Foundations of Quantum Programming

Lecture 3: Syntax and Semantics of Quantum Programs

Mingsheng Ying

University of Technology Sydney, Australia

Outline

Outline

Classical **while**-Language

$$
\begin{aligned}
S ::= \ & \mathbf{skip} \mid u := t \mid S_1; S_2 \\
& \mid \mathbf{if}\ b\ \mathbf{then}\ S_1\ \mathbf{else}\ S_2\ \mathbf{fi} \\
& \mid \mathbf{while}\ b\ \mathbf{do}\ S\ \mathbf{od}.
\end{aligned}
$$

- The conditional statement can be generalised to the case statement:

$$
\begin{aligned}
& \mathbf{if}\ G_1 \rightarrow S_1 \\
& \square\ G_2 \rightarrow S_2 \\
& \qquad \ldots\ldots \\
& \square\ G_n \rightarrow S_n \\
& \mathbf{fi}
\end{aligned}
$$

or more compactly

$$
\mathbf{if}\ (\square i \cdot G_i \rightarrow S_i)\ \mathbf{fi}
$$

Quantum **while**-Language

- Fix the alphabet of quantum **while**-language: A countably infinite set *qVar* of quantum variables. Symbols $q, q', q_0, q_1, q_2, \ldots$ denote quantum variables.

Quantum **while**-Language

- Fix the alphabet of quantum **while**-language: A countably infinite set $qVar$ of quantum variables. Symbols $q, q', q_0, q_1, q_2, ...$ denote quantum variables.
- Each quantum variable $q \in qVar$ has a type $\mathcal{H}_q$ (a Hilbert space).

Quantum **while**-Language

- Fix the alphabet of quantum **while**-language: A countably infinite set $qVar$ of quantum variables. Symbols $q, q', q_0, q_1, q_2, ...$ denote quantum variables.
- Each quantum variable $q \in qVar$ has a type $\mathcal{H}_q$ (a Hilbert space).
- For simplicity, we only consider two basic types:

$$\textbf{Boolean} = \mathcal{H}_2, \qquad \textbf{integer} = \mathcal{H}_\infty.$$

Quantum **while**-Language

- Fix the alphabet of quantum **while**-language: A countably infinite set $qVar$ of quantum variables. Symbols $q, q', q_0, q_1, q_2, ...$ denote quantum variables.
- Each quantum variable $q \in qVar$ has a type $\mathcal{H}_q$ (a Hilbert space).
- For simplicity, we only consider two basic types:

$$\textbf{Boolean} = \mathcal{H}_2, \qquad \textbf{integer} = \mathcal{H}_\infty.$$

- A quantum register is a finite sequence $\overline{q} = q_1, ..., q_n$ of distinct quantum variables. Its state Hilbert space:

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

Quantum Programs

$$
\begin{aligned}
S ::= \ & \textbf{skip} \mid q := |0\rangle \mid \overline{q} := U[\overline{q}] \mid S_1; S_2 \\
& \mid \textbf{if}\ (\square m \cdot M[\overline{q}] = m \rightarrow S_m)\ \textbf{fi} \\
& \mid \textbf{while}\ M[\overline{q}] = 1\ \textbf{do}\ S\ \textbf{od}.
\end{aligned}
$$

Classical Control Flow

- The control flow of a program is the *order of its execution*.

Classical Control Flow

- The control flow of a program is the *order of its execution*.
- In quantum **while**-language, there are only two statements — *case statement*, *loop* — whose execution is determined by a choice as to which of two or more paths should be followed.

Classical Control Flow

- The control flow of a program is the *order of its execution*.
- In quantum **while**-language, there are only two statements — *case statement*, *loop* — whose execution is determined by a choice as to which of two or more paths should be followed.
- The case statement selects a command to execute according to the outcome of measurement M: if the outcome is m_i, then the corresponding command S_{m_i} will be executed. Since the outcome of a quantum measurement is classical information, the control flow is classical.

Classical Control Flow

- The control flow of a program is the *order of its execution*.
- In quantum **while**-language, there are only two statements — *case statement*, *loop* — whose execution is determined by a choice as to which of two or more paths should be followed.
- The case statement selects a command to execute according to the outcome of measurement M: if the outcome is m_i, then the corresponding command S_{m_i} will be executed. Since the outcome of a quantum measurement is classical information, the control flow is classical.
- The control flow in the loop is classical too.

