGRAMS

In this section, we define the quantum programming language used in this paper. It is obtained by adding angelic and demonic choices to the quantum extension of **while**-language studied in [Ying 2011, 2016].

2.1 Syntax of Quantum Programs

We assume a set Var of quantum variables. For each $q \in Var$, its state Hilbert space is denoted by $\mathcal{H}_q$.

Definition 2.1 (Syntax). *Quantum programs are defined by the following grammar:*

$$\begin{aligned} P ::= \ & \mathbf{skip} \mid P_1; P_2 \mid P_1 \sqcup P_2 \mid P_1 \sqcap P_2 && (1)\\ & \mid q := |0\rangle && (2)\\ & \mid \overline{q} := U[\overline{q}] && (3)\\ & \mid \mathbf{if}\ (\square_m\, M[\overline{q}] = m \to P_m)\ \mathbf{fi} && (4)\\ & \mid \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ P\ \mathbf{od} && (5) \end{aligned}$$

Let us give a brief explanation to the program constructs introduced in the above definition. Variable $q \in Var$ and $\overline{q} \subseteq Var$. The constructs in (1) are similar to their counterparts in a classical or probabilistic programming language: $P_1; P_2$, $P_1 \sqcup P_2$, $P_1 \sqcap P_2$ stand for the sequential composition, angelic choice and demonic choice, respectively. The initialisation (2) sets quantum variable q to a basis state $|0\rangle$. The statement (3) means that unitary transformation U is applied to register $\overline{q}$, leaving the states of the variables not in $\overline{q}$ unchanged. The construct (4) is a quantum generalisation of classical case statement. In executing it, measurement $M = \{M_m\}$ is performed on $\overline{q}$, and then a subprogram P_m is selected to be executed next according to the outcome of measurement. We assume that the set $\{m\}$ of measurement outcomes is finite or countably infinite. An essential difference between (4) and a classical case statement is that the state of program variables is changed after performing the measurement in the former, whereas it is not changed after checking the

guards in the latter. The statement (5) is a quantum generalisation of **while**-loop. The measurement in (5) has only two possible outcomes 0, 1. If the outcome 0 is observed, then the program terminates, and if the outcome 1 occurs, the program executes the subprogram P and continues. The only difference between quantum loop (5) and a classical loop is that checking the loop guard in the latter does not change the state of program variables, but it is not the case in the former.

Remark 2.1. *It is worth noting that the angelic and demonic choices $\sqcup, \sqcap$ in (1) are classical rather than quantum features. As in classical programming, they are introduced mainly for specifying and reasoning about the behaviour of a program without having to consider its details of implementation. The usefulness of nondeterministic choices in quantum computation was first noticed by Zuliani [2004], where a nondeterministic choice was added into quantum programming language qGCL and used to formalise counterfactual computation [Mitchison and Jozsa 2001] and quantum systems in mixed states.*

2.2 Illustrative Examples

To illustrate the program constructs defined in Definition 2.1, we present three simple examples of quantum programs. Their termination will be considered in Section 7 as case studies of the techniques developed in this paper.

Let us first consider a quantum program without angelic and demonic choices, namely a quantum walk on a circle with an absorbing boundary. Its termination was analysed first in [Ying et al. 2013a] by a direct matrix analysis (rather than a ranking function as we do in this paper).

Example 2.1. *(Quantum walk on an n-circle with an absorbing boundary at position 1) Let $\mathcal{H}_c$ be the coin space, the 2-dimensional Hilbert space with orthonormal basis state $|L\rangle$ and $|R\rangle$, indicating directions Left and Right, respectively. Let $\mathcal{H}_p$ be the n-dimensional Hilbert space with orthonormal basis states $|0\rangle, |1\rangle, \ldots, |n-1\rangle$, where vector $|i\rangle$ denotes position i for each $0 \leq i < n$. The state space of the walk is then $\mathcal{H} = \mathcal{H}_c \otimes \mathcal{H}_p$. The initial state is assumed to be $|L\rangle|0\rangle$. Each step of the walk consists of:*

(1) *Measure the position of the system to see whether it is 1. If the outcome is "yes", then the walk terminates; otherwise, it continues. The measurement can be described as $M = \{M_{yes}, M_{no}\}$, where the measurement operators are:*

$$M_{yes} = |1\rangle\langle 1|, \ M_{no} = I_p - M_{yes} = \sum_{n \neq 1} |n\rangle\langle n|$$

and I_p is the identity operator in the position space $\mathcal{H}_p$;

(2) *The Hadamard "coin-tossing" operator*

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

is applied in the direction space $\mathcal{H}_c$;

(3) *A shift operator*

$$S = \sum_{i=0}^{n-1} |L\rangle\langle L| \otimes |i \ominus 1\rangle\langle i| + \sum_{i=0}^{n-1} |R\rangle\langle R| \otimes |i \oplus 1\rangle\langle i|$$

is performed on the space $\mathcal{H}$. Here, $\oplus$ and $\ominus$ stand for addition and subtraction modulo n, respectively.

Intuitively, operator S means that the system walks one step left or right according to the direction state. A major difference between a quantum walk and a classical random walk is that a superposition

of movement to the left and a movement to the right can happen in the former; for example, the coin is in one of the following states:

$$\frac{1}{\sqrt{2}}(|L\rangle + |R\rangle), \qquad \frac{1}{\sqrt{2}}(|L\rangle - |R\rangle)$$

This walk can be written as the quantum program QW*:*

$$\begin{aligned} &c := |L\rangle; p := |0\rangle; \\ &\textbf{while } M[p] = \mathit{no} \textbf{ do } c := H[c]; c, p := S[c, p] \textbf{ od} \end{aligned}$$

Next we consider a more complicated quantum walk with demonic choice:

Example 2.2. *(Demonic quantum walk) Let quantum variables c, p and their state Hilbert spaces be the same as in Example 2.1. The quantum walk is written as the program* DQW*:*

$$\begin{aligned} &c := |L\rangle; p := \frac{1}{\sqrt{3}}(|0\rangle + |1\rangle + |2\rangle) \\ &\textbf{while } M[p] = \mathit{no} \textbf{ do if } N[p] = \mathit{yes} \ \rightarrow W_1 \sqcap W_2 \\ &\qquad\qquad\qquad\qquad \square \qquad = \mathit{no} \ \rightarrow W_1 \sqcap W_3 \\ &\qquad\qquad\qquad\qquad \textbf{fi od} \end{aligned}$$

where measurement M is as in Example 2.1, $N = \{N_{yes}, N_{no}\}$ is the measurement to see whether the position is 2, i.e.

$$N_{yes} = |2\rangle\langle 2|, \ N_{no} = I_p - N_{yes} = \sum_{n \neq 2} |n\rangle\langle n|$$

and subprograms W_1, W_2 are given as follows:

$$\begin{aligned} W_1 &\equiv c := H[c]; c, p := S[c, p] \\ W_2 &\equiv c := Y[c]; c, p := S[c, p] \\ W_3 &\equiv c := Z[c]; c, p := S[c, p] \end{aligned}$$

Operator W_1 is indeed the same as the single-step walk in Example 2.1, but W_2, W_3 use the Pauli operators Y, Z, respectively, rather than the Hadamard operator H as their coin-tossing operator. The Pauli operators are given as follows:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

We see in the above example that a demonic choice between using the Hadamard operator and the Pauli operator(s) to toss the coin.

Finally, we consider a quantum program with both angelic and demonic choices:

Example 2.3. *(Angelic and demonic rotations) Let q and c be two qubits. Their state spaces are the 2-dimensional Hilbert space with orthonormal basis states $|0\rangle, |1\rangle$. The program* ADR *is given as follows:*

$$\begin{aligned} &q := |1\rangle; c := |1\rangle; \\ &\textbf{while } M[q] = 1 \textbf{ do } c, q := C(X)[c, q]; q, c := C(H)[q, c]; \\ &\qquad \textbf{if } N[c] = 0 \rightarrow q := R_x(\alpha)[q] \sqcup q := R_z(\beta)[q] \\ &\qquad \square \qquad = 1 \rightarrow q := R_x(\alpha)[q] \sqcap q := R_z(\beta)[q] \\ &\qquad \textbf{fi od} \end{aligned}$$

where:

- *R_x, R_z are rotations around the x and z-axis, respectively, on the Bloch sphere:*

$$R_x(\alpha) = \cos\frac{\alpha}{2} I - i \sin\frac{\alpha}{2} X = \begin{pmatrix} \cos\frac{\alpha}{2} & -i\sin\frac{\alpha}{2} \\ -i\sin\frac{\alpha}{2} & \cos\frac{\alpha}{2} \end{pmatrix},$$

$$R_z(\beta) = \cos\frac{\beta}{2} I - i \sin\frac{\beta}{2} Z = \begin{pmatrix} e^{-i\beta/2} & 0 \\ 0 & e^{i\beta/2} \end{pmatrix}$$

- *For any gate A, $C(A)$ is the controlled A gate:*

$$C(A) = \begin{pmatrix} I & 0 \\ 0 & A \end{pmatrix}$$

with $0, I$ being the 2×2 zero and unit matrices, respectively, and we simply write $C := C(H)C(X)$;

- *$M = \{M_0, M_1\}$, $N = \{N_0, N_1\}$ are both measurements in the computational basis $|0\rangle, |1\rangle$, but performed on two different qubits q and c; that is, $M_0 = |0\rangle_q\langle 0|$, $M_1 = |1\rangle_q\langle 1|$, $N_0 = |0\rangle_c\langle 0|$ and $N_1 = |1\rangle_c\langle 1|$.*

3 QUANTUM GAMES

In this section, we introduce a quantum extension of stochastic games defined by Chatterjee et al. [2016]. Then we use it to serve as a semantic model of quantum programs, which in turn provides us with a basis for studying termination problems and ranking super-martingales of quantum programs.

3.1 Quantum Game Structures

We write $\mathcal{D}(\mathcal{H})$ for the set of partial density matrices (i.e. positive semi-definite matrices with traces ≤ 1), respectively, in a Hilbert space $\mathcal{H}$. In particular, if $\rho \in \mathcal{D}(\mathcal{H})$ and $tr(\rho) = 1$, then ρ is a density operator, denoting a mixed state in $\mathcal{H}$. As in [Selinger 2004b; Ying 2011, 2016; Ying et al. 2017], we use super-operators (completely positive maps between operators) to represent the semantic functions of quantum programs. Two super-operators $\mathcal{E}, \mathcal{F}$ are said to be equivalent [Feng et al. 2013], written $\mathcal{E} \approx \mathcal{F}$, if we have:

$$tr(\mathcal{E}(\rho)) = tr(\mathcal{F}(\rho))$$

for all density operators $\rho \in \mathcal{D}(\mathcal{H})$.

Definition 3.1 (Game Structure). *A quantum game structure in a Hilbert space $\mathcal{H}$ is a 4-tuple $\mathcal{G} = \langle L, l_0, \rho_0, \rightarrow\rangle$, where:*

(1) *L is a finite set of locations partitioned into disjoint subsets L_A, L_D, L_S of angelic, demonic, and standard locations, respectively;*
(2) *$l_0 \in L$ is the initial location;*
(3) *$\rho_0 \in \mathcal{D}(\mathcal{H})$ is a density operator. It is called the initial state of quantum variables;*
(4) *$\rightarrow$ is a transition relation, whose all elements are of the form $l \xrightarrow{\mathcal{E}} l'$, where l, l' are locations, and $\mathcal{E}$ is a super-operator operator in $\mathcal{H}$, satisfying the following conditions:*
   - *if $l \in L_A \cup L_D$ and $l \xrightarrow{\mathcal{E}} l'$, then $\mathcal{E} = \mathcal{I}$ (the identity super-operator in $\mathcal{H}$);*
   - *if $l \in L_S$, then*

$$\sum \{|\mathcal{E} : l \xrightarrow{\mathcal{E}} l' \text{ for some } l'|\} \approx \mathcal{I}. \tag{6}$$

The symbol $\{|\cdot|\}$ in equation (6) stands for multi-set. We always assume that the transition relation $\rightarrow$ is countably branching; that is, for every $l \in L$, the set $\{|\mathcal{E} : l \xrightarrow{\mathcal{E}} l' \text{ for some } l'|\}$ is finite or countably infinite. Therefore, the summation in the left-hand side of equation (6) is well-defined.

Sometimes, we write $l \xrightarrow{E} l'$ for $l \xrightarrow{\mathcal{E}} l'$ whenever $\mathcal{E}$ is defined by an operator E, i.e. $\mathcal{E} = E \circ E^\dagger$ or more precisely, $\mathcal{E}(\rho) = E\rho E^\dagger$ for all density operators ρ. In particular, we write $l \xrightarrow{I} l'$ instead of $l \xrightarrow{I} l'$, where I is the identity operator in $\mathcal{H}$. If for every transition $l \xrightarrow{\mathcal{E}} l'$, super-operator $\mathcal{E}$ is defined by an operator E, then $\mathcal{G}$ is called an *operator-valued game*.

