Hoare Logic for Quantum Programs

Mingsheng Ying

University of Technology, Sydney
and
Tsinghua University

SamsonFest, May 28-30, 2013
Abramsky Conjecture:
For every $n > 2$, every $n$–partite entangled state is logically non-local
Happy Birthday, Samson!
Outline

Introduction

Syntax of Quantum Programs

Operational Semantics

Denotational Semantics

Correctness Formulas

Proof System for Quantum Programs

Conclusion
Outline

Introduction

Syntax of Quantum Programs

Operational Semantics

Denotational Semantics

Correctness Formulas

Proof System for Quantum Programs

Conclusion
Quantum Programming

- Quantum Random Access Machine (QRAM) model

E. H. Knill, *Conventions for quantum pseudocode*, Technical Report, Los Alamos National Laboratory, 1996.
Quantum Programming

- Quantum Random Access Machine (QRAM) model
- A set of conventions for writing quantum pseudocode

E. H. Knill, *Conventions for quantum pseudocode*, Technical Report, Los Alamos National Laboratory, 1996.
Quantum Programming Languages

- qGCL: quantum extension of Dijkstra’s Guarded Command Language [1]

[1] J. W. Sanders and P. Zuliani, Quantum programming, Mathematics of Program Construction, 2000.
[2] B. Ömer, Structural quantum programming, Ph.D. Thesis, Technical University of Vienna, 2003.
[3] P. Selinger, Towards a quantum programming language, Mathematical Structures in Computer Science, 14(2004)
Quantum Programming Languages

- qGCL: quantum extension of Dijkstra’s Guarded Command Language [1]
- QCL: high-level, architecture independent, with a syntax derived from classical procedural languages like C or Pascal [2]

[1] J. W. Sanders and P. Zuliani, Quantum programming, Mathematics of Program Construction, 2000.
[2] B. Ömer, Structural quantum programming, Ph.D. Thesis, Technical University of Vienna, 2003.
[3] P. Selinger, Towards a quantum programming language, Mathematical Structures in Computer Science, 14(2004)
Quantum Programming Languages

- qGCL: quantum extension of Dijkstra’s Guarded Command Language [1]
- QCL: high-level, architecture independent, with a syntax derived from classical procedural languages like C or Pascal [2]
- QPL: functional in nature, with high-level features (loops, recursive procedures, structured data types) [3]

[1] J. W. Sanders and P. Zuliani, Quantum programming, Mathematics of Program Construction, 2000.
[2] B. Ömer, Structural quantum programming, Ph.D. Thesis, Technical University of Vienna, 2003.
[3] P. Selinger, Towards a quantum programming language, Mathematical Structures in Computer Science, 14(2004)
Quantum Programming Languages

- Scaffold: Quantum programming language (Princeton, UCS, UCSB) [1]

[1] A. J. Abhari, et al., Scaffold: Quantum Programming Language, Technical Report, Department of Computer Science, Princeton University, 2012.
[2] A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger and B. Valiron, Quipper: A Scalable Quantum Programming Language, PLDI, 2013.
Quantum Programming Languages

- Scaffold: Quantum programming language (Princeton, UCS, UCSB) [1]
- Quipper: A Scalable Quantum Programming Language [2]

[1] A. J. Abhari, et al., Scaffold: Quantum Programming Language, Technical Report, Department of Computer Science, Princeton University, 2012.
[2] A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger and B. Valiron, Quipper: A Scalable Quantum Programming Language, PLDI, 2013.
Floyd-Hoare Logic for Quantum Programs

[1] O. Brunet and P. Jorrand, Dynamic quantum logic for quantum programs, *International Journal of Quantum Information*, 2(2004)
[2] A. Baltag and S. Smets, LQP: the dynamic logic of quantum information, *Mathematical Structures in Computer Science*, 16(2006)
[3] Y. Kakutani, A logic for formal verification of quantum programs, *Proceedings of 13th Asian conference on Advances in Computer Science*, 2009

[4] M. S. Ying, *TOPLAS* 39(2011), art. no. 19
[4′] M. S. Ying, arXiv (quant-ph): 0906.4586
Outline

Introduction

Syntax of Quantum Programs

Operational Semantics

Denotational Semantics

Correctness Formulas

Proof System for Quantum Programs

Conclusion
Syntax

A “core” language for imperative quantum programming

- A countably infinite set $Var$ of quantum variables
Syntax

A “core” language for imperative quantum programming

- A countably infinite set $Var$ of quantum variables
- Two basic data types: $\text{Boolean, integer}$
Syntax, Continued

