Floyd-Hoare Logic for Quantum Programs

Mingsheng Ying

University of Technology, Sydney
and
Tsinghua University

Outline

Outline

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* 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* 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* 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

[1] R. W. Floyd, Assigning meanings to programs, *Proceedings of the American Mathematical Society Symposia on Applied Mathematics*, Vol. 19, 1967.
[2] C. A. R. Hoare, An axiomatic basis for computer programming, *Communications of the ACM* 1969.
[3] S. A. Cook, Soundness and Completeness of an Axiom System for Program Verification, *SIAM Journal on Computing* 1978.

Floyd-Hoare Logic for Quantum Programs

[1] O. Brunet and P. Jorrand, Dynamic quantum logic for quantum programs, *Int. J. of Quantum Information* 2004
[2] A. Baltag and S. Smets, LQP: the dynamic logic of quantum information, *Mathematical Structures in Computer Science* 2006
[3] Y. Kakutani, A logic for formal verification of quantum programs, *Advances in Computer Science - ASIAN'*2009

This talk is based on:

[4] M. S. Ying, Floyd-Hoare logic for quantum programs, *ACM Transactions on Programming Languages and Systems (TOPLAS)* 2011

Outline

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: **Boolean, integer**

Syntax, Continued

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

$$\mathcal{H}_{\mathbf{Boolean}} = \mathcal{H}_2,$$

$$\mathcal{H}_{\mathbf{integer}} = \mathcal{H}_\infty.$$

Space l_2 of square summable sequences

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

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 $\overline{q} = q_1, ..., q_n$:

$$\mathcal{H}_{\overline{q}} = \bigotimes_{i=1}^{n} \mathcal{H}_{q_i}.$$

Syntax, Continued

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

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

- q is a quantum variable and $\overline{q}$ a quantum register

Syntax, Continued

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

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

- 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:

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

- 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**:

Syntax, Continued

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

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

- 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 $\overline{q}$

Syntax, Continued

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

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

- 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

Syntax, Continued

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

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

- 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

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:

$$\langle S, \rho \rangle \to \langle S', \rho' \rangle$$

Operational Semantics

$$(Skip) \qquad \frac{}{\langle \textbf{skip}, \rho\rangle \rightarrow \langle E, \rho\rangle}$$

$$(Initialization) \qquad \frac{}{\langle q := 0, \rho\rangle \rightarrow \langle E, \rho_0^q\rangle}$$

- $type(q) = $ **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

$$(Skip) \qquad \frac{}{\langle \mathbf{skip}, \rho \rangle \rightarrow \langle E, \rho \rangle}$$

$$(Initialization) \qquad \frac{}{\langle q := 0, \rho \rangle \rightarrow \langle E, \rho_0^q \rangle}$$

- $type(q) = $ **Boolean**:

$$\rho_0^q = |0\rangle_q\langle 0|\rho|0\rangle_q\langle 0| + |0\rangle_q\langle 1|\rho|1\rangle_q\langle 0|$$

- $type(q) = $ **integer**:

$$\rho_0^q = \sum_{n=-\infty}^{\infty} |0\rangle_q\langle n|\rho|n\rangle_q\langle 0|$$

Operational Semantics, Continued

(*Unitary Transformation*) $$\frac{}{\langle \overline{q} := U\overline{q}, \rho\rangle \rightarrow \langle E, U\rho U^{\dagger}\rangle}$$

(*Sequential Composition*) $$\frac{\langle S_1, \rho\rangle \rightarrow \langle S_1', \rho'\rangle}{\langle S_1; S_2, \rho\rangle \rightarrow \langle S_1'; S_2, \rho'\rangle}$$

$$\text{Convention}: \ E; S_2 = S_2.$$

(*Measurement*) $$\frac{}{\langle \textbf{measure}\ M[\overline{q}] : \overline{S}, \rho\rangle \rightarrow \langle S_m, M_m \rho M_m^{\dagger}\rangle}$$

for each outcome m

Operational Semantics, Continued

(*Loop* 0)
$$\frac{}{\langle \textbf{while } M[\overline{q}] = 1 \textbf{ do } S, \rho\rangle \rightarrow \langle E, M_0 \rho M_0^\dagger\rangle}$$