Remark 3.1. *Essentially, a quantum game structure is a super-operator valued Markov chain defined in [Feng et al. 2013; Gudder 2008]. The only difference between them is that locations in a game are partitioned into the angelic, demonic and standard. An operator-valued graph (i.e. game without partition of angelic, demonic and standard locations) is called a quiver in representation theory [Dersen and Weyman 2005].*

A configuration of quantum game structure $\mathcal{G}$ is a pair (l, ρ), where $l \in L$ and $\rho \in \mathcal{D}(\mathcal{H})$.

A path in $\mathcal{G}$ is a finite sequence of transitions

$$\pi = l_1 \xrightarrow{\mathcal{E}_1} l_2 \xrightarrow{\mathcal{E}_2} \cdots \xrightarrow{\mathcal{E}_{n-1}} l_n.$$

We use $\mathcal{E}_\pi$ to denote the composition of the super-operators along the path, i.e. $\mathcal{E}_\pi = \mathcal{E}_{n-1} \circ \cdots \mathcal{E}_2 \circ \mathcal{E}_1$.

Definition 3.2 (Runs). *A run of game $\mathcal{G}$ is a finite or an infinite sequence of configurations*

$$\theta = (l_0, \rho_0), (l_1, \rho_1), (l_2, \rho_2), \ldots$$

(starting from the initial configuration) such that for each $i \geq 0$, we have a transition $l_i \xrightarrow{\mathcal{E}_i} l_{i+1}$ in $\mathcal{G}$ with $\rho_{i+1} = \mathcal{E}_i(\rho_i)$.

The game $\mathcal{G}$ is played between two players, angel and demon, according to their respective schedulers.

Definition 3.3 (Schedulers). (1) *An angelic (resp. a demonic) scheduler is a function that assigns to every finite run ending in a configuration with an angelic (resp. a demonic) location l a transition outgoing from l.*

(2) *Let σ, τ be an angelic and a demonic schedulers, respectively. We say that a run θ satisfies σ and τ if the transitions in θ outgoing from all angelic locations are chosen according to σ, and those outgoing from all demonic locations are chosen according to τ.*

Remark 3.2. *There are two different ways to define a scheduler in a quantum Markov decision processes: the choice of next action depends on either the quantum states occurred previously [Barry et al. 2014] or the outcomes of measurements performed previously [Ying et al. 2014b]. But in achieving termination of quantum programs, their abilities are exactly the same, as will be seen in Section 5. Here, we follow the way used in [Barry et al. 2014], because in this way we only need to consider the angelic schedulers that are independent of histories in the termination proofs.*

3.2 Game Representation of Quantum Programs

Now a quantum program P can be modelled by a quantum game $\mathcal{G}_P$. We write $var(P)$ for the set of quantum variables occurring in P and

$$\mathcal{H}_P = \bigotimes_{q \in var(P)} \mathcal{H}_q$$

for the state Hilbert space of P. Then $\mathcal{G}_P$ can be defined in $\mathcal{H}_P$ by induction on the length of P. This game has two designated locations l^P_{in}, l^P_{out}, with the latter being a special location that has no outgoing transitions. Its initial location is l^P_{in}. We assume that the initial state ρ_0 is specified in the preamble of the program P.

- $P \equiv \textbf{skip}$. Then $\mathcal{G}_P$ has only two locations l_{in}^P, l_{out}^P and a single transition $l_{in}^P \xrightarrow{I} l_{out}^P$, where both l_{in}^P and l_{out}^P are standard locations;
- $P \equiv q := |0\rangle$. Let $\{|n\rangle\}$ be an orthonormal basis of $\mathcal{H}_q$. Then $\mathcal{G}_P$ has locations l_{in}^P, l_{out}^P together with l_n for each basis state $|n\rangle$. All of them are standard locations. The transitions are $l_{in}^P \xrightarrow{E_n} l_n$ and $l_n \xrightarrow{I} l_{out}^P$ for every basis state $|n\rangle$, where $E_n = |0\rangle\langle n|$.
- $P \equiv P_1; P_2$. Suppose that $\mathcal{G}_{P_1}, \mathcal{G}_{P_2}$ are the games for subprograms P_1, P_2, respectively. Then $\mathcal{G}_P$ is constructed as follows: we identify $l_{out}^{P_1} = l_{in}^{P_2}$, and it is deemed to be angelic, demonic or standard if so is $l_{in}^{P_2}$. We further concatenate P_1 and P_2, and put $l_{in}^P = l_{in}^{P_1}$ and $l_{out}^P = l_{out}^{P_2}$;
- $P \equiv \textbf{if } (\square_m M[\overline{q}] = m \rightarrow P_m) \textbf{ fi}$. Suppose that $\mathcal{G}_{P_m}$ is the game for subprogram P_m for every m. Then $\mathcal{G}_P$ is constructed as follows: we add a new standard location l_{in}^P and a transition $l_{in}^P \xrightarrow{M_m} l_{in}^{P_m}$ for every m. Furthermore, we identify $l_{out}^P = l_{out}^{P_m}$ for all m;
- $P \equiv \textbf{while } M[\overline{q}] = 1 \textbf{ do } Q \textbf{ od}$. We construct $\mathcal{G}_P$ from the game $\mathcal{G}_Q$ for subprogram Q as follows: we add two new standard locations l_{in}^P, l_{out}^P and two transitions $l_{in}^P \xrightarrow{M_0} l_{out}^P$, $l_{in}^P \xrightarrow{M_1} l_{in}^Q$. We identify $l_{out}^Q = l_{in}^P$.
- $P \equiv P_1 \sqcup P_2$. We construct $\mathcal{G}_P$ from the games $\mathcal{G}_{P_1}$ and $\mathcal{G}_{P_2}$ for subprograms P_1 and P_2 as follows: we add a new angelic location l_{in}^P and a transition $l_{in}^P \xrightarrow{I} l_{in}^{P_i}$ $(i = 1, 2)$, and identify $l_{out}^P = l_{out}^{P_1} = l_{out}^{P_2}$.
- $P \equiv P_1 \sqcap P_2$. The same as the above item, but with l_{in}^P being demonic.

Remark 3.3. *Actually, $\mathcal{G}_P$ defined above is an operator-valued game rather than a general quantum game. However, for initialisation $P \equiv q := |0\rangle$, we can define $\mathcal{G}_P$ as a quantum game (but not an operator-valued game) in a way much simpler than above: $\mathcal{G}_P$ has only two locations l_{in}^P, l_{out}^P and a single transition $l_{in}^P \xrightarrow{\mathcal{E}_*} l_{out}^P$, where both l_{in}^P and l_{out}^P are standard locations, and super-operator $\mathcal{E}_*$ is given as*

$$\mathcal{E}_*(\rho) = \sum_n |0\rangle\langle n|\rho|n\rangle\langle 0|$$

for all density operators ρ. In this paper, we choose to use an operator-valued game $\mathcal{G}_P$ rather an a general quantum game in modelling quantum program P because it significantly simplify the synthesis problem of ranking super-martingales.

Example 3.1. *The game structures of example programs in Subsection 2.2 are shown in Fig. 1. For clarity, the initialisation of the programs are simply represented by transitions valued by super-operators $\mathcal{E}_0$, $\mathcal{E}_1$ and $\mathcal{E}_2$, respectively.*

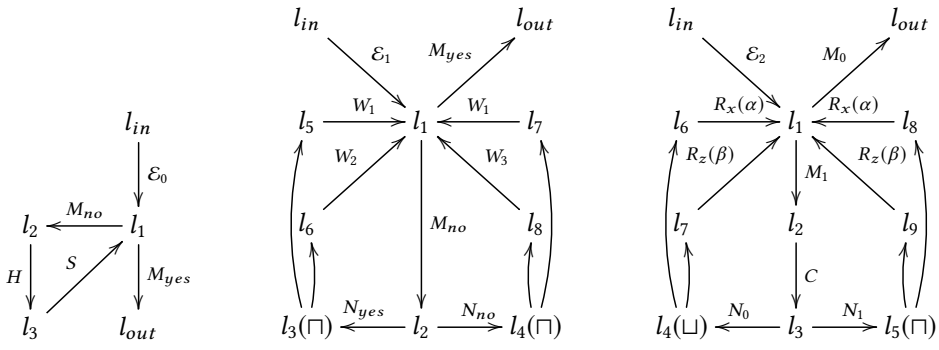
4 TERMINATION PROBLEMS

Now we can formally define the termination problems for a quantum program P based on its game representation $\mathcal{G}_P$. They are straightforward quantum generalisations of the termination problems for probabilistic programs considered in [Chatterjee et al. 2016].

Definition 4.1. (1) *A terminating run of game $\mathcal{G}_P$ is a finite run ending at location l_{out}^P; that is, a run of the form*

$$\theta = (l_{in}^P, \rho_0), (l_1, \rho_1), \ldots, (l_{k-1}, \rho_{k-1}), (l_{out}^P, \rho_k)$$

with $l_1, \ldots, l_{k-1} \neq l_{out}^P$. The running time of θ is $T(\theta) = k$, and the probability associated to θ is $\Pr(\theta) = tr(\rho_k)$.



(a) QW (b) DQW (c) ADR

Fig. 1. Game structures of example programs

(2) *For an infinite run* $\theta = (l_{in}^P, \rho_0)\ldots,(l_k,\rho_k),\ldots$ *of game* $\mathcal{G}_P$*, its running time is* $T(\theta)=\infty$*, and the probability associated to it is:*

$$\Pr(\theta) = \lim_{k\to\infty} tr(\rho_k).$$

Definition 4.2. *Given an angelic scheduler* σ*, a demonic scheduler* τ *in game* $\mathcal{G}_P$ *and a constant* $x \geq 0$*.*

(1) *The termination probability of program* P *according to* σ *and* τ*, and the probability that the termination time of* P *according to* σ *and* τ *exceeds* x *are:*

$$\Pr(P|\sigma,\tau) = \sum_{\theta} \Pr(\theta);$$

$$\Pr(P|\sigma,\tau; T > x) = \sum_{\theta \text{ s.t. } \mathrm{T}(\theta)>\mathrm{x}} \Pr(\theta),$$

respectively, where θ *ranges over all terminating runs of game* $\mathcal{G}_P$ *satisfying* σ *and* τ*.*

(2) *The expected running time of* P *according to* σ *and* τ *is:*

$$ET(P|\sigma,\tau) = \sum_{\theta} \Pr(\theta)\cdot T(\theta),$$

where θ *ranges over all runs of game* $\mathcal{G}_P$ *satisfying* σ *and* τ*.*

Definition 4.3 (Almost-sure termination, finite-termination). (1) *A program* P *is almost-surely terminating if there exists an angelic scheduler* σ *such that* $\Pr(P|\sigma,\pi)=1$ *for all demonic schedulers* τ*.*

(2) *A program* P *is finite-terminating if there exists an angelic scheduler* σ *such that* $ET(P|\sigma,\tau)<\infty$ *for all demonic schedulers* τ*.*

Definition 4.4 (Expected running time). *The expected running time of a program* P *is defined as:*

$$ET(P) = \inf_{\sigma}\sup_{\tau} ET(P|\sigma,\tau)$$

where σ,τ *ranges over all angelic and demonic schedulers of* P*, respectively.*

5 LINEAR RANKING SUPER-MARTINGALES

In this section, we first introduce the notion of linear ranking super-martingales for quantum program. Then we show that existence of linear ranking function of a quantum program with a supporting invariant guarantees its finite termination.

5.1 Definitions

First, let us recall the notion of invariant from [Ying et al. 2017]. Let P be a given quantum program and $\mathcal{G}_P$ a game representation of P. A set Π of paths in $\mathcal{G}_P$ is said to be prime if for each

$$\pi = l_1 \stackrel{\mathcal{E}_1}{\rightarrow} \ldots \stackrel{\mathcal{E}_{n-1}}{\rightarrow} l_n \in \Pi$$

its proper initial segments

$$l_1 \stackrel{\mathcal{E}_1}{\rightarrow} \ldots \stackrel{\mathcal{E}_{k-1}}{\rightarrow} l_k \notin \Pi$$

for all $k < n$. Moreover, a Hermitian operator H is called a quantum predicate if $0 \sqsubseteq H \sqsubseteq I$.