Hilbert spaces denoted by **Boolean** and **integer**:

\[ \mathcal{H}_{\text{Boolean}} = \mathcal{H}_2, \]

\[ \mathcal{H}_{\text{integer}} = \mathcal{H}_\infty. \]

Space \( l_2 \) of square summable sequences

\[ \mathcal{H}_\infty = \{ \sum_{n=-\infty}^{\infty} \alpha_n |n\rangle : \alpha_n \in \mathbb{C} \text{ for all } n \in \mathbb{Z} \text{ and } \sum_{n=-\infty}^{\infty} |\alpha_n|^2 < \infty \}, \]

where \( \mathbb{Z} \) is the set of integers.
Syntax, Continued

A quantum register is a finite sequence of distinct quantum variables.

State space of a quantum register \(\bar{q} = q_1, ..., q_n\):

\[
\mathcal{H}_{\bar{q}} = \bigotimes_{i=1}^{n} \mathcal{H}_{q_i}.
\]
Syntax, Continued

Quantum extension of classical \textbf{while}-programs:

\begin{align*}
S ::= \texttt{skip} & \mid q := 0 \mid \bar{q} := U\bar{q} \mid S_1; S_2 \mid \texttt{measure} \ M[\bar{q}] : S \\
& \mid \texttt{while} \ M[\bar{q}] = 1 \ \texttt{do} \ S
\end{align*}

$\triangleright$ $q$ is a quantum variable and $\bar{q}$ a quantum register
Syntax, Continued

Quantum extension of classical \texttt{while}-programs:

\[ S ::= \texttt{skip} \mid q := 0 \mid \overline{q} := U\overline{q} \mid S_1 ; S_2 \mid \texttt{measure } M[\overline{q}] : \overline{S} \]
\[ \mid \texttt{while } M[\overline{q}] = 1 \texttt{ do } S \]

- \( q \) is a quantum variable and \( \overline{q} \) a quantum register
- \( U \) in the statement “\( \overline{q} := U\overline{q} \)” is a unitary operator on \( \mathcal{H}_{\overline{q}} \)
Syntax, Continued

Quantum extension of classical **while**-programs:

\[
S ::= \text{skip} \mid q := 0 \mid \bar{q} := U\bar{q} \mid S_1; S_2 \mid \text{measure } M[\bar{q}] : \overline{S} \\
\mid \text{while } M[\bar{q}] = 1 \text{ do } S
\]

- \(q\) is a quantum variable and \(\bar{q}\) a quantum register
- \(U\) in the statement "\(\bar{q} := U\bar{q}\)" is a unitary operator on \(\mathcal{H}_{\bar{q}}\)
- statement **measure**:
Syntax, Continued

Quantum extension of classical while-programs:

\[
S ::= \text{skip} \mid q := 0 \mid \bar{q} := U\bar{q} \mid S_1; S_2 \mid \text{measure } M[\bar{q}] : \bar{S} \\
\mid \text{while } M[\bar{q}] = 1 \text{ do } S
\]

- $q$ is a quantum variable and $\bar{q}$ a quantum register
- $U$ in the statement “$\bar{q} := U\bar{q}$” is a unitary operator on $\mathcal{H}_{\bar{q}}$
- statement measure:
  - $M = \{M_m\}$ is a measurement on the state space $\mathcal{H}_{\bar{q}}$ of $\bar{q}$
Syntax, Continued

Quantum extension of classical \texttt{while}-programs:

\[
S ::= \texttt{skip} \mid q := 0 \mid \bar{q} := U\bar{q} \mid S_1 ; S_2 \mid \texttt{measure} \ M[\bar{q}] : S \mid \texttt{while} \ M[\bar{q}] = 1 \ \texttt{do} \ S
\]

- \(q\) is a quantum variable and \(\bar{q}\) a quantum register
- \(U\) in the statement “\(\bar{q} := U\bar{q}\)” is a unitary operator on \(\mathcal{H}_{\bar{q}}\)
- statement \texttt{measure}:
  - \(M = \{M_m\}\) is a measurement on the state space \(\mathcal{H}_{\bar{q}}\) of \(\bar{q}\)
  - \(S = \{S_m\}\) is a set of quantum programs such that each outcome \(m\) of measurement \(M\) corresponds to \(S_m\)
Syntax, Continued

Quantum extension of classical `while`-programs:

\[
S ::= \text{skip} \mid q := 0 \mid \overline{q} := U\overline{q} \mid S_1; S_2 \mid \text{measure } M[\overline{q}] : \overline{S} \mid \text{while } M[\overline{q}] = 1 \text{ do } S
\]

