Chapter 1
Invariance: Proof Methods

For assertion \( q \) and SPL program \( P \)
show \( P \models \Box q \)
(i.e., \( q \) is \( P \)-invariant)
Proving Invariances

Definitions

Recall:

- the variables of assertion:
  - free (flexible) system variables
    \[ V = Y \cup \{ \pi \} \]
  where \( Y \) are the program variables and \( \pi \) is the control variable
  - quantified (rigid) specification variables

- \( q' \) is the primed version of \( q \), obtained by replacing each free occurrence of a system variable \( y \in V \) by its primed version \( y' \).

- \( \rho_\tau \) is the transition relation of \( \tau \), expressing the relation holding between a state \( s \) and any of its \( \tau \)-successors \( s' \in \tau(s) \).

Verification Conditions
(proof obligations)

standard verification condition

For assertions \( \varphi, \psi \) and transition \( \tau \),

\[ \{ \varphi \} \tau \{ \psi \} \] ("Hoare triple") stands for the state formula

\[ \rho_\tau \land \varphi \rightarrow \psi' \]

"Verification condition (VC) of \( \varphi \) and \( \psi \) relative to transition \( \tau \)"

\[
\begin{array}{c c c}
\varphi & \tau & \psi \\
\downarrow & \downarrow & \downarrow \\
j & j + 1 & j + 1
\end{array}
\]
Verification Conditions (Con't)

Example:

\[ \rho: x \geq 0 \land y' = x + y \land x' = x \]

\[ \varphi: y = 3 \quad \psi: y = x + 3 \]

Then \{\varphi\} \tau \{\psi\}:

\[ x \geq 0 \land y' = x + y \land x' = x \land y = 3 \quad \rho \quad \tau \quad \varphi \rightarrow y' = x' + 3 \quad \psi' \]

Verification Conditions (Con't)

- for \( \tau \in T \) in \( P \):
  \[ \{\varphi\} \tau \{\psi\}: \quad \rho \tau \land \varphi \rightarrow \psi' \]
  "\( \tau \) leads from \( \varphi \) to \( \psi \) in \( P \)"

- for \( T \) in \( P \):
  \[ \{\varphi\} T \{\psi\}: \quad \{\varphi\} \tau \{\psi\} \quad \text{for every } \tau \in T \]
  "\( T \) leads from \( \varphi \) to \( \psi \) in \( P \)"

Claim (Verification Condition)

If \( \{\varphi\} \tau \{\psi\} \) is \( P \)-state valid,
then every \( \tau \)-successor of a \( \varphi \)-state is a \( \psi \)-state.
Verification Conditions (Con’t)

Special Cases

• while, conditional

\[ \rho: \rho^T \lor \rho^F \]

\[ \{ \varphi \} \tau^T \{ \psi \}: \rho^T \land \varphi \rightarrow \psi' \]

\[ \{ \varphi \} \tau^F \{ \psi \}: \rho^F \land \varphi \rightarrow \psi' \]

- idle

\[ \{ \varphi \} \tau_I \{ \varphi \}: \rho_{\tau_I} \land \varphi \rightarrow \varphi' \]

always valid, since
\[ \rho_{\tau_I} \rightarrow v' = v \quad \text{for all } v \in V, \]
so \( \varphi' = \varphi \).

Verification Conditions (Con’t)

Substituted Form of Verification Condition

Transition relation can be written as

\[ \rho: C \land (V' = E) \]

where

- \( C \): enabling condition
- \( V' \): primed variable list
- \( E \): expression list

• The substituted form of

verification condition \( \{ \varphi \} \tau \{ \psi \} \):

\[ C \land \varphi \rightarrow \psi[E/V] \]

where

\[ \psi[E/V]: \text{ replace each variable } v \in V \text{ in } \psi \text{ by the corresponding } e \in E \]

Note: No primed variables!