Definition 5.1 (Invariant). (1) *A multiplicative invariant of program P is a set $\mathcal{O} = \{O_l\}_{l \in L}$ of quantum predicates in $\mathcal{H}_P$ labelled by locations in $\mathcal{G}_P$ satisfying the condition: for every $l \in L$, for any input density operator ρ, and for every path π from l_0 to l,*

$$tr(O_{l_0}\rho) \leq \frac{tr\left(O_l \mathcal{E}_\pi(\rho)\right)}{tr\left(\mathcal{E}_\pi(\rho)\right)}. \tag{7}$$

(2) *A set $\mathcal{O} = \{O_l\}_{l \in L}$ of quantum predicates is termed an additive invariant of program P if for every $l \in L$, for any input density operator ρ, and for any prime set Π of paths from l_0 to l, we have:*

$$tr(O_{l_0}\rho) \leq 1 - tr\left(\mathcal{E}_\Pi(\rho)\right) + tr\left(O_l \mathcal{E}_\Pi(\rho)\right) \tag{8}$$

where $\mathcal{E}_\Pi = \sum \{|\mathcal{E}_\pi : \pi \in \Pi|\}$.

It is easy to see that every multiplicative invariant is an additive invariant. However, whenever it exists, a multiplicative invariant is easy to use because only a single path π appears in its definition equation (7) and in contrast a set Π of paths is involved in the definition equation (8) of a additive invariant. It is worth noting that there are two trivial multiplicative (and thus also additive) invariants: the *identity invariant* $O_l = I$ for all $l \in L$ and the *zero invariant* $O_l = 0$ for all $l \in L$.

Definition 5.2 (Pre-expectation). *Let $\eta : L \times \mathcal{D}(\mathcal{H}_P) \rightarrow \mathbb{R}$ be a function. Then the pre-expectation induced by η in game $\mathcal{G}_P$ is the function $pre_\eta : L \times \mathcal{D}(\mathcal{H}_P) \rightarrow \mathbb{R}$ defined as follows:*

$$pre_\eta(l,\rho) = \begin{cases} \min\{\eta(l',\rho) | l \stackrel{I}{\rightarrow} l'\} & \text{if } l \in L_A; \\ \max\{\eta(l',\rho) | l \stackrel{I}{\rightarrow} l'\} & \text{if } l \in L_D; \\ \sum\{\eta(l',\mathcal{E}(\rho)) | l \stackrel{\mathcal{E}}{\rightarrow} l'\} & \text{if } l \in L_S. \end{cases}$$

Definition 5.3 (Linear ranking super-martingale). *Let $\epsilon, K \in \mathbb{R}$ be two constants with $\epsilon > 0$. A function $\eta : L \times \mathcal{D}(\mathcal{H}_P) \rightarrow \mathbb{R}$ is called a (K,ϵ)-linear ranking super-martingale (LRSM for short) for program P with respect to (a multiplicative or an additive) invariant $\mathcal{O} = \{O_l\}_{l \in L}$ if:*

(1) *for each $l \in L$, function $\eta(l,\cdot) : \mathcal{D}(\mathcal{H}) \rightarrow \mathbb{R}$ is linear over partial density operators in $\mathcal{H}$, i.e., $\eta(l, p\rho_1 + q\rho_2) = p\eta(l,\rho_1) + q\eta(l,\rho_2)$ for any $p, q \geq 0$ and $\rho_1, \rho_2 \in \mathcal{D}(\mathcal{H})$;*
(2) *for all $l \in L$ and density operators ρ, $tr(O_l\rho) + K - 1 \leq \eta(l,\rho)$;*
(3) *for all $l \neq l^P_{out}$ and density operators ρ, $tr(O_l\rho) + pre_\eta(l,\rho) \leq \eta(l,\rho) + 1 - \epsilon$.*

The above definitions are the quantum generalisations of Definitions 6, 7 and 8 in [Chatterjee et al. 2016]. Definition 5.3 also generalises Definition 9.1 in [Ying 2011] where a ranking function was defined to be always nonnegative integer-valued.

Remark 5.1. *For the identity invariant (i.e. $O_l = I$ for all $l \in L$), conditions (2) and (3) in the above become a regular form: $K \leq \eta(l,\rho)$ and $pre_\eta(l,\rho) \leq \eta(l,\rho) - \epsilon$. In this case, we only need to consider $(0,1)$-LRSM, since η is a (K,ϵ)-LRSM if and only if $\eta_{K,\epsilon}$ is a $(0,1)$-LRSM, where*

$$\eta_{K,\epsilon}(l,\rho) := \frac{\eta(l,\rho) - K tr(\rho)}{\epsilon}.$$

But for non-trivial invariants, we have to consider the $(0,\epsilon)$-LRSM for generality, due to the restriction $O_l \sqsubseteq I$ for quantum predicates O_l.

5.2 Termination Theorems

Now we are ready to establish two termination theorems for quantum programs. The main mathematical tool needed in proving them is the following well-known theorem in probability theory, which were used for the same purpose in the studies of probabilistic programming.

Theorem 5.1 (Foster's Theorem [Bournez and Garnier 2005; Foster 1953]). *Given a Markov chain over a finite or countably infinite space S with matrix $(P_{st})_{s,t\in S}$ of transition probabilities. Then the Markov chain almost surely reaches a subset $X \subseteq S$ with finite expected time if and only if there exist constants $\epsilon > 0$ and $K \in \mathbb{R}$ and a function $V : S \to \mathbb{R}$ satisfying the following conditions:*

(1) *for all $s \in S$, $V(s) \geq K$;*
(2) *for all $s \notin X$, the mean drift*

$$\Delta V(s) = \sum_{t\in S} P_{st} \cdot V(t) - V(s) \leq -\epsilon.$$

Moreover, for a given initial state $s_0 \in S$, we have:

$$\text{the expected running time } \leq \frac{V(s_0) - K}{\epsilon}.$$

Based on the concept of LRSM introduced in Definition 5.3, we have a quantum generalisation of the Foster's theorem.

Theorem 5.2. *[Quantum Generalisation of Foster's Theorem] A quantum program without angelic or demonic choices is finite-terminating for every initial configuration if and only if it has a (K,ϵ)-LRSM η with respect to some additive invariant, and some constants $\epsilon > 0$ and $K \in \mathbb{R}$.*

Proof. (Sketch) We omit the proof of the "if" part, since a more general form of it will be proved as the termination theorem in the remainder of this section.

To prove the "only if" part, we assume that the program is finite-terminating. Then we can use the assumed finite expected running time $ET(\cdot)$ to define an LRSM by:

$$\eta(l,\rho) := tr(\rho) \cdot ET\left(P\left(l, \frac{\rho}{tr\rho}\right)\right)$$

where $P(l,\rho)$ is the quantum program obtained from P by replacing the initial configuration (l^P_{in}, ρ_0) by (l,ρ). According to the finite-termination of the program η is well defined. Then following immediately from the definition of expected running time, it holds that

$$\eta(l,\rho) \geq 0 = \eta(l_{out},\rho)$$

for all location l and all partial density operators ρ. The fact that $pre_\eta(l,\rho) \leq \eta(l,\rho) - 1$ for all density operators ρ is easily derived from the definition of expected running time and equation (6).

Furthermore, the absence of angelic or demonic choices in P implies the linearity of $\eta(l,\cdot)$. Thus, η is actually a $(0,1)$-LRSM with respect to the identity invariant. □

Remark 5.2. *Quantum Markov chains defined studied in [Ying et al. 2013a,b] can be regarded as a special type of quantum games without angelic or demonic choices. For a quantum Markov chain in a Hilbert space $\mathcal{H}$ with the transition super-operator $\mathcal{E}$, the initial state ρ, and a measurement $\{M_0, M_1\}$ for termination checking, we can construct a corresponding quantum game structure $\mathcal{G}$ as follows:*

- *it has only two locations, the initial location l_0 and the final location l_1;*
- *there are two transitions from l_0:*

$$l_0 \overset{\mathcal{E}(M_1\cdot M_1^\dagger)}{\longrightarrow} l_0, \qquad l_0 \overset{M_0\cdot M_0^\dagger}{\longrightarrow} l_1$$

  (Note that $\mathcal{E}(M_1\cdot M_1^\dagger)+M_0\cdot M_0^\dagger \approx \mathcal{I}$. So, the normalisation condition (6) is satisfied);
- *the execution starts from location l_0 in state ρ, and terminates when reaches location l_1. Thus, Theorem 5.2 can be directly applied to quantum Markov chains.*

Due to the angelic and demonic choices in general case, the expected running time $ET(P(l,\cdot))$ of a quantum program may not be linear for partial density operators and thus could not be an LRSM. On the other hand, whenever an LRSM exists, termination of the program is guarantted. We first prove this fact for the case of LRSMs with additive invariants.

Theorem 5.3 (Termination Theorem with Additive Invariants). *Suppose that quantum program P has a (K,ϵ)-LRSM η with respect to an additive invariant $\mathcal{O}=\{O_l\}_{l\in L}$. Then P is finite-terminating with any initial state ρ_0 satisfying $tr(\rho_0 O_{l_0})=1$, and the expected running time:*

$$ET(P)\le \frac{\eta(l_{in}^P,\rho_0)-K}{\epsilon}.$$

Proof. Given a (K,ϵ)-LRSM η for P with respect to multiplicative invariant $\mathcal{O}=\{O_l\}_{l\in L}$, we define an angelic scheduler σ as follows: for each finite run θ ending at configuration (l,ρ) with $l\in L_A$, $\sigma(\theta)$ is the transition $l\overset{\mathcal{I}}{\longrightarrow} l'$ that minimises $\eta(l',\rho)$:

$$l'=\arg\min_{l'}\{\eta(l',\rho): l\overset{\mathcal{I}}{\longrightarrow} l'\}.$$

Note that the transition relation $\longrightarrow$ is finitely branching at angelic (and demonic) locations because the syntax of programs only allows finite angelic (and demonic) choice. Thus, $\sigma(\theta)$ is well-defined.

Now by definition, it suffices to show that for any demonic scheduler τ, we have:

$$ET(P|\sigma,\tau)\le \frac{\eta(l_{in}^P,\rho_0)-K}{\epsilon}.$$

We note that under the schedulers σ and τ, a transition $(l,\rho)\overset{\mathcal{I}}{\longrightarrow}(l',\rho)$ from a angelic or demonic location $l\in L_A\cup L_C$ always satisfies $\eta(l',\rho)\le pre_\eta(l,\rho)$. Now, consider the paths $\pi=l_1\overset{\mathcal{E}_1}{\longrightarrow}\ldots\overset{\mathcal{E}_{n-1}}{\longrightarrow} l_n$ under σ and τ. We write $|\pi|$ for their length n and $L(\pi)$ for their last location l_n. Let

$$\Pi_{ter}=\left\{l_1\overset{\mathcal{E}_1}{\longrightarrow}\ldots\overset{\mathcal{E}_{n-1}}{\longrightarrow} l_n \mid l_1=l_{in}^P, l_n=out^P\right\}$$

$$\Pi_{nter}=\left\{l_1\overset{\mathcal{E}_1}{\longrightarrow}\ldots\overset{\mathcal{E}_{n-1}}{\longrightarrow} l_n \mid l_1=l_{in}^P, l_n\neq l_{out}^P\right\}$$

be the set of terminating and non-terminating paths from the initial location, respectively. Define $\Pi_{ter,n} = \{\pi \in \Pi_{ter} \mid |\pi| \leq n\}$, $\Pi_{nter,n} = \{\pi \in \Pi_{nter} \mid |\pi| = n\}$, and $\Pi_n = \Pi_{ter,n} \cup \Pi_{nter,n}$. It is easy to verify that

$$tr(\mathcal{E}_{\Pi_n}(\rho)) = tr(\mathcal{E}_{\Pi_{ter,n}}(\rho)) + tr(\mathcal{E}_{\Pi_{nter,n}}(\rho)) = 1,$$

for any density operator ρ and any n. Then,

$$\begin{aligned}
&\sum_{\pi\in\Pi_n} \eta(L(\pi),\mathcal{E}_\pi(\rho_0)) - \sum_{\pi\in\Pi_{n+1}} \eta(L(\pi),\mathcal{E}_\pi(\rho_0)) \\
&\quad= \sum_{\pi\in\Pi_{nter,n}} \left[\eta(L(\pi),\mathcal{E}_\pi(\rho_0)) - \sum\{\eta(l',(\mathcal{E}\circ\mathcal{E}_\pi)(\rho)) \mid L(\pi) \xrightarrow{\mathcal{E}} l'\}\right] \\
&\quad\geq \sum_{\pi\in\Pi_{nter,n}} \left[\eta(L(\pi),\mathcal{E}_\pi(\rho_0)) - pre_\eta(L(\pi),\mathcal{E}_\pi(\rho_0))\right] \\
&\quad\geq \sum_{\pi\in\Pi_{nter,n}} \left(\epsilon tr(\mathcal{E}_\pi(\rho_0)) + tr(O_{L(\pi)}\mathcal{E}_\pi(\rho_0)) - tr(\mathcal{E}_\pi(\rho_0))\right) \\
&\quad= \epsilon tr(\mathcal{E}_{\Pi_{nter,n}}(\rho_0)) - tr(\mathcal{E}_{\Pi_{nter,n}}(\rho_0)) + \sum_{\pi\in\Pi_{nter,n}} tr(O_{L(\pi)}\mathcal{E}_\pi(\rho_0)) \\
&\quad= \epsilon tr(\mathcal{E}_{\Pi_{nter,n}}(\rho_0)).
\end{aligned}$$

