Foundations of Quantum Programming

Lecture 3: Syntax and Semantics of Quantum Programs

Mingsheng Ying

University of Technology Sydney, Australia
Outline

Syntax

Operational Semantics

Denotational Semantics
Outline

Syntax

Operational Semantics

Denotational Semantics
Classical \textbf{while}-Language

\[ S ::= \text{skip} \mid u := t \mid S_1; S_2 \]
\[ \mid \text{if } b \text{ then } S_1 \text{ else } S_2 \text{ fi} \]
\[ \mid \text{while } b \text{ do } S \text{ od.} \]

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

\[
\text{if } G_1 \rightarrow S_1 \\
\square G_2 \rightarrow S_2 \\
\text{......} \\
\square G_n \rightarrow S_n \\
\text{fi}
\]

or more compactly

\[
\text{if } (\square i \cdot G_i \rightarrow S_i) \text{ fi}
\]
Quantum \textbf{while}-Language

- Fix the alphabet of quantum \textbf{while}-language: A countably infinite set $q\text{Var}$ 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 $q, q', q_0, q_1, q_2, \ldots$ 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:

\[
\text{Boolean} = \mathcal{H}_2, \quad \text{integer} = \mathcal{H}_\infty.
\]
Quantum **while-Language**

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

  \[
  \text{Boolean} = \mathcal{H}_2, \quad \text{integer} = \mathcal{H}_\infty.
  \]

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

  \[
  \mathcal{H}_{\bar{q}} = \bigotimes_{i=1}^{n} \mathcal{H}_{q_i}.
  \]
Quantum Programs

\[ S ::= \text{skip} | q := |0\rangle | \bar{q} := U[\bar{q}] | S_1; S_2 \\
| \text{if} (\Box m \cdot M[\bar{q}] = m \rightarrow S_m) \text{ fi} \\
| \text{while } M[\bar{q}] = 1 \text{ do } S \text{ od.} \]
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

Syntax

Operational Semantics

Denotational Semantics
Notations

- A positive operator $\rho$ is called a partial density operator if $\text{tr}(\rho) \leq 1$. 

- Write $D(H)$ for the set of partial density operators in $H$.

- Write $H_{\text{all}}$ for the tensor product of the state Hilbert spaces of all quantum variables: $H_{\text{all}} = \bigotimes_{q \in q\text{Var}} H_q$.

- Let $q = q_1, \ldots, q_n$ be a quantum register. An operator $A$ in the state Hilbert space $H_q$ of $q$ has a cylindrical extension $A \otimes I$ in $H_{\text{all}}$.

- 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. $\rho \in D(H_{\text{all}})$, denoting the (global) state of quantum variables.

- A transition between quantum configurations: $\langle S, \rho \rangle \rightarrow \langle S', \rho' \rangle$. 

Notations

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

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

\[
\mathcal{H}_{\text{all}} = \bigotimes_{q \in q\text{Var}} \mathcal{H}_q.
\]
Notations

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

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

- A positive operator $\rho$ is called a partial density operator if $\text{tr}(\rho) \leq 1$.
- Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.
- Write $\mathcal{H}_{\text{all}}$ for the tensor product of the state Hilbert spaces of all quantum variables:
  \[ \mathcal{H}_{\text{all}} = \bigotimes_{q \in q\text{Var}} \mathcal{H}_q. \]
- 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 $\mathcal{H}_{\text{all}}$.
- We will use $E$ to denote the empty program; i.e. termination.
- A configuration is a pair $\langle S, \rho \rangle$, where:
A positive operator $\rho$ is called a *partial density operator* if $\text{tr}(\rho) \leq 1$.

Write $\mathcal{D}(\mathcal{H})$ for the set of partial density operators in $\mathcal{H}$.

Write $\mathcal{H}_{\text{all}}$ for the tensor product of the state Hilbert spaces of all quantum variables:

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

Let $\overline{q} = q_1, \ldots, q_n$ 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 $\mathcal{H}_{\text{all}}$.

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 $E$;
A positive operator \( \rho \) is called a *partial density operator* if \( \text{tr}(\rho) \leq 1 \).