Classical Control Flow

- The control flow of a program is the *order of its execution*.
- In quantum **while**-language, there are only two statements — *case statement*, *loop* — whose execution is determined by a choice as to which of two or more paths should be followed.
- The case statement selects a command to execute according to the outcome of measurement M: if the outcome is m_i, then the corresponding command S_{m_i} will be executed. Since the outcome of a quantum measurement is classical information, the control flow is classical.
- The control flow in the loop is classical too.
- Programs with *quantum control flow*?

Outline

Notations

- A positive operator ρ is called a *partial density operator* if $tr(\rho) \leq 1$.

Notations

- A positive operator ρ is called a *partial density operator* if $tr(\rho) \leq 1$.
- Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.

Notations

- A positive operator ρ is called a *partial density operator* if $tr(\rho) \leq 1$.
- Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.
- Write $\mathcal{H}_{all}$ for the tensor product of the state Hilbert spaces of all quantum variables:

$$\mathcal{H}_{all} = \bigotimes_{q \in qVar} \mathcal{H}_q.$$

Notations

- A positive operator ρ is called a *partial density operator* if $tr(\rho) \leq 1$.
- Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.
- Write $\mathcal{H}_{all}$ for the tensor product of the state Hilbert spaces of all quantum variables:
$$\mathcal{H}_{all} = \bigotimes_{q \in qVar} \mathcal{H}_q.$$
- Let $\overline{q} = q_1, ..., q_n$ be a quantum register. An operator A in the state Hilbert space $\mathcal{H}_{\overline{q}}$ of $\overline{q}$ has a cylindrical extension $A \otimes I$ in $\mathcal{H}_{all}$.

Notations

- A positive operator ρ is called a *partial density operator* if $tr(\rho) \leq 1$.
- Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.
- Write $\mathcal{H}_{all}$ for the tensor product of the state Hilbert spaces of all quantum variables:
$$\mathcal{H}_{all} = \bigotimes_{q \in qVar} \mathcal{H}_q.$$
- Let $\overline{q} = q_1, ..., q_n$ be a quantum register. An operator A in the state Hilbert space $\mathcal{H}_{\overline{q}}$ of $\overline{q}$ has a cylindrical extension $A \otimes I$ in $\mathcal{H}_{all}$.
- We will use E to denote the empty program; i.e. termination.

Notations

- A positive operator ρ is called a *partial density operator* if $tr(\rho) \leq 1$.
- Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.
- Write $\mathcal{H}_{all}$ for the tensor product of the state Hilbert spaces of all quantum variables:
$$\mathcal{H}_{all} = \bigotimes_{q \in qVar} \mathcal{H}_q.$$
- Let $\overline{q} = q_1, ..., q_n$ be a quantum register. An operator A in the state Hilbert space $\mathcal{H}_{\overline{q}}$ of $\overline{q}$ has a cylindrical extension $A \otimes I$ in $\mathcal{H}_{all}$.
- We will use E to denote the empty program; i.e. termination.
- A configuration is a pair $\langle S, \rho \rangle$, where:

Notations

- A positive operator ρ is called a *partial density operator* if $tr(\rho) \leq 1$.
- Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.
- Write $\mathcal{H}_{all}$ for the tensor product of the state Hilbert spaces of all quantum variables:
$$\mathcal{H}_{all} = \bigotimes_{q \in qVar} \mathcal{H}_q.$$
- Let $\overline{q} = q_1, ..., q_n$ be a quantum register. An operator A in the state Hilbert space $\mathcal{H}_{\overline{q}}$ of $\overline{q}$ has a cylindrical extension $A \otimes I$ in $\mathcal{H}_{all}$.
- We will use E to denote the empty program; i.e. termination.
- A configuration is a pair $\langle S, \rho \rangle$, where:
    1. S is a quantum program or the empty program E;

Notations

- A positive operator ρ is called a *partial density operator* if $tr(\rho) \leq 1$.
- Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.
- Write $\mathcal{H}_{all}$ for the tensor product of the state Hilbert spaces of all quantum variables:

$$\mathcal{H}_{all} = \bigotimes_{q \in qVar} \mathcal{H}_q.$$

- Let $\overline{q} = q_1, ..., q_n$ be a quantum register. An operator A in the state Hilbert space $\mathcal{H}_{\overline{q}}$ of $\overline{q}$ has a cylindrical extension $A \otimes I$ in $\mathcal{H}_{all}$.
- We will use E to denote the empty program; i.e. termination.
- A configuration is a pair $\langle S, \rho \rangle$, where:
  1. S is a quantum program or the empty program E;
  2. $\rho \in \mathcal{D}(\mathcal{H}_{all})$, denoting the (global) state of quantum variables.