Here the last equation is due to

$$\sum_{\pi\in\Pi_{nter,n}} tr(O_{L(\pi)}\mathcal{E}_\pi(\rho_0)) = tr(\mathcal{E}_{\Pi_{nter,n}}(\rho_0))$$

which follows from the condition $tr(\rho_0 O_{l_0}) = 1$ and equation (8). In fact, divide the set $\Pi_{nter,n}$ according to the last location of the paths, namely $\Pi_{nter,n} = \cup_{l\in L}\Pi_l$ where $\Pi_l = \{\pi \in \Pi_{nter,n} \mid L(\pi) = l\}$. It suffices to prove that $tr(O_l\mathcal{E}_{\Pi_l}(\rho_0)) = tr(\mathcal{E}_{\Pi_l}(\rho_0))$ for all $l \in L$ due to the linearity of tr. $tr(O_l\mathcal{E}_{\Pi_l}(\rho_0)) \leq tr(\mathcal{E}_{\Pi_l}(\rho_0))$ is obvious since $O_l \sqsubseteq I$. Note that all the Π_l are prime sets as $\Pi_{nter,n}$ is prime, then from equation (8),

$$tr(O_l\mathcal{E}_{\Pi_l}(\rho_0)) \geq tr(O_{l_0}\rho_0) - 1 + tr(\mathcal{E}_{\Pi_l}(\rho_0)) = tr(\mathcal{E}_{\Pi_l}(\rho_0)).$$

For convenience, we write a partial density operator $\rho_n := \mathcal{E}_{\Pi_{ter,n}}(\rho)$, then $tr(\mathcal{E}_{\Pi_{nter,n}}(\rho_0)) = 1 - tr\rho_n$. We have:

$$\begin{aligned}
\epsilon\sum_{k=1}^{n}(1 - tr\rho_k) &= \sum_{k=1}^{n} \epsilon tr(\mathcal{E}_{\Pi_{nter,k}}(\rho_0)) \\
&\leq \sum_{k=1}^{n}\left(\sum_{\pi\in\Pi_k} \eta(L(\pi),\mathcal{E}_\pi(\rho_0)) - \sum_{\pi\in\Pi_{k+1}} \eta(L(\pi),\mathcal{E}_\pi(\rho_0))\right) \\
&= \sum_{\pi\in\Pi_1} \eta(L(\pi),\mathcal{E}_\pi(\rho_0)) - \sum_{\pi\in\Pi_n} \eta(L(\pi),\mathcal{E}_\pi(\rho_0)) \\
&= \eta(l_0,\rho_0) - \sum_{\pi\in\Pi_n} \eta(L(\pi),\mathcal{E}_\pi(\rho_0)),
\end{aligned}$$

combining it with the fact that

$$\begin{aligned}
\sum_{\pi\in\Pi_n} \eta(L(\pi),\mathcal{E}_\pi(\rho_0)) &\geq \sum_{\pi\in\Pi_n} \left(Ktr(\mathcal{E}_\pi(\rho_0)) + tr(O_{L(\pi)}\mathcal{E}_\pi(\rho_0)) - tr(\mathcal{E}_\pi(\rho_0))\right) \\
&= \sum_{\pi\in\Pi_n} Ktr(\mathcal{E}_\pi(\rho_0)) = Ktr\mathcal{E}_{\Pi_n}(\rho_0) = K,
\end{aligned}$$

we obtain:

$$\epsilon \sum_{k=1}^{n-1}(1 - tr\rho_k) \leq \eta(l_0, \rho_0) - K. \tag{9}$$

Note that $\sum_{k=1}^{n-1}(1 - tr\rho_k)$ has an upper bound independent of n. Thus, $\lim_{n\to\infty} tr\rho_n = 1$, which means that P is almost terminating. It further implies that $ET(P) = \sum_{n=1}^{\infty}(1 - tr\rho_n)$ converges. Therefore, P is finite-terminating, and from inequality (8), we have:

$$ET(P) \leq \frac{\eta(l_{in}^P, \rho_0) - K}{\epsilon}.$$

□

Remark 5.3. *For the identity invariant, the condition $tr(O_{l_0}\rho_0) = 1$ is satisfied for all initial state ρ_0. So, the existence of LRSMs in this case actually implies termination independent of initial states. Of course, many programs only terminate for some initial states (and does not terminate for other initial states). So, one has to find LRSMs for non-trivial invariants rather than the identity one.*

Now we establish a termination theorem based on LRSMs with multiplicative invariants.

Theorem 5.4 (Termination Theorem with Multiplicative Invariants). *Suppose that quantum program P has a (K, ϵ)-LRSM η with respect to a multiplicative invariant $\boldsymbol{O} = \{O_l\}_{l\in L}$. If $\epsilon + tr\left(O_{l_0}\rho_0\right) > 1$, then P is finite-terminating and*

$$ET(P) \leq \frac{\eta(l_{in}^P, \rho_0) - \left[K - 1 + tr\left(O_{l_0}\rho_0\right)\right]}{\epsilon - 1 + tr\left(O_{l_0}\rho_0\right)}.$$

Proof. We first define an angelic scheduler σ as the same as in the proof of Theorem 5.3. Then, it suffices to show that for any demonic scheduler τ, we have:

$$ET(P|\sigma, \tau) \leq \frac{\eta(l_{in}^P, \rho_0) - \left[K - 1 + tr\left(O_{l_0}\rho_0\right)\right]}{\epsilon - 1 + tr\left(O_{l_0}\rho_0\right)}.$$

The single-path properties of multiplicative invariants enable us to prove the result in a more explicit approach, namely, by an application of the (classical) Foster's Theorem. We first construct a (classical) Markov chain from the quantum game $\mathcal{G}_P$. A configuration (l, ρ) is said to be normalised if ρ is a density operator, i.e. $tr(\rho) = 1$. A normalised configuration is reachable in $\mathcal{G}_P$ if there exists a finite run $(l_0, \rho_0), (l_1, \rho_1), ..., (l_k, \rho_k)$ of satisfying σ and τ such that $l_k = l$ and $\rho = \frac{\rho_k}{tr(\rho_k)}$. We write Ω for the set of all reachable configurations. Obviously, $(l_0, \rho_0) \in \Omega$. It is easy to see that Ω is a finite or countably infinite set because the transition relation is countably branching. For any configurations $c = (l, \rho), c' = (l', \rho') \in \Omega$, we define the transition probability from c to c' as follows:

$$P_{cc'} = \begin{cases} tr(\mathcal{E}(\rho)) & \text{if } l \xrightarrow{\mathcal{E}} l' \text{ and } \rho' = \frac{\mathcal{E}(\rho)}{tr(\mathcal{E}(\rho))}, \\ 0 & \text{otherwise.} \end{cases}$$

Claim 1: For each $c \in \Omega$, it holds that $\sum_{c'\in\Omega} P_{cc'} = 1$.

In fact, if $c = (l, \rho)$ and $l \in L_A \cup L_D$, it is obvious because σ, τ are fixed, and thus there exists only one transition $l \xrightarrow{\mathcal{I}} l'$ leading to $c' = (l', \rho)$. If $l \in L_S$, then by definition we have:

$$\begin{aligned}\sum_{c' \in \Omega} P_{cc'} &= \sum \left\{ |tr(\mathcal{E}(\rho)) : l \xrightarrow{\mathcal{E}} l' \text{ for some } l'| \right\} \\ &= tr\left[\left(\sum \{ |\mathcal{E} : l \xrightarrow{\mathcal{E}} l' \text{ for some } l'| \} \right)(\rho) \right] \\ &= tr(\mathcal{I}(\rho)) = 1.\end{aligned}$$

Here, the second equality follows from linearity of $tr(\cdot)$ and $\mathcal{E}$, and the third equality comes from equation (6).

The above claim shows that a Markov chain is defined over the space Ω with the matrix $(P_{cc'})_{c,c' \in \Omega}$ of transition probabilities.

Claim 2: $\eta(c) \geq K - 1 + tr\left(O_{l_0}\rho_0\right)$ for all $c \in \Omega$.

Indeed, if $c = (l, \rho)$, then there exists a path π from l_0 to l such that $\rho = \frac{\mathcal{E}_\pi(\rho_0)}{tr(\mathcal{E}_\pi(\rho_0))}$. Then it follows from equation (7) that

$$tr\left(O_l\rho\right) = tr\left[O_l \frac{\mathcal{E}_\pi(\rho_0)}{tr\left(\mathcal{E}_\pi(\rho_0)\right)}\right] = \frac{tr\left(O_l\mathcal{E}_\pi(\rho_0)\right)}{tr\left(\mathcal{E}_\pi(\rho_0)\right)} \geq tr\left(O_{l_0}\rho_0\right). \tag{10}$$

Consequently, using condition 2 in Definition 5.3 we have:

$$\eta(c) = \eta(l, \rho) \geq K - 1 + tr(O_l\rho) \geq K - 1 + tr\left(O_{l_0}\rho_0\right).$$

We now write X for the set of configurations (l, ρ) in Ω with $l = l^P_{out}$.

Claim 3: $\Delta\eta(c) - \eta(c) \leq -\left[\epsilon - 1 + tr\left(O_{l_0}\rho_0\right)\right]$ for any $c \notin X$.

Indeed, if $c = (l, \rho)$ and $l \neq l^P_{out}$, then we have:

$$pre_\eta(l, \rho) - \eta(l, \rho) \leq -\left[\epsilon - 1 + tr\left(O_{l_0}\rho_0\right)\right]$$

from equation (10) and condition 3 in Definition 5.3. So, it suffices to show that $\sum_{c'} P_{cc'} \cdot \eta(c') \leq pre_\eta(l, \rho)$.

- If $l \in L_A$, let l' be the location chosen by the angelic scheduler σ at (l, ρ), then

$$\begin{aligned}\sum_{c'} P_{cc'} \cdot \eta(c') &= tr(\rho) \cdot \eta(l', \rho) = \eta(l', \rho) \\ &= \min\{\eta(l'', \rho) : l \xrightarrow{\mathcal{I}} l''\} = pre_\eta(l, \rho).\end{aligned}$$

- If $l \in L_D$, let l' be the location chosen by the demonic scheduler τ at (l, ρ), then

$$\begin{aligned}\sum_{c'} P_{cc'} \cdot \eta(c') &= tr(\rho) \cdot \eta(l', \rho) = \eta(l', \rho) \\ &\leq \max\{\eta(l'', \rho) : l \xrightarrow{\mathcal{I}} l''\} = pre_\eta(l, \rho).\end{aligned}$$

- If $l \in L_S$, then we have:

$$\begin{aligned}\sum_{c'} P_{cc'} \cdot \eta(c') &= \sum \{ |tr(\mathcal{E}(\rho)) \cdot \eta\left(l', \frac{\mathcal{E}(\rho)}{tr(\mathcal{E}(\rho))}\right) : l \xrightarrow{\mathcal{E}} l'| \} \\ &= \sum \{ |\eta(l', \mathcal{E}(\rho)) : l \xrightarrow{\mathcal{E}} l' \text{ for some } l'| \} = pre_\eta(l, \rho)\end{aligned}$$

  because $\eta(l', \cdot)$ is linear.

Combining the above arguments, we complete the proof by Foster's Theorem. □

Remark 5.4. *It is worthy to carefully compare the above theorem with Theorem 5.3. As pointed out before, multiplicative invariants are a special type of additive invariants. So, Theorem 5.3 applies to LRMSs with multiplicative invariants. On the other hand, Theorem 5.4 only consider LRMSs with multiplicative invariants but it allows a larger class of initial states ρ_0, noting that*

$$(\epsilon > 0 \wedge tr\left(O_{l_0}\rho_0\right) = 1) \Rightarrow \epsilon + tr\left(O_{l_0}\rho_0\right) > 1,$$

but the inverse is not true.

6 REALISABILITY AND SYNTHESIS OF LINEAR RANKING SUPER-MARTINGALES

Theorem 5.4 shows that existence of LRSMs for a quantum program with respect to an invariant guarantees finite-termination of the program. The problem of invariant generation for quantum programs was discussed in [Ying et al. 2017]. In this section, we further consider the problem of realizability and synthesis of LRMSs with respect to a given invariant $\mathcal{O}$, precisely stated as follows:

Problem 6.1 (Realizability and Synthesis). *Given a quantum program P and an invariant $\mathcal{O} = \{O_l\}_{l\in L}$ for P, does there exist an LRSM for P with respect to $\mathcal{O}$? If so, how to construct it?*

In particular, we extend the constrain-based approach to synthesis of linear ranking functions for classical programs [Colón et al. 2001, 2003; Podelski and Rybalchenko 2004] and for probabilistic programs [Chakarov and Sankaranarayanan 2013; Chatterjee et al. 2016] in order to solve Problem 6.1 for quantum programs.