The substituted form of a verification condition is P-state
valid iff the standard form is
Verification Conditions (Con’t)

Example:

\[ \rho \tau : x \geq 0 \land y' = x + y \land x' = x \]
\[ \varphi : y = 3 \quad \psi : y = x + 3 \]

Standard

\[ x \geq 0 \land \underbrace{y' = x + y \land x'}_{\rho \tau} \land \underbrace{y = 3}_{\varphi} \]
\[ \rightarrow \underbrace{y' = x'}_{\psi'} + 3 \]

Substituted

\[ \underbrace{x \geq 0}_{C_{\tau}} \land \underbrace{y = 3}_{\varphi} \rightarrow \underbrace{x + y = x + 3}_{\psi[E/V]} \]

Verification Conditions (Con’t)

Example:

\[ \varphi: x = y \quad \psi: x = y + 1 \]
\[ \rho_{\tau}: \underbrace{x \geq 0}_{C_{\tau}} \land \underbrace{(x', y') = (x + 1, y)}_{E} \]

The substituted form of \( \{\varphi\} \tau \{\psi\} \) is

\[ \underbrace{x \geq 0}_{C_{\tau}} \land \underbrace{x = y}_{\varphi} \rightarrow \underbrace{(x = y + 1)[(x + 1, y)/(x, y)]}_{\psi[E/V]} \]

or equivalently

\[ x \geq 0 \land x = y \rightarrow x + 1 = y + 1 \]
Simplifying Control Expressions

move\((L_1, L_2)\): \(L_1 \subseteq \pi \wedge \pi' = (\pi - L_1) \cup L_2\)

e.g., for \(L_1 = \{\ell_1\}, L_2 = \{\ell_2\}\)

move\((\ell_1, \ell_2)\): \(\ell_1 \in \pi \wedge \pi' = (\pi - \{\ell_1\}) \cup \{\ell_2\}\)

Consequences implied by move\((L_1, L_2)\):

- for every \([\ell] \in L_1\)
  \(at_-'\ell = T\) (i.e., \([\ell] \in \pi\))

- for every \([\ell] \in L_2\)
  \(at'_-\ell = T\) (i.e., \([\ell] \in \pi'\))

- for every \([\ell] \in L_1 - L_2\)
  \(at_-'\ell = T\) (i.e., \([\ell] \in \pi\)) and \(at'_-\ell = F\) (i.e., \([\ell] \notin \pi'\))

- for every \(\ell \notin L_1 \cup L_2\)
  \(at'_-\ell = at_-'\ell\) (i.e., \([\ell] \in \pi, \pi'\) or \([\ell] \notin \pi, \pi'\))

Proving invariance properties: \(P \vDash \Box q\)

We want to show that for every computation of \(P\)
\[\sigma : s_0, s_1, s_2, \ldots\]
assertion \(q\) holds in every state \(s_j, j \geq 0\), i.e., \(s_j \not\models q\).

Recall:
A sequence \(\sigma : s_0, s_1, s_2, \ldots\) is a computation if the following hold (from Chapter 0):

1. Initiality: \(s_0 \not\models \Theta\)

2. Consecution: For each \(j \geq 0\),
   \(s_{j+1}\) is a \(\tau\)-successor of \(s_j\) for some \(\tau \in T\)
   \((s_{j+1} \in \tau(s_j))\)

3, 4. Fairness conditions are respected.

Note: Truth of safety properties over programs does not depend on fairness conditions.
Proving invariance properties (Con’t)

This definition suggests a way to prove invariance properties $\square q$:

1. Base case:
   Prove that $q$ holds initially
   $\Theta \rightarrow q$
   i.e., $q$ holds at $s_0$.

2. Inductive step:
   prove that $q$ is preserved by all transitions
   $q \land \rho_\tau \rightarrow q'$ for all $\tau \in T$
   $\{q\} \tau \{q\}$
   i.e., if $q$ holds at $s_j$, then it holds at every $\tau$-successor $s_{j+1}$.

Rule B-INV (basic invariance)

Show $P \models \Box q$ (i.e. $q$ is $P$-invariant)

For assertion $q$,

B1. $P \models \Theta \rightarrow q$

B2. $P \models \{q\} \tau \{q\}$

$P \models \Box q$

where B2 stands for

$P \models \{q\} \tau \{q\}$ for every $\tau \in T$

- The rule states that if we can prove the $P$-state validity of $\Theta \rightarrow q$ and $\{q\} \tau \{q\}$
  then we can conclude that $\Box q$ is $P$-valid.