- \( q \) is a quantum variable and \( \overline{q} \) a quantum register
- \( U \) in the statement “\( \overline{q} := U\overline{q} \)” is a unitary operator on \( \mathcal{H}_{\overline{q}} \)
- statement `measure`:
  - \( M = \{M_m\} \) is a measurement on the state space \( \mathcal{H}_{\overline{q}} \) of \( \overline{q} \)
  - \( S = \{S_m\} \) is a set of quantum programs such that each outcome \( m \) of measurement \( M \) corresponds to \( S_m \)
- statement `while`: \( M = \{M_0, M_1\} \) is a yes-no measurement on \( \mathcal{H}_{\overline{q}} \)
Outline

Introduction

Syntax of Quantum Programs

**Operational Semantics**

Denotational Semantics

Correctness Formulas

Proof System for Quantum Programs

Conclusion
Notation

- A quantum configuration is a pair
  \[ \langle S, \rho \rangle \]
Notation

- A quantum configuration is a pair

\[ \langle S, \rho \rangle \]

- \( S \) is a quantum program or \( E \) (the empty program)
Notation

- A quantum configuration is a pair
  \[ \langle S, \rho \rangle \]

- $S$ is a quantum program or $E$ (the empty program)
- $\rho \in \mathcal{D}^-(\mathcal{H}_{\text{all}})$ is a partial density operator on $\mathcal{H}_{\text{all}}$ — (global) state of quantum variables
Notation

- A quantum configuration is a pair 
  \[ \langle S, \rho \rangle \]

- \( S \) is a quantum program or \( E \) (the empty program)
- \( \rho \in \mathcal{D}^{-}(\mathcal{H}_{\text{all}}) \) is a partial density operator on \( \mathcal{H}_{\text{all}} \) — (global) state of quantum variables
- Tensor product of the state spaces of all quantum variables:
  \[ \mathcal{H}_{\text{all}} = \bigotimes_{\text{all } q} \mathcal{H}_q \]
Notation

- A quantum configuration is a pair
  \[ \langle S, \rho \rangle \]

- \( S \) is a quantum program or \( E \) (the empty program)
- \( \rho \in \mathcal{D}^- (\mathcal{H}_{\text{all}}) \) is a partial density operator on \( \mathcal{H}_{\text{all}} \) — (global) state of quantum variables
- Tensor product of the state spaces of all quantum variables:
  \[ \mathcal{H}_{\text{all}} = \bigotimes_{\text{all } q} \mathcal{H}_q \]

- Transitions between configurations:
  \[ \langle S, \rho \rangle \rightarrow \langle S', \rho' \rangle \]
Operational Semantics

(Skip) \[ \langle \text{skip}, \rho \rangle \rightarrow \langle E, \rho \rangle \]

(Initialization) \[ \langle q := 0, \rho \rangle \rightarrow \langle E, \rho_0^q \rangle \]

- \( type(q) = \text{Boolean}: \)
  \[ \rho_0^q = |0\rangle_q \langle 0| \rho |0\rangle_q \langle 0| + |0\rangle_q \langle 1| \rho |1\rangle_q \langle 0| \]
Operational Semantics

\((\text{Skip})\)
\[
\langle \text{skip}, \rho \rangle \rightarrow \langle E, \rho \rangle
\]

\((\text{Initialization})\)
\[
\langle q := 0, \rho \rangle \rightarrow \langle E, \rho^q_0 \rangle
\]

- \(\text{type}(q) = \text{Boolean}:\)
\[
\rho^q_0 = |0\rangle_q |0\rangle |0\rangle_{q} |0\rangle + |0\rangle_q |1\rangle |1\rangle_{q} |0\rangle
\]

- \(\text{type}(q) = \text{integer}:\)
\[
\rho^q_0 = \sum_{n=-\infty}^{\infty} |0\rangle_q |n\rangle |n\rangle_{q} |0\rangle
\]
Operational Semantics, Continued

(Unitary Transformation) \[ \langle \bar{q} := Uq, \rho \rangle \rightarrow \langle E, U\rho U^{\dagger} \rangle \]

(Sequential Composition) \[ \langle S_1, \rho \rangle \rightarrow \langle S'_1, \rho' \rangle \]
\[ \langle S_1; S_2, \rho \rangle \rightarrow \langle S'_1; S_2, \rho' \rangle \]

Convention: \( E; S_2 = S_2 \).