6.1 Templates of LRSMs and Gleason's Theorem

The first step of the constraint-based approach is to choose an appropriate template of LRSMs. For both classical and probabilistic programs, the template of linear ranking functions can be straightforward taken as an affine expression over program variables. However, a reasonable template of linear ranking functions for quantum programs is determined by a fundamental theorem in quantum mechanics. Let $\mathcal{H}$ be a Hilbert space. We write $\mathcal{S}(\mathcal{H})$ for the set of all closed subspaces of $\mathcal{H}$. Recall from [Dvurečenskij 1993] that a state in $\mathcal{S}(\mathcal{H})$ is a mapping $m : \mathcal{S}(\mathcal{H}) \to [0, 1]$ such that $m(\mathcal{H}) = 1$ and

$$m\left(\bigvee_{i=0}^{\infty} X_i\right) = \sum_{i=0}^{\infty} m\left(X_i\right)$$

for any family $\{X_i\}_{i=0}^{\infty}$ of mutually orthogonal subspaces of $\mathcal{H}$, where $\bigvee_{i=0}^{\infty} X_i$ stands for the smallest closed subspace of $\mathcal{H}$ that contains all X_i $(i \geq 0)$. The following fundamental theorem gives an elegant characterisation of states in $\mathcal{S}(\mathcal{H})$.

Theorem 6.1 (Gleason's Theorem [Dvurečenskij 1993; Gleason 1957]). *If $\mathcal{H}$ is separable and $\dim \mathcal{H} > 2$, then for each state m in $\mathcal{S}(\mathcal{H})$, there exists a unique positive Hermitian matrix R with $tr(R) = 1$ such that*

$$m(X) = tr(RP_X)$$

for all $X \in \mathcal{S}(\mathcal{H})$, where P_X is the project onto X.

The conclusion of the above theorem is not true when $\dim \mathcal{H} = 2$ (see [Dvurečenskij 1993], page 130 for a counter-example).

As we will see in the proofs of Theorems 6.2 and 6.3, Gleason's theorem plays an essential role in determining the form of templates of LRSMs for quantum programs. The basic idea is as follows: Gleason's theorem, together with the definition conditions of an LRSM, implies that each LRSM η can be written in a trace form

$$\eta(l, \rho) = tr(R_l\rho)$$

where for any l, R_l is a Hermitian trace class operator. For a finite dimensional Hilbert space $\mathcal{H}$, the operator (matrix) R_l can be derived directly from the theorem: define a mapping $m : \mathcal{S}(\mathcal{H}) \to \mathbb{R}$ by

$$m(X) = \dim X \cdot \eta\left(l, \frac{P_X}{\dim X}\right) - tr(O_l P_X) + (1 - K) \cdot \dim X,$$

then if $m(\mathcal{H}) = 0$, let $R_l = O_l + (K-1) \cdot I$; otherwise, $\frac{m}{m(\mathcal{H})}$ is a state in $\mathcal{S}(\mathcal{H})$ due to the linearity of $\eta(l, \cdot)$, and thus let $m(X) = tr(RP_X)$ and $R_l = R + O_l + (K-1) \cdot I$. When the state Hilbert space $\mathcal{H}$ is infinite dimensional, the projection of operator R_l onto any finite dimensional subspaces can be defined as above. In particular, the proof can be achieved by considering the projection onto one dimensional subspaces, just like the proof of Gleason's theorem [Gleason 1957]. This argument shows that it is sufficient to use the trace form $tr(R_l \rho)$ as a template of the LRSM $\eta(l, \rho)$.

6.2 Reduction to SDP (Semi-Definite Programming) Problem

Of course, for a quantum program P with a 2-dimensional state Hilbert space, Problem 6.1 can be easily solved. In this subsection, using Gleason's Theorem, we are able to reduce Problem 6.1 to a constraint satisfaction problem in the case where the dimension of the state Hilbert space of program P: $\dim \mathcal{H}_P > 2$. Before doing it, let us first introduce a notation. For any super-operator $\mathcal{E}$, we write $\mathcal{E}^*$ for its dual, i.e. if $\mathcal{E}$ has the Kraus operator-sum representation $\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger$, then

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

for every operator A.

Let us first consider quantum programs that contains no angelic choices but may have demonic choices. In this case, the following theorem shows that Problem 6.1 can be reduced to an SDP (Semi-Definite Programming) problem.

Theorem 6.2. *Let P be a quantum program without angelic choice. Given an invariant $\mathcal{O} = \{O_l\}_{l \in L}$ for P. If $n := \dim \mathcal{H}_P > 2$, then the realizability and synthesis problem with respect to $\mathcal{O}$ is equivalent to the following constraint satisfaction problem:*

- *arbitrarily fix the value of K, then find a real number $\epsilon > 0$ (or $> 1 - tr\left(O_{l_0}\rho_0\right)$ for multiplicative invariants) and complex Hermitian matrices R_l ($l \in L$) satisfying the constraint:*

$$\left(\bigwedge_{l \in L} \gamma_l\right) \wedge \left(\bigwedge_{l \in L \setminus \{l^P_{out}\}} \delta_l\right)$$

where:

(1) *for each $l \in L$,*

$$\gamma_l := 0 \sqsubseteq R_l - O_l - (K-1) \cdot I;$$

(2) *for each $l \in L_D$,*

$$\delta_l := \bigwedge_{l \xrightarrow{I} l'} \left(0 \sqsubseteq R_l - R_{l'} - O_l + (1-\epsilon) \cdot I\right);$$

(3) *for each $l \in L_S \setminus \{l^P_{out}\}$,*

$$\delta_l := 0 \sqsubseteq R_l - \sum_{l \xrightarrow{\mathcal{E}} l'} \mathcal{E}^*\left(R_{l'}\right) - O_l + (1-\epsilon) \cdot I.$$

Proof. We reduce the realizability and synthesis problem to the constrain satisfaction problem in two steps.

Step 1: With Definition 5.3, we see that Problem 6.1 can be equivalently stated as the following constraint satisfaction problem: for some $K > -\infty$ (which is usually chosen as $K = 0$), find a real number $\epsilon > 0$ (or $> 1 - tr\left(O_{l_0}\rho_0\right)$ for multiplicative invariants) and linear functions $\eta(l,\cdot) : \mathcal{D}(\mathcal{H}_P) \to \mathbb{R}$ $(l \in L)$ satisfying the constraint:

$$\left(\bigwedge_{l\in L} \mu_l\right) \wedge \left(\bigwedge_{l\in L\setminus\{l^P_{out}\}} \nu_l\right)$$

where

$$\mu_l := (\forall \rho \in \mathcal{D}(\mathcal{H}_P) \text{ with } tr(\rho) = 1)\left[tr(O_l\rho) + K - 1 \le \eta(l,\rho)\right],$$
$$\nu_l := (\forall \rho \in \mathcal{D}(\mathcal{H}_P) \text{ with } tr(\rho) = 1)\xi(l,\rho),$$

and

$$\xi(l,\rho) := tr(O_l\rho) + pre_\eta(l,\rho) \le \eta(l,\rho) + 1 - \epsilon.$$

Furthermore, with Definition 5.2 we have:

- if $l \in L_D$, then

$$\xi(l,\rho) \Leftrightarrow \bigwedge \left\{ tr(O_l\rho) + \eta(l',\rho) \le \eta(l,\rho) + 1 - \epsilon : l \xrightarrow{\mathcal{I}} l' \right\}.$$

- if $l \in L_S \setminus \{l^P_{out}\}$, then

$$\xi(l,\rho) \Leftrightarrow tr(O_l\rho) + \sum \left\{ |\eta(l', \mathcal{E}(\rho)) : l \xrightarrow{\mathcal{E}} l'| \right\} \le \eta(l,\rho) + 1 - \epsilon.$$

Step 2: For each $l \in L$, since $\dim \mathcal{H}_P > 2$, by Gleason's Theorem it is easy to see that there exists a Hermitian matrix R_l such that $\eta(l,\rho) = tr(R_l\rho)$ for all $\rho \in \mathcal{D}(\mathcal{H})$. Then we assert:

(1) for each $l \in L$,

$$\begin{aligned}\mu_l &\Leftrightarrow (\forall \rho \in \mathcal{D}(\mathcal{H}_P) \text{ with } tr(\rho) = 1)\{tr\left[(R_l - O_l - (K-1)\cdot I)\,\rho\right] \ge 0\}\\ &\Leftrightarrow 0 \sqsubseteq R_l - O_l - (K-1)\cdot I.\end{aligned}$$

(2) for each $l \in L_D$,

$$\begin{aligned}\nu_l &\Leftrightarrow (\forall \rho \in \mathcal{D}(\mathcal{H}_P) \text{ with } tr(\rho) = 1) \bigwedge_{l \xrightarrow{I} l'} \{tr\left[(R_l - R_{l'} - O_l + (1-\epsilon)\cdot I)\,\rho\right] \ge 0\}\\ &\Leftrightarrow \bigwedge_{l \xrightarrow{I} l'} (0 \sqsubseteq R_l - R_{l'} - O_l + (1-\epsilon)\cdot I).\end{aligned}$$

(3) for each $l \in L_S \setminus \{l^P_{out}\}$,

$$\begin{aligned}\nu_l &\Leftrightarrow (\forall \rho \in \mathcal{D}(\mathcal{H}_P) \text{ with } tr(\rho) = 1)\left\{ tr\left[\left(R_l - \sum_{l \xrightarrow{\mathcal{E}} l'} \mathcal{E}^*(R_{l'}) - O_l + (1-\epsilon)\cdot I\right)\rho\right] \ge 0\right\}\\ &\Leftrightarrow 0 \sqsubseteq R_l - \sum_{l \xrightarrow{\mathcal{E}} l'} \mathcal{E}^*(R_{l'}) - O_l + (1-\epsilon)\cdot I\end{aligned}$$

□

For quantum programs with angelic choices, a constraint satisfaction problems can be derived in the same way as in the proof of the above theorem. However, such a constraint satisfaction problem cannot directly be solved by SDP solvers because the constraint at each location $l \in L_A$ is a disjunction of matrix positivity restrictions. In fact, a similar situation happens in termination analysis of classical programs with linear ranking functions, where the celebrated Farkas's lemma has been employed to transform it into a standard form of Linear Programming. We are going to solve our problem for quantum programs with the same strategy. But first of all, we have to establish a generalisation of Farkas's lemma in terms of observables (Hermitian operators) in quantum physics.

Lemma 6.1 (Generalized Farkas' Lemma for SDP). *Let $H_1, \cdots, H_n$ be a finite number of Hermitian operators in a finite-dimensional Hilbert space $\mathcal{H}$. Then the following two statements are equivalent:*

(1) *For any $\rho \in \mathcal{D}(\mathcal{H})$,*
$$\bigvee_k (tr(\rho H_k) > 0);$$
(2) *There exist non-negative numbers $p_1 \geq 0, \cdots, p_n \geq 0$, such that $p_1 + \cdots + p_n > 0$ and*
$$p_1 H_1 + p_2 H_2 + \cdots + p_n H_n \sqsupset 0.$$

Proof. It is obvious that statement 2 implies statement 1. We prove the converse in a similar way to the original Farkas's lemma, namely by invoking the Hyperplane Separation Theorem. Specifically, let d be the dimension of $\mathcal{H}$. We note that the set of all Hermitian operators of $\mathcal{H}$, denoted by $\mathcal{L}$, is in fact a $\frac{d(d+1)}{2}$-dimensional Hilbert space over reals, i.e.
$$\mathcal{L} \simeq \mathbb{R}^{\frac{d(d+1)}{2}}$$
where the inner product of two Hermitian operators $A, B \in \mathcal{L}$ is naturally defined as $tr(AB)$, and an orthonormal basis of $\mathcal{L}$ is
$$\{|k\rangle\langle k| \mid 0 \leq k \leq d-1\} \cup \left\{ \frac{|i\rangle\langle j| + |j\rangle\langle i|}{2} \mid 0 \leq i < j \leq d-1 \right\}.$$
Now denote by $\mathcal{P}$ the set of all positive definite operators in $\mathcal{H}$, and define
$$\mathcal{A} := \{p_1 H_1 + \cdots + p_n H_n \mid p_1 \geq 0, \cdots, p_n \geq 0, p_1 + \cdots + p_n > 0\}.$$
We assume by contradiction that $\mathcal{A} \cap \mathcal{P} = \emptyset$. Noting that both $\mathcal{P}$ and $\mathcal{A}$ are nonempty convex sets in $\mathcal{L}$. Then following the Hyperplane Separation Theorem, there exists a hyperplane to separate them; that is, there is a real number $d \in \mathbb{R}$ and a nonzero Hermitian operator $C \in \mathcal{L}$ such that:

(1) $tr(CP) + d \geq 0$ for all $P \in \mathcal{P}$; and
(2) $tr(CA) + d \leq 0$ for all $A \in \mathcal{A}$.