(*Loop* 1)
$$\frac{}{\langle \textbf{while } M[\overline{q}] = 1 \textbf{ do } S, \rho\rangle \rightarrow \langle S; \textbf{while } M[\overline{q}] = 1 \textbf{ do } S, M_1 \rho M_1^\dagger\rangle}$$

Outline

Definition

Semantic function of quantum program S:

$$[\![S]\!] : \mathcal{D}^-(\mathcal{H}_{\text{all}}) \to \mathcal{D}^-(\mathcal{H}_{\text{all}})$$

is defined by

$$[\![S]\!](\rho) = \sum \{|\rho' : \langle S, \rho\rangle \to^* \langle E, \rho'\rangle|\}$$

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

Representation of Semantic Function

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

Representation of Semantic Function

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

Representation of Semantic Function

1. $[\![\mathbf{skip}]\!](\rho) = \rho.$
2. - $type(q) = \mathbf{Boolean}$:

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

   - $type(q) = \mathbf{integer}$:

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

Representation of Semantic Function

1. $[\![\mathbf{skip}]\!](\rho) = \rho$.
2. - $type(q) = \mathbf{Boolean}$:

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

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

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

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

Representation of Semantic Function

1. $[\![\mathbf{skip}]\!](\rho) = \rho$.
2. - $type(q) = \mathbf{Boolean}$:

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

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

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

3. $[\![\overline{q} := U\overline{q}]\!](\rho) = U\rho U^\dagger$.
4. $[\![S_1; S_2]\!](\rho) = [\![S_2]\!]([\![S_1]\!](\rho))$.

Representation of Semantic Function

1. $[\![\textbf{skip}]\!](\rho) = \rho.$
2. - $type(q) = \textbf{Boolean}$:

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

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

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

3. $[\![\overline{q} := U\overline{q}]\!](\rho) = U\rho U^\dagger.$
4. $[\![S_1; S_2]\!](\rho) = [\![S_2]\!]([\![S_1]\!](\rho)).$
5. $[\![\textbf{measure}\ M[\overline{q}] : \overline{S}]\!](\rho) = \sum_m [\![S_m]\!](M_m \rho M_m^\dagger).$

Representation of Semantic Function

1. $[\![\mathbf{skip}]\!](\rho) = \rho$.
2. - $type(q) = \mathbf{Boolean}$:
   $$[\![q := 0]\!](\rho) = |0\rangle_q\langle 0|\rho|0\rangle_q\langle 0| + |0\rangle_q\langle 1|\rho|1\rangle_q\langle 0|.$$
   - $type(q) = \mathbf{integer}$:
   $$[\![q := 0]\!](\rho) = \sum_{n=-\infty}^{\infty} |0\rangle_q\langle n|\rho|n\rangle_q\langle 0|.$$
3. $[\![\overline{q} := U\overline{q}]\!](\rho) = U\rho U^\dagger$.
4. $[\![S_1; S_2]\!](\rho) = [\![S_2]\!]([\![S_1]\!](\rho))$.
5. $[\![\mathbf{measure}\ M[\overline{q}] : \overline{S}]\!](\rho) = \sum_m [\![S_m]\!](M_m \rho M_m^\dagger)$.
6. $[\![\mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ S]\!](\rho) = \bigvee_{n=0}^{\infty} [\![(\mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ S)^n]\!](\rho)$.

Notation

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

where:

- Ω is a program such that $[\![\Omega]\!] = 0_{\mathcal{H}_{\text{all}}}$ for all $\rho \in \mathcal{D}(\mathcal{H})$

Notation

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

where:

- Ω is a program such that $[\![\Omega]\!] = 0_{\mathcal{H}_{\text{all}}}$ for all $\rho \in \mathcal{D}(\mathcal{H})$
- $\overline{S} = S_0, S_1,$

Notation

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

where:

- Ω is a program such that $[\![\Omega]\!] = 0_{\mathcal{H}_{\text{all}}}$ for all $\rho \in \mathcal{D}(\mathcal{H})$
- $\overline{S} = S_0, S_1,$
  -

$$S_0 = \textbf{skip},$$
$$S_1 = S; (\textbf{while } M[\overline{q}] = 1 \textbf{ do } S)^n$$

for all $n \geq 0$.

Recursion

$$[\![\mathbf{while}]\!](\rho) = M_0 \rho M_0^\dagger + [\![\mathbf{while}]\!]([\![S]\!](M_1 \rho M_1^\dagger))$$

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

- **while** is the quantum loop "**while** $M[\overline{q}] = 1$ **do** S".