(Measurement) \[ \langle \text{measure } M[\bar{q}] : S, \rho \rangle \rightarrow \langle S_m, M_m \rho M_m^\dagger \rangle \]

for each outcome \( m \)
Operational Semantics, Continued

(Loop 0) \[ \langle \text{while } M[q] = 1 \text{ do } S, \rho \rangle \rightarrow \langle E, M_0 \rho M^+_0 \rangle \]

(Loop 1) \[ \langle \text{while } M[q] = 1 \text{ do } S, \rho \rangle \rightarrow \langle S; \text{while } M[q] = 1 \text{ do } S, M_1 \rho M^+_1 \rangle \]
Outline

Introduction

Syntax of Quantum Programs

Operational Semantics

Denotational Semantics

Correctness Formulas

Proof System for Quantum Programs

Conclusion
Definition

Semantic function of quantum program $S$:

$\llbracket S \rrbracket : \mathcal{D}^-(\mathcal{H}_{\text{all}}) \rightarrow \mathcal{D}^-(\mathcal{H}_{\text{all}})$

is defined by

$\llbracket S \rrbracket (\rho) = \sum \{ |\rho' : \langle S, \rho \rangle \rightarrow^* \langle E, \rho' \rangle | \}$

for all $\rho \in \mathcal{D}^-(\mathcal{H}_{\text{all}})$. 
Representation of Semantic Function

1. $[[\text{skip}] (\rho) = \rho$. 
Representation of Semantic Function

1. $\llbracket \text{skip} \rrbracket(\rho) = \rho$. 
Representation of Semantic Function

1. \( \llbracket \text{skip} \rrbracket (\rho) = \rho \).
2.  
   - \( \text{type}(q) = \text{Boolean} \):
     \[
     \llbracket q := 0 \rrbracket (\rho) = |0\rangle_q \langle 0|\rho|0\rangle_q\langle 0| + |0\rangle_q \langle 1|\rho|1\rangle_q\langle 0|.
     \]
     \( \text{type}(q) = \text{integer} \):
     \[
     \llbracket q := 0 \rrbracket (\rho) \sum_{n=-\infty}^{\infty} |0\rangle_q \langle n|\rho|n\rangle_q\langle 0|.
     \]
Representation of Semantic Function

1. $[[\text{skip}]](\rho) = \rho$.

2. ▶ $\text{type}(q) = \text{Boolean}$:

   $[[q := 0]](\rho) = |0\rangle_q \langle 0| \rho |0\rangle_q \langle 0| + |0\rangle_q \langle 1| \rho |1\rangle_q \langle 0|.$

   $\text{type}(q) = \text{integer}$:

   $[[q := 0]](\rho) \sum_{n=-\infty}^{\infty} |0\rangle_q \langle n| \rho |n\rangle_q \langle 0|.$

3. $[[\overline{q} := U\overline{q}]](\rho) = U\rho U^\dagger.$
Representation of Semantic Function

1. \([\text{skip}]\)(\(\rho\)) = \(\rho\).

2. \(\triangleright\) type\((q) = \text{Boolean}:\)

\[
[q := 0](\rho) = |0\rangle_q \langle 0| \rho |0\rangle_q \langle 0| + |0\rangle_q |1\rangle_q |0\rangle_q \langle 0|.
\]

\(\text{type}(q) = \text{integer}:\)

\[
[q := 0](\rho) = \sum_{n=-\infty}^{\infty} |0\rangle_q |n\rangle_q \langle n| \rho |n\rangle_q \langle 0|.
\]

3. \([\overline{q} := U\overline{q}]\)(\(\rho\)) = \(U\rho U^\dagger\).

4. \([S_1; S_2]\)(\(\rho\)) = \([S_2]\)(\([S_1]\)(\(\rho\))).
Representation of Semantic Function

1. $\llbracket \text{skip} \rrbracket(\rho) = \rho$.

2. $\triangleright$ \text{type}(q) = \text{Boolean}:

$$\llbracket q := 0 \rrbracket(\rho) = |0\rangle_q <0|\rho|0\rangle_q <0| + |0\rangle_q <1|\rho|1\rangle_q <0|.$$ 

\text{type}(q) = \text{integer}:

$$\llbracket q := 0 \rrbracket(\rho) \sum_{n=-\infty}^{\infty} |0\rangle_q <n|\rho|n\rangle_q <0|.$$ 

3. $\llbracket \bar{q} := U\bar{q} \rrbracket(\rho) = U\rho U^\dagger$.