Notations

- A positive operator ρ is called a *partial density operator* if $tr(\rho) \leq 1$.
- Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.
- Write $\mathcal{H}_{all}$ for the tensor product of the state Hilbert spaces of all quantum variables:

$$\mathcal{H}_{all} = \bigotimes_{q \in qVar} \mathcal{H}_q.$$

- Let $\overline{q} = q_1, ..., q_n$ be a quantum register. An operator A in the state Hilbert space $\mathcal{H}_{\overline{q}}$ of $\overline{q}$ has a cylindrical extension $A \otimes I$ in $\mathcal{H}_{all}$.
- We will use E to denote the empty program; i.e. termination.
- A configuration is a pair $\langle S, \rho \rangle$, where:
  1. S is a quantum program or the empty program E;
  2. $\rho \in \mathcal{D}(\mathcal{H}_{all})$, denoting the (global) state of quantum variables.
- A transition between quantum configurations:

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

Operational Semantics

The operational semantics of quantum programs is the transition relation $\to$ between quantum configurations defined by the transition rules:

(SK) $$\frac{}{\langle \mathbf{skip}, \rho\rangle \to \langle E, \rho\rangle}$$

(IN) $$\frac{}{\langle q := |0\rangle, \rho\rangle \to \langle E, \rho_0^q\rangle}$$

where

$$\rho_0^q = \begin{cases} |0\rangle_q\langle 0|\rho|0\rangle_q\langle 0| + |0\rangle_q\langle 1|\rho|1\rangle_q\langle 0| & \text{if } \mathit{type}(q) = \mathbf{Boolean}, \\ \sum_{n=-\infty}^{\infty} |0\rangle_q\langle n|\rho|n\rangle_q\langle 0| & \text{if } \mathit{type}(q) = \mathbf{integer}. \end{cases}$$

(UT) $$\frac{}{\langle \overline{q} := U[\overline{q}], \rho\rangle \to \langle E, U\rho U^\dagger\rangle}$$

Operational Semantics (Continued)

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

where $E; S_2 = S_2$.

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

for each possible outcome m of measurement $M = \{M_m\}$.

(L0) $$\frac{}{\langle \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ S\ \mathbf{od}, \rho\rangle \rightarrow \langle E, M_0 \rho M_0^\dagger\rangle}$$

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

Computation of a Program

Let S be a quantum program and $\rho \in \mathcal{D}(\mathcal{H}_{all})$.

1. A transition sequence of S starting in ρ is a finite or infinite sequence of configurations:

$$\langle S, \rho \rangle \rightarrow \langle S_1, \rho_1 \rangle \rightarrow ... \rightarrow \langle S_n, \rho_n \rangle \rightarrow \langle S_{n+1}, \rho_{n+1} \rangle \rightarrow ...$$

   such that $\rho_n \neq 0$ for all n (except the last n in the case of a finite sequence).

Computation of a Program

Let S be a quantum program and $\rho \in \mathcal{D}(\mathcal{H}_{all})$.

1. A transition sequence of S starting in ρ is a finite or infinite sequence of configurations:

$$\langle S, \rho \rangle \rightarrow \langle S_1, \rho_1 \rangle \rightarrow ... \rightarrow \langle S_n, \rho_n \rangle \rightarrow \langle S_{n+1}, \rho_{n+1} \rangle \rightarrow ...$$

   such that $\rho_n \neq 0$ for all n (except the last n in the case of a finite sequence).

2. If this sequence cannot be extended, then it is called a computation of S starting in ρ.

Computation of a Program

Let S be a quantum program and $\rho \in \mathcal{D}(\mathcal{H}_{all})$.

1. A transition sequence of S starting in ρ is a finite or infinite sequence of configurations:

$$\langle S, \rho \rangle \to \langle S_1, \rho_1 \rangle \to \ldots \to \langle S_n, \rho_n \rangle \to \langle S_{n+1}, \rho_{n+1} \rangle \to \ldots$$

   such that $\rho_n \neq 0$ for all n (except the last n in the case of a finite sequence).

2. If this sequence cannot be extended, then it is called a computation of S starting in ρ.
   - If a computation is finite and its last configuration is $\langle E, \rho' \rangle$, then we say that it terminates in ρ'.

Computation of a Program

Let S be a quantum program and $\rho \in \mathcal{D}(\mathcal{H}_{all})$.