Observation:

$$tr([\![S]\!](\rho)) \leq tr(\rho)$$

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

- $tr(\rho) - tr([\![S]\!](\rho))$ is the probability that program S diverges from input state ρ.

Outline

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

Definition

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

- For any $X \subseteq \mathit{Var}$, a quantum predicate on $\mathcal{H}_X$ is a Hermitian operator P:
$$0_{\mathcal{H}_X} \sqsubseteq P \sqsubseteq I_{\mathcal{H}_X}.$$
- $\mathcal{P}(\mathcal{H}_X)$ 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 ρ.

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 *precondition* and Q the *postcondition*.

Definition

1. The correctness formula $\{P\}S\{Q\}$ is true in the sense of *total correctness*, written
$$\models_{\rm tot} \{P\}S\{Q\},$$
if
$$tr(P\rho) \leq tr(Q[\![S]\!](\rho))$$
for all $\rho \in \mathcal{D}^-(\mathcal{H}_{all})$.

Definition

1. The correctness formula $\{P\}S\{Q\}$ is true in the sense of *total correctness*, written
$$\models_{\text{tot}} \{P\}S\{Q\},$$
if
$$tr(P\rho) \leq tr(Q[\![S]\!](\rho))$$
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
$$tr(P\rho) \leq tr(Q[\![S]\!](\rho)) + [tr(\rho) - tr([\![S]\!](\rho))]$$
for all $\rho \in \mathcal{D}^-(\mathcal{H}_{all})$.

Outline

Proof System *PD* for Partial Correctness

(*Axiom Skip*) $\{P\}\mathbf{Skip}\{P\}$

(*Axiom Initialization*)

$type(q) = \mathbf{Boolean}$:

$$\{|0\rangle_q\langle 0|P|0\rangle_q\langle 0| + |1\rangle_q\langle 0|P|0\rangle_q\langle 1|\}\; q := 0\; \{P\}$$

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

$$\left\{\sum_{n=-\infty}^{\infty} |n\rangle_q\langle 0|P|0\rangle_q\langle n|\right\} q := 0\; \{P\}$$

(*Axiom Unitary Transformation*) $\{U^\dagger PU\}\overline{q} := U\overline{q}\{P\}$

Proof System *PD* for Partial Correctness, Continued

(*Rule Sequential Composition*) $$\frac{\{P\}S_1\{Q\} \qquad \{Q\}S_2\{R\}}{\{P\}S_1;S_2\{R\}}$$

(*Rule Measurement*) $$\frac{\{P_m\}S_m\{Q\} \text{ for all } m}{\{\sum_m M_m^\dagger P_m M_m\}\textbf{measure } M[\overline{q}] : \overline{S}\{Q\}}$$

(*Rule Loop Partial*) $$\frac{\{Q\}S\{M_0^\dagger P M_0 + M_1^\dagger Q M_1\}}{\{M_0^\dagger P M_0 + M_1^\dagger Q M_1\}\textbf{while } M[\overline{q}] = 1 \textbf{ do } S\{P\}}$$

(*Rule Order*) $$\frac{P \sqsubseteq P' \quad \{P'\}S\{Q'\} \quad Q' \sqsubseteq Q}{\{P\}S\{Q\}}$$

Soundness Theorem for *PD*

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

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

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

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

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

if:

1. $t(\llbracket S \rrbracket(M_1 \rho M_1^{\dagger})) \leq t(\rho)$;

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

Proof System *TD* for Total Correctness

Let $P \in \mathcal{P}(\mathcal{H}_{\text{all}})$ 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:

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

if:

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

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

Proof System *TD* for Total Correctness

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

Proof System TD for Total Correctness

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

Rule: Total Correctness for Loop

$$(Rule\ Loop\ Total)\quad \frac{\begin{array}{l}(1)\ \{Q\}S\{M_0^\dagger PM_0 + M_1^\dagger QM_1\}\\ (2)\ \text{for any } \epsilon > 0,\ t_\epsilon \text{ is a } (M_1^\dagger QM_1, \epsilon) - \text{bound}\\ \qquad\qquad \text{function of loop } \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ S\end{array}}{\{M_0^\dagger PM_0 + M_1^\dagger QM_1\}\mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ S\{P\}}$$

Soundness Theorem for *TD*

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

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

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

Proof Outline

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

Induction on the structure of S.