4. $\llbracket S_1; S_2 \rrbracket(\rho) = \llbracket S_2 \rrbracket(\llbracket S_1 \rrbracket(\rho))$.

5. $\llbracket \text{measure } M[\bar{q}] : S \rrbracket(\rho) = \sum_m \llbracket S_m \rrbracket(M_m\rho M_m^\dagger)$.
Representation of Semantic Function

1. $\llbracket \text{skip} \rrbracket (\rho) = \rho$.
2. ▶ $\text{type}(q) = \text{Boolean}$:
   
   $\llbracket q := 0 \rrbracket (\rho) = |0\rangle_q 0 \rho |0\rangle_q 0 + |0\rangle_q 1 \rho |1\rangle_q 0$.

   $\text{type}(q) = \text{integer}$:

   $\llbracket q := 0 \rrbracket (\rho) = \sum_{n=-\infty}^{\infty} |0\rangle_q n \rho |n\rangle_q 0$.

3. $\llbracket \bar{q} := U\bar{q} \rrbracket (\rho) = U \rho U^\dagger$.
4. $\llbracket S_1; S_2 \rrbracket (\rho) = \llbracket S_2 \rrbracket (\llbracket S_1 \rrbracket (\rho))$.
5. $\llbracket \text{measure } M[\bar{q}] : S \rrbracket (\rho) = \sum_m \llbracket S_m \rrbracket (M_m \rho M_m^\dagger)$.
6. $\llbracket \text{while } M[\bar{q}] = 1 \text{ do } S \rrbracket (\rho) = \bigvee_{n=0}^{\infty} \llbracket (\text{while } M[\bar{q}] = 1 \text{ do } S)^n \rrbracket (\rho)$.
Notation

\((\text{while } M[\overline{q}] = 1 \text{ do } S)^0 = \Omega,\)
\((\text{while } M[\overline{q}] = 1 \text{ do } S)^{n+1} = \text{measure } M[\overline{q}] : \overline{S},\)

where:

- \(\Omega\) is a program such that \(\llbracket\Omega\rrbracket = 0\) for all \(\rho \in \mathcal{D}(\mathcal{H})\)
Notation

\[
(\text{while } M[\bar{q}] = 1 \text{ do } S)^0 = \Omega,
\]
\[
(\text{while } M[\bar{q}] = 1 \text{ do } S)^{n+1} = \text{measure } M[\bar{q}] : \bar{S},
\]

where:

- $\Omega$ is a program such that $|\Omega| = 0_H$ for all $\rho \in D(H)$
- $\bar{S} = S_0, S_1,$
Notation

(while \( M[\bar{q}] = 1 \) do \( S \))^{0} = \Omega,
(while \( M[\bar{q}] = 1 \) do \( S \))^{n+1} = \text{measure} \ M[\bar{q}] : \bar{S},

where:

- \( \Omega \) is a program such that \( \llbracket \Omega \rrbracket = 0_{\mathcal{H}_{\forall}} \) for all \( \rho \in \mathcal{D}(\mathcal{H}) \)
- \( \bar{S} = S_0, S_1 \)
  - \( S_0 = \text{skip} \),
  - \( S_1 = S; (\text{while } M[\bar{q}] = 1 \text{ do } S)^n \)

for all \( n \geq 0 \).
Recursion

\[
\llbracket \text{while} \rrbracket (\rho) = M_0 \rho M_0^\dagger + \llbracket \text{while} \rrbracket (\llbracket S \rrbracket (M_1 \rho M_1^\dagger))
\]

for all \(\rho \in D^- (\mathcal{H}_{all})\), where:

- \textbf{while} is the quantum loop “\textbf{while} \ M[\bar{q}] = 1 \ \textbf{do} \ S”.
Observation:

\[ tr(\|S\|(\rho)) \leq tr(\rho) \]

for any quantum program \( S \) and all \( \rho \in D^- (\mathcal{H}_{\text{all}}) \).

- \( tr(\rho) - tr(\|S\|(\rho)) \) is the probability that program \( S \) diverges from input state \( \rho \).
Definition

E. D’Hondt and P. Panangaden, Quantum weakest preconditions, *Mathematical Structures in Computer Science*, 16(2006)

- For any $X \subseteq Var$, a quantum predicate on $\mathcal{H}_X$ is a Hermitian operator $P$:

$$0_{\mathcal{H}_X} \sqsubseteq P \sqsubseteq I_{\mathcal{H}_X}.$$
Definition

E. D’Hondt and P. Panangaden, Quantum weakest preconditions, Mathematical Structures in Computer Science, 16(2006)

- For any $X \subseteq \text{Var}$, a quantum predicate on $\mathcal{H}_X$ is a Hermitian operator $P$:
  \[ 0_{\mathcal{H}_X} \subseteq P \subseteq I_{\mathcal{H}_X}. \]
- $\mathcal{P}(\mathcal{H}_X)$ denotes the set of quantum predicates on $\mathcal{H}_X$. 