- Thus the proof of a temporal property is reduced to the proof of $1 + |T|$
  first-order verification conditions.
Example 1: request-release

Example 1: request-release (Con’t)

\[ \Theta: \quad x = 1 \land \pi = \{\ell_0\} \Rightarrow \frac{\Box x \geq 0}{q} \]

B1: \[ x = 1 \land \pi = \{\ell_0\} \Rightarrow \frac{x \geq 0}{q} \]
holds since \( x = 1 \Rightarrow x \geq 0 \)

B2:

\[ \tau_{\ell_0}: \quad \frac{x \geq 0 \land \text{move}(\ell_0, \ell_1) \land x > 0 \land x' = x - 1}{\rho_{\tau_{\ell_0}}} \Rightarrow \frac{x' \geq 0}{q'} \]
holds since \( x > 0 \Rightarrow x - 1 \geq 0 \)

\[ \tau_{\ell_1}: \quad \frac{x \geq 0 \land \text{move}(\ell_1, \ell_2) \land x' = x}{\rho_{\tau_{\ell_1}}} \Rightarrow \frac{x' \geq 0}{q'} \]
holds since \( x \geq 0 \Rightarrow x \geq 0 \)

\[ \tau_{\ell_2}: \quad \frac{x \geq 0 \land \text{move}(\ell_2, \ell_3) \land x' = x + 1}{\rho_{\tau_{\ell_2}}} \Rightarrow \frac{x' \geq 0}{q'} \]
holds since \( x \geq 0 \Rightarrow x + 1 \geq 0 \)
Example 1: request-release (Con’t)

\[
\begin{array}{ll}
\text{local } x: \text{ integer where } x = 1 \\
\ell_0: \text{ request } x \\
\ell_1: \text{ critical} \\
\ell_2: \text{ release } x
\end{array}
\]

We proved
\[ P \models \square x \geq 0 \]
using B-INV.

Now we want to prove
\[ P \models \square (\text{at}_- \ell_1 \rightarrow x = 0) \]

Example 1: request-release (Con’t)

Attempted proof:

\[
\text{B1: } x = 1 \land \pi = \{\ell_0\} \rightarrow (\text{at}_- \ell_1 \rightarrow x = 0)
\]

holds since \( \pi = \{\ell_0\} \rightarrow \text{at}_- \ell_1 = F \)

\[
\text{B2: } \{q\} \tau_{\ell_0} \{q\} \\
\text{at}_- \ell_1 \rightarrow x = 0 \land \text{move}(\ell_0, \ell_1) \land x > 0 \land x' = x - 1 \rightarrow \text{at}'_1 \rightarrow x' = 0 \\
\]

We have \( \text{move}(\ell_0, \ell_1) \rightarrow \text{at}_- \ell_1 = F, \text{at}'_1 \ell_1 = T \)

BUT
\( (F \rightarrow x = 0) \land x > 0 \land x' = x - 1 \rightarrow (T \rightarrow x' = 0) \)

Cannot prove: not state-valid

What is the problem?
We need a stronger rule.
Strategies for invariance proofs

Rule b-inv (basic invariance)

For assertion $q$,

B1. $P \models \Theta \rightarrow q$

B2. $P \models \{q\} T \{q\}$

$P \models \square q$

- $q$ is inductive if B1 and B2 are (state) valid

- By rule b-inv,
  every inductive assertion $q$ is $P$-invariant

- The converse is not true

Example: In REQUEST-RELEASE

$at_{-\ell_1} \rightarrow x = 0$

is $P$-invariant, but not inductive

Rule b-inv (Con’t)

The problem is:

“The invariant is not inductive”
i.e., it is not strong enough to be preserved by all transitions.

Another way to look at it is to observe that

$\{q\} \tau_{\ell_0} \{q\}$

is not state valid, but it is $P$-state valid, i.e., it is true in all $P$-accessible states, since in all $P$-accessible states

$x = 1$ when at location $\ell_0$.

This suggests two strategies to overcome this problem:

- strengthening
- incremental proof
Strategy 1: Strengthening

Find a stronger assertion \( \varphi \) that is inductive and implies the assertion \( q \) we want to prove.

\[
\Sigma
\]

\( P \)-accessible

In Chapter 2 it will be shown that there always exists such an assertion \( \varphi \).

Example:

To show
\[
\Box (\text{at}_{-l_1} \rightarrow x = 0)
\]
strengthen \( q \) to
\[
\varphi : (\text{at}_{-l_1} \rightarrow x = 0) \land (\text{at}_{-l_0} \rightarrow x = 1)
\]
and show
\[
\Box (\text{at}_{-l_1} \rightarrow x = 0) \land (\text{at}_{-l_0} \rightarrow x = 1)
\]
by rule \text{B-INV}.
Strategy 1: Strengthening (Con’t)

The strengthening strategy relies on the following rule, MON-I, which, combined with B-INV leads to the general invariance rule INV.