1. A transition sequence of S starting in ρ is a finite or infinite sequence of configurations:

$$\langle S, \rho \rangle \rightarrow \langle S_1, \rho_1 \rangle \rightarrow \dots \rightarrow \langle S_n, \rho_n \rangle \rightarrow \langle S_{n+1}, \rho_{n+1} \rangle \rightarrow \dots$$

   such that $\rho_n \neq 0$ for all n (except the last n in the case of a finite sequence).

2. If this sequence cannot be extended, then it is called a computation of S starting in ρ.
   - If a computation is finite and its last configuration is $\langle E, \rho' \rangle$, then we say that it terminates in ρ'.
   - If it is infinite, then we say that it diverges.

Outline

Semantic Function

- If configuration $\langle S', \rho' \rangle$ can be reached from $\langle S, \rho \rangle$ in n steps: there are configurations $\langle S_1, \rho_1 \rangle, ..., \langle S_{n-1}, \rho_{n-1} \rangle$ such that

$$\langle S, \rho \rangle \rightarrow \langle S_1, \rho_1 \rangle \rightarrow ... \rightarrow \langle S_{n-1}, \rho_{n-1} \rangle \rightarrow \langle S', \rho' \rangle,$$

then we write:

$$\langle S, \rho \rangle \rightarrow^n \langle S', \rho' \rangle.$$

Semantic Function

- If configuration $\langle S', \rho' \rangle$ can be reached from $\langle S, \rho \rangle$ in n steps: there are configurations $\langle S_1, \rho_1 \rangle, ..., \langle S_{n-1}, \rho_{n-1} \rangle$ such that

$$\langle S, \rho \rangle \to \langle S_1, \rho_1 \rangle \to ... \to \langle S_{n-1}, \rho_{n-1} \rangle \to \langle S', \rho' \rangle,$$

  then we write:

$$\langle S, \rho \rangle \to^n \langle S', \rho' \rangle.$$

- Write $\to^*$ for the reflexive and transitive closures of $\to$:

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

  if and only if $\langle S, \rho \rangle \to^n \langle S', \rho' \rangle$ for some $n \geq 0$.

Semantic Function

- If configuration $\langle S', \rho' \rangle$ can be reached from $\langle S, \rho \rangle$ in n steps: there are configurations $\langle S_1, \rho_1 \rangle, \ldots, \langle S_{n-1}, \rho_{n-1} \rangle$ such that

$$\langle S, \rho \rangle \rightarrow \langle S_1, \rho_1 \rangle \rightarrow \ldots \rightarrow \langle S_{n-1}, \rho_{n-1} \rangle \rightarrow \langle S', \rho' \rangle,$$

  then we write:

$$\langle S, \rho \rangle \rightarrow^n \langle S', \rho' \rangle.$$

- Write $\rightarrow^*$ for the reflexive and transitive closures of $\rightarrow$:

$$\langle S, \rho \rangle \rightarrow^* \langle S', \rho' \rangle$$

  if and only if $\langle S, \rho \rangle \rightarrow^n \langle S', \rho' \rangle$ for some $n \geq 0$.

- Let S be a quantum program. Then its semantic function

$$[\![S]\!] : \mathcal{D}(\mathcal{H}_{all}) \rightarrow \mathcal{D}(\mathcal{H}_{all})$$

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

Linearity

Let $\rho_1, \rho_2 \in \mathcal{D}(\mathcal{H}_{all})$ and $\lambda_1, \lambda_2 \geq 0$. If $\lambda_1\rho_1 + \lambda_2\rho_2 \in \mathcal{D}(\mathcal{H}_{all})$, then for any quantum program S:

$$[\![S]\!](\lambda_1\rho_1 + \lambda_2\rho_2) = \lambda_1[\![S]\!](\rho_1) + \lambda_2[\![S]\!](\rho_2).$$

Structural Representation

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

Structural Representation

1. $[\![\mathbf{skip}]\!](\rho) = \rho$.
2.