$\rho \in D^{-}(\mathcal{H}_X)$, $\text{tr}(P\rho)$ stands for the probability that predicate $P$ is satisfied in state $\rho$. 
Definition

E. D’Hondt and P. Panangaden, Quantum weakest preconditions, *Mathematical Structures in Computer Science*, 16(2006)

- For any $X \subseteq \text{Var}$, a quantum predicate on $\mathcal{H}_X$ is a Hermitian operator $P$:
  \[ 0_{\mathcal{H}_X} \subseteq P \subseteq I_{\mathcal{H}_X}. \]

- $\mathcal{P}(\mathcal{H}_X)$ denotes the set of quantum predicates on $\mathcal{H}_X$.

- For any $\rho \in \mathcal{D}^-(\mathcal{H}_X)$, $tr(P\rho)$ stands for the probability that predicate $P$ is satisfied in state $\rho$. 
Definition

A correctness formula (Hoare triple) is a statement of the form:

\[ \{P\} S \{Q\} \]

where:

- S is a quantum program
Definition

A correctness formula (Hoare triple) is a statement of the form:

$$\{P\} S \{Q\}$$

where:

- $S$ is a quantum program
- $P$ and $Q$ are quantum predicates on $\mathcal{H}_{all}$. 
Definition

A correctness formula (Hoare triple) is a statement of the form:

\[
\{ P \} S \{ Q \}
\]

where:
- \( S \) is a quantum program
- \( P \) and \( Q \) are quantum predicates on \( \mathcal{H}_{all} \).
- Operator \( P \) is called the \textit{precondition} and \( Q \) the \textit{postcondition}.
Definition

1. The correctness formula \( \{P\} S \{Q\} \) is true in the sense of total correctness, written

\[ \models_{\text{tot}} \{P\} S \{Q\}, \]

if

\[ tr(P\rho) \leq tr(Q[S]_{\lambda}(\rho)) \]

for all \( \rho \in D^-(H_{all}) \).
Definition

1. The correctness formula $\{P\}S\{Q\}$ is true in the sense of *total correctness*, written

$$\models_{\text{tot}} \{P\}S\{Q\},$$

if

$$\text{tr}(P\rho) \leq \text{tr}(Q[S](\rho))$$

for all $\rho \in \mathcal{D}^-(\mathcal{H}_{all})$.

2. The correctness formula $\{P\}S\{Q\}$ is true in the sense of *partial correctness*, written

$$\models_{\text{par}} \{P\}S\{Q\},$$

if

$$\text{tr}(P\rho) \leq \text{tr}(Q[S](\rho)) + [\text{tr}(\rho) - \text{tr}([S](\rho))]$$

for all $\rho \in \mathcal{D}^-(\mathcal{H}_{all})$. 
Outline

Introduction

Syntax of Quantum Programs

Operational Semantics

Denotational Semantics

Correctness Formulas

Proof System for Quantum Programs

Conclusion
Proof System PD for Partial Correctness

(Axiom Skip) \( \{ P \} \text{Skip} \{ P \} \)

(Axiom Initialization)

\[
\text{type}(q) = \text{Boolean} :\\
\{ |0\rangle_q \langle 0| P |0\rangle_q \langle 0| + |1\rangle_q \langle 0| P |0\rangle_q \langle 1| \} q := 0\{ P \}
\]

\[
\text{type}(q) = \text{integer} :\\
\{ \sum_{n=-\infty}^{\infty} |n\rangle_q \langle 0| P |0\rangle_q \langle n| \} q := 0\{ P \}
\]

(Axiom Unitary Transformation) \( \{ U^\dagger P U \} \bar{q} := U \bar{q} \{ P \} \)
Proof System PD for Partial Correctness, Continued

(Rule Sequential Composition) \[ \frac{\{P\} S_1 \{Q\} \quad \{Q\} S_2 \{R\}}{\{P\} S_1 ; S_2 \{R\}} \]

(Rule Measurement) \[ \frac{\{P_m\} S_m \{Q\} \text{ for all } m}{\{\sum_m M^\dagger_m P_m M_m\} \text{measure } M[\bar{q}] : S\{Q\}} \]