Write \( D(\mathcal{H}) \) 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:

\[
\mathcal{H}_{all} = \bigotimes_{q \in q\text{Var}} \mathcal{H}_q.
\]

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. \( \rho \in D(\mathcal{H}_{all}) \), denoting the (global) state of quantum variables.
Notations

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

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

\[(IN)\quad \langle q := |0\rangle, \rho \rangle \rightarrow \langle E, \rho^q_0 \rangle\]

where

\[
\rho^q_0 = \begin{cases} 
|0\rangle_q\langle 0|\rho|0\rangle_q\langle 0| + |0\rangle_q\langle 1|\rho|1\rangle_q\langle 0| & \text{if } \text{type}(q) = \text{Boolean}, \\
\sum_{n=-\infty}^{\infty} |0\rangle_q\langle n|\rho|n\rangle_q\langle 0| & \text{if } \text{type}(q) = \text{integer}.
\end{cases}
\]

\[(UT)\quad \langle \bar{q} := U[q], \rho \rangle \rightarrow \langle E, U\rho U^\dagger \rangle\]
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) \[
\frac{\langle \text{if } (\Box m \cdot M[q] = m \rightarrow S_m) \text{ fi}, \rho \rangle \rightarrow \langle S_m, M_m \rho M^\dagger_m \rangle}{\text{for each possible outcome } m \text{ of measurement } M = \{M_m\}.}
\]

(L0) \[
\frac{\langle \text{while } M[q] = 1 \text{ do } S \text{ od}, \rho \rangle \rightarrow \langle E, M_0 \rho M^\dagger_0 \rangle}{\}
\]

(L1) \[
\frac{\langle \text{while } M[q] = 1 \text{ do } S \text{ od}, \rho \rangle \rightarrow \langle S; \text{while } M[q] = 1 \text{ do } S \text{ od}, M_1 \rho M^\dagger_1 \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 $\rho$ is a finite or infinite sequence of configurations:

   \[
   \langle S, \rho \rangle \rightarrow \langle S_1, \rho_1 \rangle \rightarrow \ldots \rightarrow \langle S_n, \rho_n \rangle \rightarrow \langle S_{n+1}, \rho_{n+1} \rangle \rightarrow \ldots
   \]

   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 $\rho$ is a finite or infinite sequence of configurations:

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

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 $\rho$. 

▶ If a computation is finite and its last configuration is $\langle E, \rho' \rangle$, then we say that it terminates in $\rho'$.

▶ If it is infinite, then we say that it diverges.
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 $\rho$ 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 $\rho$.

   - If a computation is finite and its last configuration is $\langle E, \rho' \rangle$, then we say that it terminates in $\rho'$.
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 $\rho$ is a finite or infinite sequence of configurations:

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

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 $\rho$.
   - If a computation is finite and its last configuration is $\langle E, \rho' \rangle$, then we say that it terminates in $\rho'$.
   - If it is infinite, then we say that it diverges.
Outline

Syntax

Operational Semantics

Denotational Semantics
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 $\langle S_1, \rho_1 \rangle, \ldots, \langle S_{n-1}, \rho_{n-1} \rangle$ 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 $n \geq 0$. 


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:

\[
\langle S, \rho \rangle \rightarrow^n \langle S', \rho' \rangle.
\]

- 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

\[
\llbracket S \rrbracket : \mathcal{D}(\mathcal{H}_{all}) \rightarrow \mathcal{D}(\mathcal{H}_{all})
\]

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

$$\left\langle S \right\rangle (\lambda_1 \rho_1 + \lambda_2 \rho_2) = \lambda_1 \left\langle S \right\rangle (\rho_1) + \lambda_2 \left\langle S \right\rangle (\rho_2).$$
Structural Representation

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

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

2.
Structural Representation

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

2.

2.1 If $\text{type}(q) = \text{Boolean}$, then

$$\llbracket q := |0\rangle \rrbracket (\rho) = |0\rangle_q \langle 0|_q \rho \langle 0|_q + |0\rangle_q \langle 1|_q \rho \langle 1|_q \langle 0|.$$
Structural Representation

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

2. 
   
   2.1 If $\text{type}(q) = \text{Boolean}$, then
   
   $\llbracket q := \left| 0 \right\rangle \rrbracket(\rho) = \left| 0 \right\rangle_q \left\langle 0 | \rho | 0 \right\rangle_q + \left| 0 \right\rangle_q \left\langle 1 | \rho | 1 \right\rangle_q \left| 0 \right\rangle.$

   2.2 If $\text{type}(q) = \text{integer}$, then
   
   $\llbracket q := \left| 0 \right\rangle \rrbracket(\rho) = \sum_{n=-\infty}^{\infty} \left| 0 \right\rangle_q \left\langle n | \rho | n \right\rangle_q \left| 0 \right\rangle.$
Structural Representation

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

2. 
   