Structural Representation

1. $[\![\mathbf{skip}]\!](\rho) = \rho.$
2.
   2.1 If $type(q) = \mathbf{Boolean}$, then

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

Structural Representation

1. $[\![\mathbf{skip}]\!](\rho) = \rho$.
2.
   2.1 If $type(q) = \mathbf{Boolean}$, then

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

   2.2 If $type(q) = \mathbf{integer}$, then

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

Structural Representation

1. $[\![\mathbf{skip}]\!](\rho) = \rho$.
2.
   2.1 If $type(q) = \mathbf{Boolean}$, then

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

   2.2 If $type(q) = \mathbf{integer}$, then

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

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

Structural Representation

1. $[\![\mathbf{skip}]\!](\rho) = \rho$.
2.
   2.1 If $type(q) = \mathbf{Boolean}$, then

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

   2.2 If $type(q) = \mathbf{integer}$, then

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

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

Structural Representation

1. $[\![\mathbf{skip}]\!](\rho) = \rho.$
2.
   2.1 If $type(q) = \mathbf{Boolean}$, then

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

   2.2 If $type(q) = \mathbf{integer}$, then

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

3. $[\![\overline{q} := U[\overline{q}]]\!](\rho) = U\rho U^\dagger.$
4. $[\![S_1; S_2]\!](\rho) = [\![S_2]\!]([\![S_1]\!](\rho)).$
5. $[\![\mathbf{if}\ (\square m \cdot M[\overline{q}] = m \rightarrow S_m)\ \mathbf{fi}]\!](\rho) = \sum_m [\![S_m]\!](M_m\rho M_m^\dagger).$

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:
  1. Reflexivity: $x \sqsubseteq x$ for all $x \in L$;

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:
  1. Reflexivity: $x \sqsubseteq x$ for all $x \in L$;
  2. Antisymmetry: $x \sqsubseteq y$ and $y \sqsubseteq x$ imply $x = y$ for all $x, y \in L$;

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:
  1. Reflexivity: $x \sqsubseteq x$ for all $x \in L$;
  2. Antisymmetry: $x \sqsubseteq y$ and $y \sqsubseteq x$ imply $x = y$ for all $x, y \in L$;
  3. Transitivity: $x \sqsubseteq y$ and $y \sqsubseteq z$ imply $x \sqsubseteq z$ for all $x, y, z \in L$.

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:
  1. Reflexivity: $x \sqsubseteq x$ for all $x \in L$;
  2. Antisymmetry: $x \sqsubseteq y$ and $y \sqsubseteq x$ imply $x = y$ for all $x, y \in L$;
  3. Transitivity: $x \sqsubseteq y$ and $y \sqsubseteq z$ imply $x \sqsubseteq z$ for all $x, y, z \in L$.
- Let $(L, \sqsubseteq)$ be a partial order.

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:
  1. Reflexivity: $x \sqsubseteq x$ for all $x \in L$;
  2. Antisymmetry: $x \sqsubseteq y$ and $y \sqsubseteq x$ imply $x = y$ for all $x, y \in L$;
  3. Transitivity: $x \sqsubseteq y$ and $y \sqsubseteq z$ imply $x \sqsubseteq z$ for all $x, y, z \in L$.
- Let $(L, \sqsubseteq)$ be a partial order.
  1. An element $x \in L$ is called the least element of L when $x \sqsubseteq y$ for all $y \in L$. The least element is denoted by 0.

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:
    1. Reflexivity: $x \sqsubseteq x$ for all $x \in L$;
    2. Antisymmetry: $x \sqsubseteq y$ and $y \sqsubseteq x$ imply $x = y$ for all $x, y \in L$;
    3. Transitivity: $x \sqsubseteq y$ and $y \sqsubseteq z$ imply $x \sqsubseteq z$ for all $x, y, z \in L$.
- Let $(L, \sqsubseteq)$ be a partial order.
    1. An element $x \in L$ is called the least element of L when $x \sqsubseteq y$ for all $y \in L$. The least element is denoted by 0.
    2. An element $x \in L$ is called an upper bound of a subset $X \subseteq L$ if $y \sqsubseteq x$ for all $x \in X$.

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:
    1. Reflexivity: $x \sqsubseteq x$ for all $x \in L$;
    2. Antisymmetry: $x \sqsubseteq y$ and $y \sqsubseteq x$ imply $x = y$ for all $x, y \in L$;
    3. Transitivity: $x \sqsubseteq y$ and $y \sqsubseteq z$ imply $x \sqsubseteq z$ for all $x, y, z \in L$.
- Let $(L, \sqsubseteq)$ be a partial order.
    1. An element $x \in L$ is called the least element of L when $x \sqsubseteq y$ for all $y \in L$. The least element is denoted by 0.
    2. An element $x \in L$ is called an upper bound of a subset $X \subseteq L$ if $y \sqsubseteq x$ for all $x \in X$.
    3. x is called the least upper bound of X, written $x = \bigsqcup X$, if