(Rule Loop Partial) \[ \frac{\{Q\} S\{M^\dagger_0 P M_0 + M^\dagger_1 Q M_1\}}{\{M^\dagger_0 P M_0 + M^\dagger_1 Q M_1\} \text{while } M[\bar{q}] = 1 \text{ do } S\{P\}} \]

(Rule Order) \[ \frac{\quad P \sqsubseteq P' \quad \{P'\} S\{Q'\} \quad Q' \sqsubseteq Q}{\quad \{P\} S\{Q\}} \]
Soundness Theorem for PD

Proof system PD is sound for partial correctness of quantum programs.

For any quantum program $S$ and quantum predicates $P, Q \in \mathcal{P}(\mathcal{H}_{all})$, we have:

$$\vdash_{PD} \{P\} S \{Q\} \text{ implies } \models_{\text{par}} \{P\} S \{Q\}.$$
Completeness Theorem for PD

Proof system PD is complete for partial correctness of quantum programs.

- For any quantum program $S$ and quantum predicates $P, Q \in \mathcal{P}(\mathcal{H}_{\text{all}})$, we have:

  \[ \models_{\text{par}} \{P\} S\{Q\} \text{ implies } \vdash_{PD} \{P\} S\{Q\}. \]
Proof System $TD$ for Total Correctness

Let $P \in \mathcal{P}(\mathcal{H}_{\text{all}})$ and $\epsilon > 0$. A function

$$ t : \mathcal{D}^- (\mathcal{H}_{\text{all}}) \rightarrow \mathbb{N} $$

is called a $(P, \epsilon)$–bound function of quantum loop:

$$ \text{while } M[\bar{q}] = 1 \text{ do } S $$

if:

1. $t(\|[S](M_1\rho M_1^\dagger)) \leq t(\rho)$;

for all $\rho \in \mathcal{D}^- (\mathcal{H}_{\text{all}})$. 
Proof System $TD$ for Total Correctness

Let $P \in \mathcal{P}(\mathcal{H}_{all})$ and $\epsilon > 0$. A function

$$t : \mathcal{D}^{-}(\mathcal{H}_{all}) \rightarrow \mathbb{N}$$

is called a $(P, \epsilon)$–bound function of quantum loop:

$$\textbf{while } M[q] = 1 \textbf{ do } S$$

if:

1. $t(\mathcal{S}(M_1 \rho M_1^\dagger)) \leq t(\rho)$;
2. $tr(P \rho) \geq \epsilon$ implies $t(\mathcal{S}(M_1 \rho M_1^\dagger)) < t(\rho)$

for all $\rho \in \mathcal{D}^{-}(\mathcal{H}_{all})$. 
Proof System $TD$ for Total Correctness

Proof System $TD = (\text{Proof System } PD - \text{Rule Loop Partial}) + \text{Rule Loop Total}$
Proof System $TD$ for Total Correctness

Proof System $TD = \text{(Proof System } PD - \text{ Rule Loop Partial)} + \text{ Rule Loop Total}$

Rule: Total Correctness for Loop

(Rule Loop Total)

(1) $\{Q\} S\{M_0^\dagger PM_0 + M_1^\dagger QM_1\}$

(2) for any $\epsilon > 0$, $t_\epsilon$ is a $(M_1^\dagger QM_1, \epsilon) -$ bound function of loop $\text{while } M[\bar{q}] = 1 \text{ do } S$

$\{M_0^\dagger PM_0 + M_1^\dagger QM_1\} \text{while } M[\bar{q}] = 1 \text{ do } S\{P\}$
Soundness Theorem for TD

Proof system TD is sound for total correctness of quantum programs.

- For any quantum program $S$ and quantum predicates $P, Q \in \mathcal{P}(\mathcal{H}_{all})$, we have:

$$\vdash_{TD} \{P\} S \{Q\} \text{ implies } \models_{\text{tot}} \{P\} S \{Q\}.$$
Completeness Theorem

The proof system $TD$ is complete for total correctness of quantum programs.

- For any quantum program $S$ and quantum predicates $P, Q \in \mathcal{P}(\mathcal{H}_{all})$, we have:

\[
\models \text{tot} \{P\}S\{Q\} \implies \vdash_{TD} \{P\}S\{Q\}.
\]
Proof Outline

- Claim: $\vdash_{PD} \{ wp.S.Q \}S\{ Q \}$ for any quantum program $S$ and quantum predicate $P \in \mathcal{P}(\mathcal{H}_{all})$.