Thus, $d = 0$ follows immediately from these two conditions since the zero matrix 0 is in both of the closure of $\mathcal{P}$ and the closure of $\mathcal{A}$. Then the first condition implies that C is positive semi-definite. Consequently, the second condition implies that C/trC is a density matrix that makes the first statement of the lemma unsatisfied. □

Now with the help of the above lemma, we are able to find an SDP problem for realisation and synthesis of LRSMs for general quantum programs (with both angelic and demonic choices).

Theorem 6.3. *Let P be a quantum program (with angelic choice in general). Given an invariant $\mathcal{O} = \{O_l\}_{l \in L}$ for P. If $n := \dim \mathcal{H}_P > 2$, then the realizability and synthesis problem with respect to $\mathcal{O}$ is equivalent to the following constraint satisfaction problem:*

- *arbitrarily fix the value of K, then find real numbers $\epsilon > 0$ (or $> 1 - tr\left(O_{l_0}\rho_0\right)$ for multiplicative invariants), $p_{l,l'} \geq 0$ for all $l \in L_A, l \xrightarrow{I} l'$ with $\sum_{l'} p_{l,l'} = 1$ for all l, and complex Hermitian matrices R_l $(l \in L)$ satisfying the constraint:*

$$\left(\bigwedge_{l\in L} \gamma_l\right) \wedge \left(\bigwedge_{l\in L\setminus\{l^P_{out}\}} \delta_l\right)$$

where:

(1) *for each $l \in L$,*

$$\gamma_l := 0 \sqsubseteq R_l - O_l - (K-1)\cdot I;$$

(2) *for each $l \in L_A$,*

$$\delta_l := 0 \sqsubseteq R_l - \sum_{l \xrightarrow{I} l'} p_{l,l'} R_{l'} - O_l + (1-\epsilon)\cdot I;$$

(3) *for each $l \in L_D$,*

$$\delta_l := \bigwedge_{l \xrightarrow{I} l'} \left(0 \sqsubseteq R_l - R_{l'} - O_l + (1-\epsilon)\cdot I\right);$$

(4) *for each $l \in L_S \setminus \{l^P_{out}\}$,*

$$\delta_l := 0 \sqsubseteq R_l - \sum_{l \xrightarrow{\mathcal{E}} l'} \mathcal{E}^*\left(R_{l'}\right) - O_l + (1-\epsilon)\cdot I.$$

Proof. The constraint for linear functions $\eta(l,\cdot)$ can be characterised by

$$\left(\bigwedge_{l\in L} \mu_l\right) \wedge \left(\bigwedge_{l\in L\setminus\{l^P_{out}\}} \nu_l\right)$$

where μ_l and ν_l are almost the same as in the proof of Theorem 6.2. The only difference comes from the treatment of $\xi(l,\rho)$. For $l \in L_A$, we have:

$$\xi(l,\rho) \Leftrightarrow \bigvee \left\{ tr(O_l\rho) + \eta(l',\rho) \leq \eta(l,\rho) + 1 - \epsilon : l \xrightarrow{I} l' \right\}.$$

Then by using Gleason's Theorem $\eta(l,\rho)$ can be replaced by $tr(R_l\rho)$ in the constraints for Hermitian matrices R_l. We particularly consider the constraint for each $l \in L_A$:

$$\begin{aligned}
\nu_l &\Leftrightarrow (\forall \rho \in \mathcal{D}(\mathcal{H}_P) \text{ with } tr(\rho) = 1) \bigvee_{l \xrightarrow{I} l'} \left[\epsilon - 1 \leq tr\left(\rho\left(T_l - O_l - T_{l'}\right)\right)\right] \\
&\Leftrightarrow (\forall \epsilon' \in [0,\epsilon)) \bigvee_{l \xrightarrow{I} l'} \left[0 < tr\left(\rho\left(T_l - O_l - T_{l'} + (1-\epsilon')\cdot I\right)\right)\right] \\
&\Leftrightarrow 0 \sqsubset \sum_{l \xrightarrow{I} l'} p_{l,l'}\left(T_l - O_l - T_{l'} + (1-\epsilon')\cdot I\right) \text{ (for some } p_{l,l'} \geq 0).
\end{aligned}$$

Note that the last step is from Lemma 6.1. It actually means that if a (K,ϵ)-LRSM exists, then the constraints of γ_l and δ_l are satisfiable for (K,ϵ'), where ϵ' can be arbitrarily close to ϵ; and conversely, if the constraints of γ_l and δ_l are satisfiable for (K,ϵ), then a (K,ϵ)-LRSM exists. So, the proof is completed. □

Remark 6.1. *Essentially, the application of the generalized Farkas's lemma for angelic locations in the proof of the above theorem implies that if a quantum program has an LRSM, then it is finite-terminating under a probabilistic angelic scheduler: at each location $l \in L_A$, randomly choose the transitions with probability distribution $\{p_{l,l'} \mid l \xrightarrow{I} l'\}$. We note that this scheduler is not only independent of the history, but also independent of the current state.*

6.3 Discussions

In this subsection, we briefly discuss the complexity of the realisability and synthesis problem of LRSMs for quantum programs. First of all, we note that this problem is generally undecidable for programs that contains variables with infinite-dimensional state Hilbert spaces, since with Theorem 5.2 we see that it is equivalent to the undecidable termination problem. So, here we only consider the complexity of solving the constraint problems in the case of finite-dimension. The constraint problem in Theorem 6.3 can be precisely solved by Quantifier Elimination (QE), but usually requires double exponential time complexity by popular algorithms such as Cylindrical Algebraic Decomposition (CAD). However, if certain errors of the results are allowed, then better time complexity may be achieved for many cases by efficient SDP algorithms. The method is outlined into three cases as follows:

(1) For a quantum program without angelic choice, we find a $(0, 1)$-LRSM with respect to the identity invariant. The constraint problem in this case can be expressed as a standard SDP problem, which can be solved in polynomial time with respect to any given error ε. In other words, a point x_n that approximates some solution x^* of the problem (if exists) with precision ε, i.e. $\|x_n - x^*\| < \varepsilon$, can be computed within polynomial time in $\log(1/\varepsilon)$ and the size of input. Infeasibility can also be detected if no points in the ε-ball $\{y \mid \|x_n - y\| < \varepsilon\}$ satisfies the constraint.

(2) For a quantum program without angelic choice, we find a $(0, \epsilon)$-LRSM with respect to a general invariant. Since the constraint problem can be solved by SDP in polynomial time for a fixed $\epsilon > 0$, the optimal value of feasible ϵ can be approximated by binary search in polynomial time with respect to any given error.

(3) For the most general quantum programs (with both angelic and demonic choices), we find the values of probabilities $p_{l,l'}$ for all $l \in L_A$ and $l \xrightarrow{I} l'$. By uniform sampling in the value space of all $p_{l,l'}$ with a sufficiently large sampling density, feasible values can be approximated with any precision. Then one can solve the SDP constraint for each sample point. However, this algorithm requires exponential time since the number of sample points are exponential in the number of angelic transitions $l \xrightarrow{I} l'$. So, this method would be more applicable when there are few angelic locations and transitions.

The time complexity of realisability and synthesis problem of ranking super-martingales for probabilistic programs is given in [Chatterjee et al. 2016]. A comparison between the probabilistic case and the quantum case is shown in Table 1:

Table 1. A comparison between probabilistic and quantum programs

	Probabilistic	Quantum
The General Problem	PSPACE	2-EXPTIME by QE with CAD
Without Angelic Choice	PTIME	PTIME w.r.t. an error
With Angelic Choice	NP-hard	EXPTIME w.r.t. an error

7 CASE STUDIES

In this section, we apply the method based on LRSMs developed in the last two sections to prove termination of the example quantum programs presented in Subsection 2.2. In addition, we use this method to the termination analysis of quantum Bernoulli factory [Dale et al. 2015] which is a celebrated quantum algorithm for randomness generation and processing.

For each quantum program, we first reduce the realizability and synthesis of its LRSMs to a specific SDP problem by Theorem 6.2 and Theorem 6.3 and then show the experimental result of solving this SDP. In order to do it in a more effective and simpler way, we adopt the strategy which has been widely used in analysis and verification of classical programs: instead of considering all locations in L, we only set the template matrices R_l for the so-called *significant locations* l, i.e. entrances of **while** loops. Then the matrices R_l for other locations can be defined as a minimal one satisfying the SDP constraints according to the control flow (as shown by the game structure) of the program.

This section is organised as follows. The constraints for the programs from Subsection 2.2 are derived in Subsection 7.1, and the constraints for quantum Bernoulli factory are derived in Subsection 7.2. The experimental results of solving these constraints are presented in Subsection 7.3.

7.1 Illustrative Quantum Programs

In this subsection, we deal with the termination of the illustrative programs presented in Subsection 2.2 by generating the SDP constraints for them with respect to a given invariant $\{O_l \mid l \in L\}$. We only consider (K, ϵ)-LRAMs with a special value $K = 0$ without any loss of generality.

Example 7.1. *The game structure of the program* QW *of Example 2.1 has been shown in Fig 1a. We set a template matrix R_1 at the entrance l_1 of the* **while** *loop. Then the smallest matrices R_i for other location l_i can directly obtained from the constraints δ_l in Theorem 6.2 according to the control flow. They are specifically computed as follows:*

$$\begin{aligned} R_{out} &:= 0, \qquad R_3 := S^\dagger R_1 S + O_3 + (\epsilon - 1)\cdot I;\\ R_2 &:= H^\dagger R_3 H + O_2 + (\epsilon-1)\cdot I = (SH)^\dagger R_1 SH + H^\dagger O_3 H + O_2 + 2(\epsilon-1)\cdot I;\\ R_{in} &:= \mathcal{E}_0(R_1) + O_0 + (\epsilon-1)\cdot I. \end{aligned}$$

Then the constraint can be obtained at location l_1 as

$$\begin{aligned} &R_1 \sqsupseteq O_1 - I, \text{ and}\\ &R_1 \sqsupseteq M_{no}^\dagger R_2 M_{no} + M_{yes}^\dagger R_{out} M_{yes} + O_1 + (\epsilon-1)\cdot I\\ &\quad = (SHM_{no})^\dagger R_1 (SHM_{no}) + (HM_{no})^\dagger O_3 (HM_{no}) + M_{no}^\dagger O_2 M_{no}\\ &\qquad + O_1 + 2(\epsilon-1)\cdot M_{no}^\dagger M_{no} + (\epsilon-1)\cdot I. \end{aligned}$$

We particularly aim at proving the program termination for all initial states, i.e. under the identity invariant, so it suffices to find a Hermitian operator R_1 satisfying the following constraint:

$$R_1 \sqsupseteq 0 \wedge R_1 \sqsupseteq (SHM_{no})^\dagger R_1 (SHM_{no}) + 2\epsilon \cdot M_{no}^\dagger M_{no} + \epsilon \cdot I. \tag{11}$$

Example 7.2. *For the program* DQW *of Example 2.2 with demonic choice, we can obtain the constraints in a similar way according to the control flow as shown in Fig. 1b. In order to represent the matrix R_l for $l \in L_D$, we simply denote by $\{A_1, A_2, \cdots, A_n\}$ a (minimal) matrix A satisfying $A \sqsupseteq A_k$ for all $k = 1, 2, \ldots, n$. For simplicity, we only consider the constraint for identity invariant. Set the matrix R_1*

for the location l_1, then other matrices are computed as follows:

$$\begin{aligned}
R_{out} &:= 0, \qquad R_{in} := \mathcal{E}_1(R_1) + \epsilon \cdot I;\\
R_3 &:= \{W_1^\dagger R_1 W_1 + 2\epsilon \cdot I, W_2^\dagger R_1 W_2 + 2\epsilon \cdot I\};\\
R_4 &:= \{W_1^\dagger R_1 W_1 + 2\epsilon \cdot I, W_3^\dagger R_1 W_3 + 2\epsilon \cdot I\};\\
R_2 &:= \{(W_1 N_{yes})^\dagger R_1 (W_1 N_{yes}) + (W_1 N_{no})^\dagger R_1 (W_1 N_{no}) + 3\epsilon \cdot I,\\
&\qquad (W_1 N_{yes})^\dagger R_1 (W_1 N_{yes}) + (W_3 N_{no})^\dagger R_1 (W_3 N_{no}) + 3\epsilon \cdot I,\\
&\qquad (W_2 N_{yes})^\dagger R_1 (W_2 N_{yes}) + (W_1 N_{no})^\dagger R_1 (W_1 N_{no}) + 3\epsilon \cdot I,\\
&\qquad (W_2 N_{yes})^\dagger R_1 (W_2 N_{yes}) + (W_3 N_{no})^\dagger R_1 (W_3 N_{no}) + 3\epsilon \cdot I\}.
\end{aligned}$$

Then the constraint for R_1 is

$$R_1 \sqsupseteq 0 \wedge \bigwedge_{i,j} R_1 \sqsupseteq \mathcal{A}_{i,j}(R_1), \tag{12}$$

where

$$\begin{aligned}
\mathcal{A}_{i,j}(R_1) := {} & (W_i N_{yes} M_{no})^\dagger R_1 (W_i N_{yes} M_{no})\\
& + (W_j N_{no} M_{no})^\dagger R_1 (W_j N_{no} M_{no}) + 3\epsilon \cdot M_{no}^\dagger M_{no} + \epsilon \cdot I,
\end{aligned}$$

and (i, j) ranges over $\{(1,1),(1,3),(2,1),(2,3)\}$.