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:
  1. Reflexivity: $x \sqsubseteq x$ for all $x \in L$;
  2. Antisymmetry: $x \sqsubseteq y$ and $y \sqsubseteq x$ imply $x = y$ for all $x, y \in L$;
  3. Transitivity: $x \sqsubseteq y$ and $y \sqsubseteq z$ imply $x \sqsubseteq z$ for all $x, y, z \in L$.
- Let $(L, \sqsubseteq)$ be a partial order.
  1. An element $x \in L$ is called the least element of L when $x \sqsubseteq y$ for all $y \in L$. The least element is denoted by 0.
  2. An element $x \in L$ is called an upper bound of a subset $X \subseteq L$ if $y \sqsubseteq x$ for all $x \in X$.
  3. x is called the least upper bound of X, written $x = \bigsqcup X$, if
     - x is an upper bound of X;

Basic Lattice Theory

- A partial order is a pair $(L, \sqsubseteq)$ where L is a nonempty set and $\sqsubseteq$ is a binary relation on L satisfying:
  1. Reflexivity: $x \sqsubseteq x$ for all $x \in L$;
  2. Antisymmetry: $x \sqsubseteq y$ and $y \sqsubseteq x$ imply $x = y$ for all $x, y \in L$;
  3. Transitivity: $x \sqsubseteq y$ and $y \sqsubseteq z$ imply $x \sqsubseteq z$ for all $x, y, z \in L$.
- Let $(L, \sqsubseteq)$ be a partial order.
  1. An element $x \in L$ is called the least element of L when $x \sqsubseteq y$ for all $y \in L$. The least element is denoted by 0.
  2. An element $x \in L$ is called an upper bound of a subset $X \subseteq L$ if $y \sqsubseteq x$ for all $x \in X$.
  3. x is called the least upper bound of X, written $x = \bigsqcup X$, if
     - x is an upper bound of X;
     - for any upper bound y of X, $x \sqsubseteq y$.

Basic Lattice Theory (Continued)

- A complete partial order (CPO) is a partial order $(L, \sqsubseteq)$:

Basic Lattice Theory (Continued)

- A complete partial order (CPO) is a partial order $(L, \sqsubseteq)$:
  1. it has the least element 0;

Basic Lattice Theory (Continued)

- A complete partial order (CPO) is a partial order $(L, \sqsubseteq)$:
  1. it has the least element 0;
  2. $\bigsqcup_{n=0}^{\infty} x_n$ exists for any increasing sequence $\{x_n\}$:

$$x_0 \sqsubseteq ... \sqsubseteq x_n \sqsubseteq x_{n+1} \sqsubseteq ....$$

Basic Lattice Theory (Continued)

- A complete partial order (CPO) is a partial order $(L, \sqsubseteq)$:
  1. it has the least element 0;
  2. $\bigsqcup_{n=0}^{\infty} x_n$ exists for any increasing sequence $\{x_n\}$:

$$x_0 \sqsubseteq ... \sqsubseteq x_n \sqsubseteq x_{n+1} \sqsubseteq ....$$

- Let $(L, \sqsubseteq)$ be a CPO. Then a function f from L into itself is continuous if

$$f\left(\bigsqcup_n x_n\right) = \bigsqcup_n f(x_n)$$

  for any increasing sequence $\{x_n\}$ in L.

Knaster-Tarski Theorem

Let $(L, \sqsubseteq)$ be a CPO and function $f : L \to L$ is continuous. Then f has the least fixed point

$$\mu f = \bigsqcup_{n=0}^{\infty} f^{(n)}(0)$$

where

$$\begin{cases} f^{(0)}(0) & = 0, \\ f^{(n+1)}(0) & = f(f^{(n)}(0)) \text{ for } n \geq 0. \end{cases}$$

Domain of Partial Density Operators

$(\mathcal{D}(\mathcal{H}), \sqsubseteq)$ is a CPO with the zero operator $0_{\mathcal{H}}$ as its least element.

Domain of Partial Density Operators

$(\mathcal{D}(\mathcal{H}), \sqsubseteq)$ is a CPO with the zero operator $0_{\mathcal{H}}$ as its least element.

Domain of Quantum Operations

- Each quantum operation in a Hilbert space $\mathcal{H}$ is a continuous function from $(\mathcal{D}(\mathcal{H}), \sqsubseteq)$ into itself.
- Write $\mathcal{QO}(\mathcal{H})$ for the set of quantum operations in Hilbert space $\mathcal{H}$.

Domain of Partial Density Operators

$(\mathcal{D}(\mathcal{H}), \sqsubseteq)$ is a CPO with the zero operator $0_{\mathcal{H}}$ as its least element.