Induction on the structure of $S$.

$$wp.\text{while}.Q = M_0^\dagger QM_0 + M_1^\dagger(wp.S.(wp.\text{while}.Q))M_1.$$ 

Our aim is to derive:

$$\{ M_0^\dagger QM_0 + M_1^\dagger(wp.S.(wp.\text{while}.Q))M_1 \} \text{while}\{ Q \}.$$
Proof Outline

Claim: $\vdash_{PD} \{\text{wp}.S.Q\}S\{Q\}$ for any quantum program $S$ and quantum predicate $P \in \mathcal{P}(\mathcal{H}_\text{all})$.

Induction on the structure of $S$.

Example case: $S = \textbf{while } M[\bar{q}] = 1 \textbf{ do } S'$.

$$\text{wp.while}.Q = M_0^+QM_0 + M_1^+(\text{wp}.S.(\text{wp.while}.Q))M_1.$$ 

Our aim is to derive:

$$\{M_0^+QM_0 + M_1^+(\text{wp}.S.(\text{wp.while}.Q))M_1\}\textbf{while}\{Q\}.$$
Proof Outline, Continued

- Induction hypothesis on $S'$:

$$\{wp.S'.(wp.\text{while}.Q)\}S\{wp.\text{while}.Q\}.$$
Proof Outline, Continued

- Induction hypothesis on $S'$:
  \[
  \{wp.S'.(wp.\textbf{while}.Q)} \vdash \{wp.\textbf{while}.Q\}.
  \]

- Rule Loop Total: It suffices to show that for any $\epsilon > 0$, there exists a $(\hat{M}_1^\dagger(wp.S'.(wp.S.Q))M_1, \epsilon)$—bound function of quantum loop \textbf{while}. 
Proof Outline, Continued

- Induction hypothesis on $S'$:

$$\{wp.S'.(wp.\texttt{while}.Q)\}S\{wp.\texttt{while}.Q\}.$$ 

- Rule Loop Total: It suffices to show that for any $\epsilon > 0$, there exists a $(M_1^+(wp.S'.(wp.S.Q))M_1, \epsilon)$—bound function of quantum loop \texttt{while}.

- Bound Function Lemma: We only need to prove:

$$\lim_{n \to \infty} tr(M_1^+(wp.S'.(wp.\texttt{while}.Q))M_1([S'] \circ \mathcal{E}_1)^n(\rho)) = 0.$$
Proof Outline, Continued

We observe:

\[
tr(M_1^\dagger (wp.S'.(wp.\textbf{while}.Q))).M_1([S'] \circ E_1)^n(\rho))
\]

\[
= tr(wp.S'.(wp.\textbf{while}.Q))M_1([S'] \circ E_1)^n(\rho)M_1^\dagger
\]

\[
= tr(wp.\textbf{while}.Q[S'](M_1([S'] \circ E_1)^n(\rho)M_1^\dagger))
\]

\[
= tr(wp.\textbf{while}.Q([S'] \circ E_1)^{n+1}(\rho))
\]

\[
= tr(Q[\textbf{while}]([S'] \circ E_1)^{n+1}(\rho))
\]

\[
= \sum_{k=n+1}^{\infty} tr(Q[E_0 \circ ([S'] \circ E_1)^k](\rho))
\]
Proof Outline, Continued

We consider the infinite series of nonnegative real numbers:

\[ \sum_{n=0}^{\infty} tr(Q[\mathcal{E}_0 \circ ([\mathcal{S}'] \circ \mathcal{E}_1)^k](\rho)) = tr(Q \sum_{n=0}^{\infty} [\mathcal{E}_0 \circ ([\mathcal{S}'] \circ \mathcal{E}_1)^k](\rho)). \]

Since \( Q \sqsubseteq I_{\mathcal{H}_{all}} \), it follows that

\[ tr(Q \sum_{n=0}^{\infty} [\mathcal{E}_0 \circ ([\mathcal{S}'] \circ \mathcal{E}_1)^k](\rho)) = tr(Q[\texttt{while}](\rho)) \]

\[ \leq tr([\texttt{while}](\rho)) \leq tr(\rho) \leq 1. \]
Outline

Introduction

Syntax of Quantum Programs

Operational Semantics

Denotational Semantics

Correctness Formulas

Proof System for Quantum Programs

Conclusion
Conclusion

Hoare logic for deterministic quantum programs!

- Classical control flow $\Rightarrow$ quantum control flow?
Thank You!