Example 7.3. *Similarly, we can obtain the constraint satisfaction problem for the program* ADR *of Example 2.3 with respect to a general invariant $\{O_l \mid l \in L\}$ (according to the control flow shown in Fig. 1c): find real numbers $\epsilon > 0, p \in [0,1]$ and a complex matrix R_1 satisfying*

$$(R_1 \sqsupseteq 0) \wedge (R_1 \sqsupseteq \mathcal{A}_1(R_1)) \wedge (R_1 \sqsupseteq \mathcal{A}_2(R_1)), \tag{13}$$

where

$$\begin{aligned}
\mathcal{A}_1(R_1) := {} & p[R_x(\alpha)N_0CM_1]^\dagger R_1 [R_x(\alpha)N_0CM_1] + p[N_0CM_1]^\dagger O_6 [N_0CM_1]\\
& + (1-p)[R_z(\beta)N_0CM_1]^\dagger R_1 [R_z(\beta)N_0CM_1] + (1-p)[N_0CM_1]^\dagger O_7 [N_0CM_1]\\
& + [R_x(\alpha)N_0CM_1]^\dagger R_1 [R_x(\alpha)N_0CM_1] + [N_1CM_1]^\dagger O_8 [N_1CM_1]\\
& + [N_0CM_1]^\dagger O_4 [N_0CM_1] + [N_1CM_1]^\dagger O_5 [N_1CM_1]\\
& + [CM_1]^\dagger O_3 [CM_1] + M_1^\dagger O_2 M_1 + O_1 + 4(\epsilon - 1) \cdot M_1^\dagger M_1 + (\epsilon - 1) \cdot I,
\end{aligned}$$

$$\begin{aligned}
\mathcal{A}_2(R_1) := {} & p[R_x(\alpha)N_0CM_1]^\dagger R_1 [R_x(\alpha)N_0CM_1] + p[N_0CM_1]^\dagger O_6 [N_0CM_1]\\
& + (1-p)[R_z(\beta)N_0CM_1]^\dagger R_1 [R_z(\beta)N_0CM_1] + (1-p)[N_0CM_1]^\dagger O_7 [N_0CM_1]\\
& + [R_z(\beta)N_0CM_1]^\dagger R_1 [R_z(\beta)N_0CM_1] + [N_1CM_1]^\dagger O_9 [N_1CM_1]\\
& + [N_0CM_1]^\dagger O_4 [N_0CM_1] + [N_1CM_1]^\dagger O_5 [N_1CM_1]\\
& + [CM_1]^\dagger O_3 [CM_1] + M_1^\dagger O_2 M_1 + O_1 + 4(\epsilon - 1) \cdot M_1^\dagger M_1 + (\epsilon - 1) \cdot I.
\end{aligned}$$

We note that the parameter p is involved by Theorem 6.3 to deal with the disjunction constraint for the angelic location l_4.

7.2 Quantum Bernoulli Factory

In this subsection, we consider another quantum program, namely quantum Bernoulli factory (QBF). QBF was proposed in [Dale et al. 2015] as a quantum counterpart of the classical Bernoulli Factory (CBF) [Keane and O'Brien 1994]. The aim of CBF is to use a coin with an unknown probability p of heads flipped a finite number of times to simulate a new coin that has probability $f(p)$ of heads for a given function $f : [0,1] \mapsto [0,1]$. For example, a CBF protocol for $f(p) = p^2$ is to flip the given coin twice, and consider the new coin as showing head when both flips are heads. The function $f(p) = \frac{1}{2}$ can also be achieved in two flips if their outcomes are different (and if the outcomes are the same, just try it again). QBF is designed for the same purpose, namely generating classical randomness $f(p)$. In contrast to CBF, QBF uses of quantum coins which can be in a quantum state like

$$|p\rangle = \sqrt{p}|0\rangle + \sqrt{1-p}|1\rangle.$$

Moreover, it can perform quantum operations on the quantum coins. A significant result proved by Dale et al. [2015] is that QBF can simulate a strictly larger class of functions f than those simulated by CBF. An instance is the *probability amplification function* defined by:

$$f_0 := 1 - |2p-1| = \begin{cases} 2p, & p \in [0, 1/2]; \\ 2(1-p), & p \in (1/2, 1]. \end{cases}$$

As shown in [Dale et al. 2015], CBF cannot simulate this function, but QBF can. The key of simulating the function f_0 is simulating another function

$$f_1(p) = (1 - f_0(p))^2 = (2p-1)^2$$

since the function $1-p$ and $\sqrt{p}$ can be generated by CBF. To generate f_1 by QBF, it suffices to flip the given quantum coin twice to output the state $|p\rangle^{\otimes 2}$, and then measure the state in the Bell basis:

$$\{|\Phi^{\pm}\rangle = (|00\rangle \pm |11\rangle)/\sqrt{2}, \qquad |\Psi^{\pm}\rangle = (|01\rangle \pm |10\rangle)/\sqrt{2}\}.$$

In fact, one can first measure it by $M = \{M_0 = I - M_1, M_1 = |\Phi^+\rangle\langle\Phi^+|\}$, then a quantum randomness

$$(2p-1)|\Phi^-\rangle + 2\sqrt{p(1-p)}|\Psi^+\rangle$$

of f_0 is obtained with outcome 0. The classical randomness is simply achieved by a further measurement in the basis $\{|\Phi^-\rangle, |\Psi^+\rangle\}$. This QBF protocol can be formalized as the following quantum program `QBF1` with two qubit variables q_1 and q_2:

$q_1 := |1\rangle; q_2 := |1\rangle;$
while $B[q_2] = 1$ **do** $q_1 := |p\rangle; q_2 := |p\rangle; q_1, q_2 = U[q_1, q_2];$
od

where $B = \{|0\rangle\langle 0|, |1\rangle\langle 1|\}$ is the measurement in the standard basis $|0\rangle, |1\rangle$, and U is defined by

$$U|\Phi^+\rangle = |01\rangle, \quad U|\Phi^-\rangle = |00\rangle, \quad U|\Psi^+\rangle = |10\rangle, \quad U|\Psi^-\rangle = |11\rangle.$$

It is easy to show that when the program terminates, the state of q_1 is always

$$|f_1(p)\rangle := (2p-1)|0\rangle + 2\sqrt{p(1-p)}|1\rangle,$$

e.g. by proving the correctness formula $\{I\}\texttt{QBF1}\{|f_1(p)\rangle\langle f_1(p)|\}$ in quantum Hoare logic. However, termination of the program still remains to be proved for total correctness. To this end, we use the template $tr(R_1\rho)$ for a $(0,\epsilon)$-LRSM at the entrance of the **while** loop, and then to compute R_1 by solving the constraint problem of

$$R_1 \sqsupseteq 0 \wedge R_1 \sqsupseteq N^\dagger \mathcal{E}^*(U^\dagger R_1 U)N + 2\epsilon \cdot N^\dagger N + \epsilon \cdot I, \tag{14}$$

where $N = I_1 \otimes |1\rangle\langle 1|$ and

$$\mathcal{E}^*(A) = \sum_{i,j} |ij\rangle\langle pp|A|pp\rangle\langle ij| = \langle pp|A|pp\rangle \cdot I.$$

Inequality (14) can be directly solved for any given parameter p. In fact, this parameter can even be eliminated by a simple transformation. Consider a solution R_2 of the following constraint:

$$R_2 \sqsupseteq 0 \wedge R_2 \sqsupseteq \mathcal{E}^* \left[(NU)^\dagger R(NU)\right] + 3\epsilon \cdot N^\dagger N. \tag{15}$$

We put $R_1 = N^\dagger R_2 N + \epsilon \cdot I$, then it is easy to verify that R_1 must be a solution of inequality (14). So, it suffices to solve the constraint (15), and more importantly, the parameter p is useless in computation of the super-operator $\mathcal{T}(R) := \mathcal{E}^*(\left[(NU)^\dagger R(NU)\right])$. Therefore, termination of the program as well as the expected running time is independent of the value of p.

Furthermore, the process of generating $f_0(p)$, as proposed in [Dale et al. 2015], can be formalized as a quantum program QBF0, in which QBF1 is invoked as a subprogram. Specifically, there are four quantum variables in the program: a qubit variable c with basis states $\{|h\rangle, |t\rangle\}$ to represent the results of head and tail of the coin; a quantum integer variable z for a random walk; and two auxiliary qubit variables q_1, q_2 for the subprogram QBF1. The program is presented as follows:

QBF0 $\equiv$ $c := |h\rangle; z := |0\rangle; q_1 := |0\rangle; q_2 := |0\rangle;$
QBF1;
if $B[q_1] = 0 \rightarrow c := |t\rangle; z := |0\rangle;$
$\square$ $= 1 \rightarrow c := |h\rangle; z := |1\rangle;$
while $M[z] = 1$ **do**
QBF1;
if $B[q_1] = 0 \rightarrow c := X[c];$
$\square$ $= 1 \rightarrow q_1 := \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle);$
if $B[q_1] = 0 \rightarrow z := U[z];$
$\square$ $= 1 \rightarrow z := D[z];$
fi
fi od

Here, $M = \{M_0, M_1\}$ is a measurement on z with $M_0 = |0\rangle\langle 0|$, $M_1 = I - M_0$; U and D are the raising and lowering operators defined by

$$U|n\rangle = |n+1\rangle, \quad D|n\rangle = |n-1\rangle \text{ for all integers } n,$$

respectively. This program will output $c := |h\rangle$ with probability $f_0(p)$ and $c := |t\rangle$ with probability $1 - f_0(p)$.

With the constraint (14) for QBF1, it is not difficult to further compute the constraint for QBF0. However, since the state Hilbert space of QBF0 is infinite-dimensional, the constraint can hardly be solved by directly using the SDP solver. In fact, due to the undecidability of general termination problem, it is impossible to find an universal template with a finite number of parameters. In this scenario, templates with more detailed structures are usually adopted to deal with special cases. Here, we prove termination of QBF0 in this way; namely, by assigning a specific structure to the template matrix R at each location. Put $R = \sum_n |n\rangle\langle n| \otimes S_n$, where S_n are positive operators on variables c, q_1 and q_2. Then the constraint of an infinite-dimensional matrix R can be transformed into a constraint for the finite-dimensional matrix S_n with a parameter n. We find that termination

of QBF0 can be proved with $S_n := \sqrt{n} \cdot S$, where S is a finite-dimensional template which can be solved by SDP. Moreover, the expected running time ET has an $O(\sqrt{n})$ upper bound for the initial states with $z := |n\rangle$.

7.3 Experimental Results

We have implemented the termination analysis procedures based on LRSM generation in *Matlab* for the example programs in the previous two subsections. The procedures first transform the corresponding constrain problem into a standard SDP form, and then solve it by employing the solver *SDPT3*.The output of the solver will be checked again with the constraint due to the allowed errors of the solver. In particular, we set the objective function of the SDP as

$$\frac{\eta(l_{in}, \rho_0)}{\epsilon} = \frac{tr(\rho_0 R_{in})}{\epsilon}$$

which would be minimized. Thus, according to the termination theorems, it is actually an upper bound of the expected running time ET of the quantum program which can be obtained whenever a solution has been successfully found. When the solver finds infeasibility of the problem, we will try to find a counter example, i.e., an non-terminating initial state. The procedures executed on a 64-bit Windows computer with a 2.80GHz Intel Core-i7 processor and 16GB of RAM.

Table 2. Evaluation results of different methods

Name(Para)		Para Value	Dimension	Feasibility	ET	Time (sec)
QW(n)		4	8	yes	10	0.13
		32	64	yes	94	5.57
		64	160	yes	190	111.18
		100	200	yes	298	978.34
		110	220	–	–	TO
DQW(n)		4	8	yes	9	0.46
		32	64	yes	3.94×10^6	1450.00
		50	100	yes	3.06×10^4	426.50
		55	110	–	–	TO
ADR(α, β)	I	$(\pi/3, \pi/2)$	4	yes	10	3.96
		$(\pi/2, 2\pi/3)$	4	yes	10	3.93
		$(0, \pi/2)$	4	no	–	3.89
	O	$(0, \pi/2)$	4	yes	4	3.88
QBF1(p)	Eq. (14)	0.2	4	yes	7	0.08
		0.5	4	yes	7	0.09
		0.9	4	yes	7	0.08
	Eq.(15)	$*$	4	yes	7	0.08
QBF0(p)		0.5	4	yes	14.4	0.09

Legends: the first column indicates the names of the programs, and the parameters (in the parentheses) to which the value is assigned in the second column. The third column specifies the dimensions of the problem. The fourth column show feasibility output of the solver, and if it is "yes" (resp. "no") an upper bound of the expected running time is given (resp. is marked by "–") in the fifth column. The last column gives the time (in seconds) taken. Timeouts here are set to 1800 sec (30 minutes) and are represented by TO; and in this case both of the fourth and fifth columns are marked by "–".