Domain of Quantum Operations

- Each quantum operation in a Hilbert space $\mathcal{H}$ is a continuous function from $(\mathcal{D}(\mathcal{H}), \sqsubseteq)$ into itself.
- Write $\mathcal{QO}(\mathcal{H})$ for the set of quantum operations in Hilbert space $\mathcal{H}$.
- The Löwner order between operators induces a partial order between quantum operations: for any $\mathcal{E}, \mathcal{F} \in \mathcal{QO}(\mathcal{H})$,

Domain of Partial Density Operators

$(\mathcal{D}(\mathcal{H}), \sqsubseteq)$ is a CPO with the zero operator $0_{\mathcal{H}}$ as its least element.

Domain of Quantum Operations

- Each quantum operation in a Hilbert space $\mathcal{H}$ is a continuous function from $(\mathcal{D}(\mathcal{H}), \sqsubseteq)$ into itself.
- Write $\mathcal{QO}(\mathcal{H})$ for the set of quantum operations in Hilbert space $\mathcal{H}$.
- The Löwner order between operators induces a partial order between quantum operations: for any $\mathcal{E}, \mathcal{F} \in \mathcal{QO}(\mathcal{H})$,
  - $\mathcal{E} \sqsubseteq \mathcal{F} \Leftrightarrow \mathcal{E}(\rho) \sqsubseteq \mathcal{F}(\rho)$ for all $\rho \in \mathcal{D}(\mathcal{H})$.

Domain of Partial Density Operators

$(\mathcal{D}(\mathcal{H}), \sqsubseteq)$ is a CPO with the zero operator $0_{\mathcal{H}}$ as its least element.

Domain of Quantum Operations

- Each quantum operation in a Hilbert space $\mathcal{H}$ is a continuous function from $(\mathcal{D}(\mathcal{H}), \sqsubseteq)$ into itself.
- Write $\mathcal{QO}(\mathcal{H})$ for the set of quantum operations in Hilbert space $\mathcal{H}$.
- The Löwner order between operators induces a partial order between quantum operations: for any $\mathcal{E}, \mathcal{F} \in \mathcal{QO}(\mathcal{H})$,
  - $\mathcal{E} \sqsubseteq \mathcal{F} \Leftrightarrow \mathcal{E}(\rho) \sqsubseteq \mathcal{F}(\rho)$ for all $\rho \in \mathcal{D}(\mathcal{H})$.
- $(\mathcal{QO}(\mathcal{H}), \sqsubseteq)$ is a CPO.

Syntactic Approximation

- **abort** denotes a quantum program such that

$$[\![\mathbf{abort}]\!](\rho) = 0_{\mathcal{H}_{\mathrm{all}}} \text{ for all } \rho \in \mathcal{D}(\mathcal{H}).$$

Syntactic Approximation

- **abort** denotes a quantum program such that

$$[\![\mathbf{abort}]\!](\rho) = 0_{\mathcal{H}_{\mathrm{all}}} \text{ for all } \rho \in \mathcal{D}(\mathcal{H}).$$

- Consider a quantum loop

$$\mathbf{while} \equiv \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ S\ \mathbf{od}.$$

Syntactic Approximation

- **abort** denotes a quantum program such that

$$[\![\mathbf{abort}]\!](\rho) = 0_{\mathcal{H}_{\text{all}}} \text{ for all } \rho \in \mathcal{D}(\mathcal{H}).$$

- Consider a quantum loop

$$\mathbf{while} \equiv \mathbf{while}\ M[\overline{q}] = 1\ \mathbf{do}\ S\ \mathbf{od}.$$

- For any integer $k \geq 0$, the kth syntactic approximation $\mathbf{while}^{(k)}$ of **while**:

$$\begin{cases} \mathbf{while}^{(0)} & \equiv\ \mathbf{abort}, \\ \mathbf{while}^{(k+1)} & \equiv\ \mathbf{if}\ M[\overline{q}] = 0 \rightarrow \mathbf{skip} \\ & \qquad \square \qquad\qquad 1 \rightarrow S; \mathbf{while}^{(k)} \\ & \qquad \mathbf{fi} \end{cases}$$

Semantic Function of Loops

$$[\![\mathbf{while}]\!] = \bigsqcup_{k=0}^{\infty} \left[\!\left[\mathbf{while}^{(k)}\right]\!\right],$$

where symbol $\bigsqcup$ stands for the supremum of quantum operations; i.e. the least upper bound in CPO $(\mathcal{QO}(\mathcal{H}_{all}), \sqsubseteq)$.