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

Our aim is to derive:

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

Proof Outline

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

Induction on the structure of S.

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

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

Our aim is to derive:

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

Proof Outline, Continued

- Induction hypothesis on S':

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

Proof Outline, Continued

- Induction hypothesis on S':

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

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

Proof Outline, Continued

- Induction hypothesis on S':

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

- Rule Loop Total: It suffices to show that for any $\epsilon > 0$, there exists a $(M_1^\dagger(wp.S'.\,(wp.S.Q))M_1, \epsilon)-$bound function of quantum loop **while**.
- Bound Function Lemma: We only need to prove:

$$\lim_{n\to\infty} tr(M_1^\dagger(wp.S'.(wp.\textbf{while}.Q))M_1([\![S']\!] \circ \mathcal{E}_1)^n(\rho)) = 0.$$

Proof Outline, Continued

We observe:

$$
\begin{aligned}
tr(M_1^\dagger(wp.S'.(wp.\textbf{while}.Q))M_1(\llbracket S'\rrbracket \circ \mathcal{E}_1)^n(\rho))
&= tr(wp.S'.(wp.\textbf{while}.Q)M_1(\llbracket S'\rrbracket \circ \mathcal{E}_1)^n(\rho)M_1^\dagger) \\
&= tr(wp.\textbf{while}.Q\llbracket S'\rrbracket(M_1(\llbracket S'\rrbracket \circ \mathcal{E}_1)^n(\rho)M_1^\dagger)) \\
&= tr(wp.\textbf{while}.Q(\llbracket S'\rrbracket \circ \mathcal{E}_1)^{n+1}(\rho)) \\
&= tr(Q\llbracket\textbf{while}\rrbracket(\llbracket S'\rrbracket \circ \mathcal{E}_1)^{n+1}(\rho)) \\
&= \sum_{k=n+1}^{\infty} tr(Q[\mathcal{E}_0 \circ (\llbracket S'\rrbracket \circ \mathcal{E}_1)^k](\rho)).
\end{aligned}
$$

Proof Outline, Continued

We consider the infinite series of nonnegative real numbers:

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

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

$$\begin{aligned} tr(Q \sum_{n=0}^{\infty} [\mathcal{E}_0 \circ ([\![S']\!] \circ \mathcal{E}_1)^k](\rho)) &= tr(Q[\![\mathbf{while}]\!](\rho)) \\ &\leq tr([\![\mathbf{while}]\!](\rho)) \leq tr(\rho) \leq 1. \end{aligned}$$

Outline

Conclusion

- Superposition-of-data $\Rightarrow$ Superposition-of-programs

Conclusion

- Superposition-of-data $\Rightarrow$ Superposition-of-programs
- Or, classical control flow $\Rightarrow$ quantum control flow

Conclusion

- Superposition-of-data ⇒ Superposition-of-programs
- Or, classical control flow ⇒ quantum control flow
- In particular, *quantum recursion defined by second quantization*.

Conclusion

- Superposition-of-data $\Rightarrow$ Superposition-of-programs
- Or, classical control flow $\Rightarrow$ quantum control flow
- In particular, *quantum recursion defined by second quantization*.

- M. S. Ying, N. K. Yu and Y. Feng, Alternation in quantum programming: from superposition of data to superposition of programs, *http://arxiv.org/abs/1402.5172*.

Conclusion

- Superposition-of-data ⇒ Superposition-of-programs
- Or, classical control flow ⇒ quantum control flow
- In particular, *quantum recursion defined by second quantization*.

- M. S. Ying, N. K. Yu and Y. Feng, Alternation in quantum programming: from superposition of data to superposition of programs, *http://arxiv.org/abs/1402.5172*.
- M. S. Ying, Quantum recursion and second quantization, *draft, 2014*.

Conclusion

- Superposition-of-data ⇒ Superposition-of-programs
- Or, classical control flow ⇒ quantum control flow
- In particular, *quantum recursion defined by second quantization*.

- M. S. Ying, N. K. Yu and Y. Feng, Alternation in quantum programming: from superposition of data to superposition of programs, *http://arxiv.org/abs/1402.5172*.
- M. S. Ying, Quantum recursion and second quantization, *draft, 2014*.

- **Problem**: How can we build a Floyd-Hoare logic for quantum program with quantum control flows, in particular with *quantum loops defined by second quantization*?

Thank You!