The experimental results of the case studies are illustrated in Table 2 and are specifically explained as follows:

(1) Termination of the program QW is proved by the procedures even for the case of more than one hundred of dimension, which indicates the efficiency of our method for small-scale quantum programs. When $n = 110$ the procedure also solves the problem but runs more than 30 minutes. The derived upper bound of ET is precisely $3n - 2$, which is tight. It coincides with the hand-proof result using the method of [Ying and Feng 2010]. For DQW, termination is perfectly proved for $n \leq 32$. The case of $n = 50$ can also be proved feasible but with an unavoidable error (although quite small), so the corresponding bound of ET is of low confidence. The procedure fails for $n = 55$ due to the out of memory. It seems still possible to improve the efficiency by further refinement, such as design a specific SDP solver for the problems in their original form, since representing the constraint problem in a standard SDP form requires much more memory as well as more processing time.
(2) Termination of program ADR can be proved with the identity invariant (marked by I in Table 2) for almost all parameters (α, β) except the cases of $\alpha = 0$. A counter example (i.e., a non-terminating initial state) can be generated by solving the dual SDP. However, the problem with this kind of parameters can be further solved with a non-trivial invariant (marked by O in Table 2) such that at the initial location $O_0 = |11\rangle\langle 11|$. It actually implies that the program is finite-terminating with the given initial state $|1\rangle_q|1\rangle_c$.
(3) Termination of the Quantum Bernoulli Factory program is proved by the procedures too. For QBF1, the constraint problem (14) dependent of p and the problem (15) independent of p are both solved by the procedure and with almost the same experimental results. For QBF0, we simply choose $p = 0.5$, but the solution stands uniformly for the parameter n; for example, the upper bound of ET is shown for $n = 1$, and should be multiplied by $\sqrt{n}$ for general n.

8 CONCLUSION

In this paper, we presented an algorithmic solution to the termination problems of quantum programs by generalising the constraint-based approach developed in [Colón et al. 2001, 2003; Podelski and Rybalchenko 2004] for classical and probabilistic programs [Chakarov and Sankaranarayanan 2013; Chatterjee et al. 2016]. The main new ideas proposed in this paper that are not needed in the case of classical and probabilistic programs are: (1) using the fundamental Gleason's theorem in quantum mechanics to guide the choices of templates of LRSMs; and (2) a generalised Farkas's lemma in terms of observables (Hermitian operators) in quantum physics. The former shows an interesting connection between LRSMs and the representation of states of quantum systems. We believe that the latter will find more applications in analysis and verification of quantum programs, as the classical Farkas's lemma did for classical and probabilistic programs.

For future studies, we are going to carefully examine the structure of those SDP problems in the termination analysis of quantum programs in order to find more efficient algorithms for solving them so that we can deal with larger quantum programs. Since the dynamics of quantum systems are always modelled by linear operators, it is especially desirable to prove the completeness of our method for general quantum programs (if possible), as what was done in [Podelski and Rybalchenko 2004] for classical linear programs. Recently, the notion of weakest precondition has been generalised in [Kaminski et al. 2016, 2017; Olmedo et al. 2016] for reasoning about termination and the expected running time of probabilistic programs. So, another interesting research topic is to extend the weakest precondition-based reasoning to the quantum setting.

ACKNOWLEDGMENTS

This paper was partly supported by the National Natural Science Foundation of China (Grant No: 61502467), the Australian Research Council (Grant No: DP160101652) and the Key Research Program of Frontier Sciences, Chinese Academy of Sciences.

REFERENCES

D. Aharonov, A. Ambainis, J. Kempe and U. Vazirani, Quantum walks on graphs, In: *Proceedings of the 33rd ACM Symposium on Theory of Computing (STOC)*, 2001, 50-59.

T. Altenkirch and J. Grattage, A functional quantum programming language, In: *Proceedings of the 20th IEEE Symposium on Logic in Computer Science (LICS)*, 2005, 249-258.

A. Baltag and S. Smets, LQP: The dynamic logic of quantum information, *Mathematical Structures in Computer Science* 16(2006)491-525.

J. Barry, D. T. Barry and S. Aaronson, Quantum partially observable Markov decision processes, *Physical Review A*, 90(2014) art. no. 032311.

O. Bournez and F. Garnier, Proving positive almost-sure termination, In: *Proceedingds of the 16th International Conference on Rewriting Techniques and Applications (RTA)*, 2005, Springer LNCS 3467, 323-337.

A. R. Bradley, Z. Manna and H. B. Sipma, Linear ranking with reachability, In: *Proceedings of the 17th International Conference on Computer Aided Verification (CAV)*, 2005, Springer LNCS 3576, 491-504.

O. Brunet and P. Jorrand, Dynamic quantum logic for quantum programs, *International Journal of Quantum Information*, 2(2004)45-54.

R. Chadha, P. Mateus and A. Sernadas, Reasoning about imperative quantum programs, *Electronic Notes in Theoretical Computer Science*, 158(2006)19-39.

A. Chakarov and S. Sankaranarayanan, Probabilistic program analysis with martingales, In: *Proceedings of the 25th International Conference on Computer Aided Verification (CAV)*, 2013, Springer LNCS 8044, 511-526.

K. Chatterjee, H. F. Fu, P. Novotný and R. Hasheminezhad, Algorithmic analysis of qualitative and quantitative termination problems for affine probabilistic programs, In: *Proceedings of the 43rd Annual ACM Symposium on Principles of Programming Languages (POPL)*, 2016, 327-342.

M. A. Colón and H. B. Sipma, Synthesis of linear ranking functions, In: *Proceedings of the 7th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, 2001, 67-81.

M. A. Colón, S. Sankaranarayanan and H. B. Sipma, Linear invariant generation using non-linear constraint solving, In: *Proceedings of the 15th International Conference on Computer Aided Verification (CAV)*, 2003, Springer LNCS, 420-433.

H. Dale, D. Jennings and T. Rudolph, Provable quantum advantage in randomness processing, *Nature Communications* 6(2015), art. no. 8203.

H. Derksen and J. Weyman, Quiver representations, *Notices of the American Mathematical Society* 52 (2005) 200-206.

A. Dvurečenskij, *Gleason's Theorem and Its Applications*, Kluwer, 1993.

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.

Y. Feng, N. K. Yu and M. S. Ying, Model checking quantum Markov chains, *Journal of Computer and System Sciences* 79(2013)1181-1198.

L. M. F. Fioriti and H. Hermanns, Probabilistic termination: soundness, completeness, and compositionality. In: *Proceedings of the 42nd Annual ACM Symposium on Principles of Programming Languages (POPL)*, 2015, 489-501.

R. W. Floyd, Assigning meanings to programs, In: *Proceedings of the Symposium on Mathematical Aspects of Computer Science*, 1967, 19-33.

F. G. Foster, On the stochastic matrices associated with certain queuing processes, *The Annals of Mathematical Statistics* 24(1953)355-360.

S. Gay, Quantum programming languages: survey and bibliography, *Mathematical Structures in Computer Science* 16(2006)581-600.

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)*, Springer LNCS 5123, 2008, 543-547.

A. M. Gleason, Measures on the closed subspaces of a Hilbert space, Journal of Mathematics and Mechanics 6(1957)885-893.

A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger and B. Valiron, Quipper: A scalable quantum programming language, In: *Proceedings of the 34th ACM Conference on Programming Language Design and Implementation (PLDI)*, 2013, 333-342.

S. Gudder, Quantum Markov chains, *Journal of Mathematical Physics* 49(2008) art. no. 072105.

A. JavadiAbhari, 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.

A. JavadiAbhari, S. Patil, D. Kudrow, J. Heckey, A. Lvov, F. T. Chong and M. Martonosi, ScaffCC: Scalable compilation and analysis of quantum programs, *Parallel Computing*, 45(2015)2-17.

Y. Kakutani, A logic for formal verification of quantum programs, In: *Proceedings of the 13th Asian Computing Science Conference (ASIAN 2009)*, Springer LNCS 5913, 79-93.

B. L. Kaminski, J. Katoen, C. Matheja and F. Olmedo, Weakest precondition reasoning for expected run-times of probabilistic programs, In: *Proceedings of the 25th European Symposium on Programming Languages and Systems (ESOP 2016)*, Springer

LNCS 9632, 364-389.
B. L. Kaminski and J. Katoen, A weakest pre-expectation semantics for mixed-sign expectations, In: *Proceedings of the 32nd ACM/IEEE Symposium on Logic in Computer Science (LICS 2017)*, 1-12.
M. S. Keane and G. L. O'Brien, A Bernoulli factory, *ACM Transactions on Modelling and Computer Simulation* 4(1994) 213-219.
Y. J. Li, N. K. Yu and M. S. Ying, Termination of nondeterministic quantum programs, *Acta Informatica* 51(2015)1-24.
T. Liu, Y. J. Li, 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*
G. Mitchison and R. Jozsa, Counterfactual computation, *Proceedings of the Royal Society of London A* 457(2001)1175-1193.
F. Olmedo, B. L. Kaminski, J. Katoen and C. Matheja, Reasoning about recursive probabilistic programs, In: *Proceedings of the 31st ACM/IEEE Symposium on Logic in Computer Science (LICS 2016)*, 672-681.
B. Ömer, *Structured Quantum Programming*, Ph.D thesis, Technical University of Vienna, 2003.
J. Paykin, R. Rand and S. Zdancewic, QWIRE: a core language for quantum circuits, In: *Proceedings of 44th ACM Symposium on Principles of Programming Languages (POPL)*, 2017, 846-858.
A. Podelski and A. Rybalchenko, A complete method for the synthesis of linear ranking functions, In: *Proceedings of the 5th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI)*, 2004, 239-251.
A. Rybalchenko, Constraint solving for program verification: theory and practice by example, In:*Proceedings of the 22nd International Conference on Computer Aided Verification (CAV)*, 2010, 57-71.
A Sabry, Modelling quantum computing in Haskell, *Proceedings of the 2003 ACM SIGPLAN workshop on Haskell*, 39-49.
J. W. Sanders and P. Zuliani, Quantum programming, In: *Proceedings of 5th International Conference on Mathematics of Program Construction (MPC)*, Springer LNCS 1837, Springer 2000, 88-99.
P. Seinger, A brief survey of quantum programming languages, In: *Proc. of 7th International Symposium on Functional and Logic Programming*, Springer LNCS 2998, 2004, 1-6.
P. Selinger, Towards a quantum programming language, *Mathematical Structures in Computer Science 14* (2004), 527-586.
D. Wecker and K. M. Svore, LIQUi|⟩: A software design architecture and domain-specific language for quantum computing, http://research.microsoft.com/pubs/209634/1402.4467.pdf.
M. S. Ying, Floyd-hoare logic for quantum programs, *ACM Transactions on Programming Languages and Systems* 33(2011), 1-49.
M. S. Ying, *Foundations of Quantum Programming*, Morgan-Kaufmann, 2016.
M. S. Ying and Y. Feng, Quantum loop programs, *Acta Informatica 47* (2010), 221-250.
M. S. Ying, Y. J. Li, N. K. Yu and Y. Feng, Model-checking linear-time properties of quantum systems, *ACM Transactions on Computational Logic*, 15(2014), art. no. 22.
M. S. Ying, S. G. Ying and X. D. Wu, Invariants of quantum programs: characterisations and generation, In: *Proceedings of the 44th ACM Symposium on Principles of Programming Languages (POPL)*, 2017, 818-832.
M. S. Ying, N. K. Yu, Y. Feng and R. Y. Duan, Verification of quantum programs, *Science of Computer Programming* 78(2013)1679-1700.
S. G. Ying, Y. Feng, N. K. Yu and M. S. Ying, Reachability probabilities of quantum Markov chains, In: *Proceedings of the 24th International Conference on Concurrency Theory (CONCUR)*, 2013, 334-348.
S. G. Ying and M. S. Ying, Reachability analysis of quantum Markov decision processes, *arXiv*:1406.6146
N. K. Yu and M. S. Ying, Reachability and termination analysis of concurrent quantum programs, In: *Proceedings of the 23th International Conference on Concurrency Theory (CONCUR)*, 2012, 69-83.
P. Zuliani, Nondeterministic quantum programming, In: *Proceedings of the 2nd International Workshop on Quantum Programming Languages (QPL)*, 2004, 179-195.