   2.1 If $\text{type}(q) = \text{Boolean}$, then
   
   $$\llbracket q := |0\rangle \rrbracket(\rho) = |0\rangle_q \langle 0|\rho|0\rangle_q \langle 0| + |0\rangle_q \langle 1|\rho|1\rangle_q \langle 0|.$$ 

   2.2 If $\text{type}(q) = \text{integer}$, then
   
   $$\llbracket q := |0\rangle \rrbracket(\rho) = \sum_{n=-\infty}^{\infty} |0\rangle_q \langle n|\rho|n\rangle_q \langle 0|.$$ 

3. $\llbracket \overline{q} := U[\overline{q}] \rrbracket(\rho) = U\rho U^\dagger$. 
Structural Representation

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

2. \( 2.1 \text{ If } \text{type}(q) = \text{Boolean}, \text{ then} \)

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

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

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

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

4. \( [S_1; S_2] (\rho) = [S_2] ([S_1] (\rho)). \)
Structural Representation

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

2. 
   2.1 If $\text{type}(q) = \text{Boolean}$, then
   
   $\llbracket q := |0\rangle \rrbracket (\rho) = |0\rangle_q \langle 0|\rho|0\rangle_q \langle 0| + |0\rangle_q \langle 1|\rho|1\rangle_q \langle 0|$.

   2.2 If $\text{type}(q) = \text{integer}$, then
   
   $\llbracket q := |0\rangle \rrbracket (\rho) = \sum_{n=-\infty}^{\infty} |0\rangle_q \langle n|\rho|n\rangle_q \langle 0|.$

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

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

5. $\llbracket \textbf{if } (\square m \cdot M[\overline{q}] = m \rightarrow S_m ) \textbf{fi} \rrbracket (\rho) = \sum_m \llbracket S_m \rrbracket (M_m \rho M_m^\dagger)$.
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:

  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 \(x \in L\);
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 \(x = \bigvee X\), 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 \(x = \bigvee X\), 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 \(x = \bigvee X\), 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 \((L, \sqsubseteq)\):

1. it has the least element 0;
2. \(\bigsqcup \infty n x_n = 0x_n\) exists for any increasing sequence \(\{x_n\}\):
   
   \[
   x_0 \sqsubseteq \ldots \sqsubseteq x_n \sqsubseteq x_{n+1} \sqsubseteq \ldots
   \]

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

\[
\bigvee n x_n = \bigvee n f(x_n)
\]

for any increasing sequence \(\{x_n\}\) in \(L\).
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 \(\{x_n\}\):

\[
x_0 \sqsubseteq \ldots \sqsubseteq x_n \sqsubseteq x_{n+1} \sqsubseteq \ldots
\]
Basic Lattice Theory (Continued)

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

\[
f \left( \bigcup_n x_n \right) = \bigcup_n f(x_n)
\]

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

$$\begin{align*}
  f^{(0)}(0) &= 0, \\
  f^{(n+1)}(0) &= f(f^{(n)}(0)) \text{ for } n \geq 0.
\end{align*}$$
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 \(QO(\mathcal{H})\) 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 \(QO(\mathcal{H})\) 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 \(E, F \in QO(\mathcal{H})\),

\[ E \sqsubseteq F \iff E(\rho) \sqsubseteq F(\rho) \text{ for all } \rho \in \mathcal{D}(\mathcal{H}). \]
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 \(\mathcal{E}, \mathcal{F} \in \mathcal{QO}(\mathcal{H})\),
  - \(\mathcal{E} \sqsubseteq \mathcal{F} \iff \mathcal{E}(\rho) \sqsubseteq \mathcal{F}(\rho)\) 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 \(\mathcal{E}, \mathcal{F} \in \mathcal{QO}(\mathcal{H})\),
  - \(\mathcal{E} \sqsubseteq \mathcal{F} \iff \mathcal{E}(\rho) \sqsubseteq \mathcal{F}(\rho)\) 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

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

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

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

- Consider a quantum loop

\[
\text{while } \equiv \text{while } M[\bar{q}] = 1 \text{ do } S \text{ od.}
\]
Syntactic Approximation

- **abort** denotes a quantum program such that
  \[ \langle \text{abort} \rangle (\rho) = 0_{\mathcal{H}_{\text{all}}} \text{ for all } \rho \in \mathcal{D}(\mathcal{H}). \]