Semantic Function of Loops

$$[\![\mathbf{while}]\!] = \bigsqcup_{k=0}^{\infty} \left[\!\!\left[\mathbf{while}^{(k)}\right]\!\!\right],$$

where symbol $\bigsqcup$ stands for the supremum of quantum operations; i.e. the least upper bound in CPO $(\mathcal{QO}(\mathcal{H}_{all}), \sqsubseteq)$.

Fixed Point Characterisation

For any $\rho \in \mathcal{D}(\mathcal{H}_{all})$:

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

Termination and Divergence Probabilities

- For any quantum program S and for all partial density operators $\rho \in \mathcal{D}(\mathcal{H}_{all})$:

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

Termination and Divergence Probabilities

- For any quantum program S and for all partial density operators $\rho \in \mathcal{D}(\mathcal{H}_{all})$:

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

- $tr([\![S]\!](\rho))$ is the probability that program S terminates when starting in state ρ.

Semantic Functions as Quantum Operations

- For any quantum program S, its semantic function $[\![S]\!]$ is a quantum operation in $\mathcal{H}_{qvar(S)}$.

Semantic Functions as Quantum Operations

- For any quantum program S, its semantic function $[\![S]\!]$ is a quantum operation in $\mathcal{H}_{qvar(S)}$.
- Let S be a quantum program, $\overline{q}$ a sequence of quantum variables.

Semantic Functions as Quantum Operations

- For any quantum program S, its semantic function $[\![S]\!]$ is a quantum operation in $\mathcal{H}_{qvar(S)}$.
- Let S be a quantum program, $\overline{q}$ a sequence of quantum variables.
  1. The block command defined by S with local variables $\overline{q}$:

$$\textbf{begin local } \overline{q} : S \textbf{ end}.$$

Semantic Functions as Quantum Operations

- For any quantum program S, its semantic function $[\![S]\!]$ is a quantum operation in $\mathcal{H}_{qvar(S)}$.
- Let S be a quantum program, $\overline{q}$ a sequence of quantum variables.
  1. The block command defined by S with local variables $\overline{q}$:

     $$\textbf{begin local } \overline{q} : S \textbf{ end}.$$

  2. The quantum variables of the block command:

     $$qvar\left(\textbf{begin local } \overline{q} : S \textbf{ end}\right) = qvar(S) \setminus \overline{q}.$$

Semantic Functions as Quantum Operations

- For any quantum program S, its semantic function $[\![S]\!]$ is a quantum operation in $\mathcal{H}_{qvar(S)}$.
- Let S be a quantum program, $\overline{q}$ a sequence of quantum variables.
  1. The block command defined by S with local variables $\overline{q}$:

     $$\textbf{begin local } \overline{q} : S \textbf{ end}.$$

  2. The quantum variables of the block command:

     $$qvar\left(\textbf{begin local } \overline{q} : S \textbf{ end}\right) = qvar(S) \setminus \overline{q}.$$

  3. The denotational semantics of the block command is the quantum operation from $\mathcal{H}_{qvar(S)}$ to $\mathcal{H}_{qvar(S)\setminus\overline{q}}$:

     $$[\![\textbf{begin local } \overline{q} : S \textbf{ end}]\!]\left(\rho\right) = tr_{\mathcal{H}_{\overline{q}}}([\![S]\!](\rho))$$

Semantic Functions as Quantum Operations

- For any quantum program S, its semantic function $[\![S]\!]$ is a quantum operation in $\mathcal{H}_{qvar(S)}$.
- Let S be a quantum program, $\overline{q}$ a sequence of quantum variables.
  1. The block command defined by S with local variables $\overline{q}$:
  $$\textbf{begin local } \overline{q} : S \textbf{ end}.$$
  2. The quantum variables of the block command:
  $$qvar\left(\textbf{begin local } \overline{q} : S \textbf{ end}\right) = qvar(S) \setminus \overline{q}.$$
  3. The denotational semantics of the block command is the quantum operation from $\mathcal{H}_{qvar(S)}$ to $\mathcal{H}_{qvar(S)\setminus\overline{q}}$:
  $$[\![\textbf{begin local } \overline{q} : S \textbf{ end}]\!](\rho) = tr_{\mathcal{H}_{\overline{q}}}([\![S]\!](\rho))$$
- For any finite subset V of $qVar$, for any quantum operation $\mathcal{E}$ in $\mathcal{H}_V$, there exists a quantum program (a block command) S such that $[\![S]\!] = \mathcal{E}$.