- Consider a quantum loop
  \[
  \text{while} \equiv \text{while } M[\bar{q}] = 1 \text{ do } S \text{ od}.
  \]

- For any integer \( k \geq 0 \), the \( k \)th syntactic approximation \( \text{while}^{(k)} \) of \( \text{while} \):
  \[
  \begin{cases}
    \text{while}^{(0)} & \equiv \text{abort}, \\
    \text{while}^{(k+1)} & \equiv \text{if } M[\bar{q}] = 0 \rightarrow \text{skip} \\
    & \quad \square 1 \rightarrow S; \text{while}^{(k)} \\
    & \quad \text{fi}
  \end{cases}
  \]
Semantic Function of Loops

\[
\begin{align*}
\llbracket \text{while} \rrbracket &= \bigcup_{k=0}^{\infty} \llbracket \text{while}^{(k)} \rrbracket, \\
\text{where symbol } \bigcup \text{ stands for the supremum of quantum operations; i.e. the least upper bound in CPO } (QO(\mathcal{H}_{all}), \sqsubseteq).
\end{align*}
\]
Semantic Function of Loops

\[ \llbracket \text{while} \rrbracket = \bigsqcup_{k=0}^{\infty} \llbracket \text{while}^{(k)} \rrbracket, \]

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

Fixed Point Characterisation

For any \( \rho \in D(H_{all}) \):

\[ \llbracket \text{while} \rrbracket (\rho) = M_0 \rho M_0^\dagger + \llbracket \text{while} \rrbracket \left( \llbracket S \rrbracket \left( M_1 \rho M_1^\dagger \right) \right). \]
Termination and Divergence Probabilities

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

$$tr(\llbracket S \rrbracket(\rho)) \leq tr(\rho).$$
Termination and Divergence Probabilities

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

- $\text{tr}([S](\rho))$ is the probability that program $S$ terminates when starting in state $\rho$. 
Semantic Functions as Quantum Operations

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

- For any finite subset $V$ of $qVar$, for any quantum operation $E$ in $\mathcal{H}_V$, there exists a quantum program (a block command) $S$ such that $[[S]] = E$. 

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, $\bar{q}$ a sequence of quantum variables.
Semantic Functions as Quantum Operations

- For any quantum program $S$, its semantic function $[[S]]$ is a quantum operation in $H_{qvar}(S)$.

- Let $S$ be a quantum program, $\bar{q}$ a sequence of quantum variables.
  1. The block command defined by $S$ with local variables $\bar{q}$:

    \[
    \text{begin local } \bar{q} : S \text{ end.}
    \]
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, \( \bar{q} \) a sequence of quantum variables.
  1. The block command defined by \( S \) with local variables \( \bar{q} \):

      ```
      begin local \( \bar{q} : S \) end.
      ```
  2. The quantum variables of the block command:

      \[
      qvar \left( \texttt{begin local } \bar{q} : S \texttt{ end} \right) = qvar(S) \setminus \bar{q}.
      \]
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, $\bar{q}$ a sequence of quantum variables.
  1. The block command defined by $S$ with local variables $\bar{q}$:
     
     \[
     \text{begin local } \bar{q} : S \text{ end.}
     \]

  2. The quantum variables of the block command:
     
     \[
     qvar(\text{begin local } \bar{q} : S \text{ end}) = qvar(S) \setminus \bar{q}.
     \]

  3. The denotational semantics of the block command is the quantum operation from $\mathcal{H}_{qvar(S)}$ to $\mathcal{H}_{qvar(S) \setminus \bar{q}}$:
     
     \[
     [[\text{begin local } \bar{q} : S \text{ end}]](\rho) = tr_{\mathcal{H}_{\bar{q}}}([[S]](\rho))
     \]
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, $\bar{q}$ a sequence of quantum variables.

1. The block command defined by $S$ with local variables $\bar{q}$:

   \[
   \text{begin local } \bar{q} : S \text{ end.}
   \]

2. The quantum variables of the block command:

   \[
   qvar(\text{begin local } \bar{q} : S \text{ end}) = qvar(S) \setminus \bar{q}.
   \]

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

   \[
   [[\text{begin local } \bar{q} : S \text{ end}])(\rho) = tr_{\mathcal{H}_{\bar{q}}}([[S])(\rho))
   \]

▶ 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 $[[S]] = \mathcal{E}$. 