Rule MON-I (Monotonicity)

For assertions \( q_1, q_2 \),

\[
\begin{align*}
P &\not\models \Box q_1 & P &\not\models q_1 \rightarrow q_2 \\
P &\models \Box q_2
\end{align*}
\]

Strategy 1: Strengthening (Con’t)

Rule INV (general invariance)

For assertions \( q, \varphi \)

\[
\begin{align*}
&\text{I1. } P \not\models \varphi \rightarrow q \\
&\text{I2. } P \not\models \Theta \rightarrow \varphi \\
&\text{I3. } P \not\models \{\varphi\} T \{\varphi\} \\
P &\models \Box q
\end{align*}
\]
**Soundness:** If we manage to prove \( \square q \) using the INV rule for some program \( P \), is \( q \) really an invariant for the program?

We can prove that this is indeed the case. So INV rule is *sound*.

**Completeness:** What if \( q \) is an invariant for a program \( P \) but there is no way of proving it under the INV rule?

We can prove that this never happens. There always exists an appropriate \( \varphi \). In other words INV rule is *complete*.

---

**Strategy 1: Strengthening (Con’t)**

Motivation:

\[
P \vdash \square \varphi \quad \text{(by I2 and I3)}
\]

\[
P \not\vdash \varphi \rightarrow q \quad \text{(by I1)}
\]

Therefore,

\[
P \vdash \square q \quad \text{(by MON-1)}
\]

i.e., this rule requires that \( \square \varphi \) holds and \( \varphi \) implies \( q \), then \( \square q \) can be concluded to hold by monotonicity.
Control Invariants

Some control invariants that can always be used (without mentioning them)

• CONFLICT:
  for labels $\ell_i, \ell_j$ that are in conflict (i.e., not $\sim_L$, not parallel):
  $\Box \neg (\text{at}_-\ell_i \land \text{at}_-\ell_j)$

• SOMEWHERE:
  for the set of labels $\mathcal{L}_i$ in a top-level process:
  $\Box \lor_{\ell \in \mathcal{L}_i} \text{at}_-\ell$

• EQUAL:
  for labels $l, m$, s.t. $l \sim_L m$:
  $\Box (\text{at}_-l \leftrightarrow \text{at}_-m)$

Control Invariants (Con’t)

• PARALLEL:
  for substatement $[S_1||S_2]$:
  $\Box (\text{in}_-_S_1 \leftrightarrow \text{in}_-_S_2)$
  i.e., if control is in $S_1$ it must also be in $S_2$ and vice versa.

Example:
Using the invariant CONFLICT,

$move(\ell_2, \ell_3)$ implies $l_0 \not\in \pi$, $l_1 \not\in \pi$, $l_3 \not\in \pi$

$l_0 \not\in \pi'$, $l_1 \not\in \pi'$, $l_2 \not\in \pi'$
Example:

We proposed the strengthened invariant
\[ \varphi : (at_{-\ell_0} \rightarrow x = 1) \land (at_{-\ell_1} \rightarrow x = 0) \]

Consider \( \{ \varphi \} \tau_{\ell_0} \{ \varphi \} : \)

\[ (at_{-\ell_0} \rightarrow x = 1) \land (at_{-\ell_1} \rightarrow x = 0) \land \]

\[ move(\ell_0, \ell_1) \land x > 0 \land x' = x - 1 \]

\[ \xrightarrow{\rho_{\tau_{\ell_0}}} \]

\[ \rightarrow (at'_{-\ell_0} \rightarrow x' = 1) \land (at'_{-\ell_1} \rightarrow x' = 0) \]

\[ \xrightarrow{\varphi} \]

\[ move(\ell_0, \ell_1) \text{ implies } \ell_0 \in \pi, \ell_1 \not\in \pi, \ell_1 \in \pi', \ell_0 \not\in \pi' \]

Therefore

\[ (T \rightarrow x = 1) \land (F \rightarrow \ldots) \land \ldots \land x' = x - 1 \land \ldots \]

\[ \rightarrow (F \rightarrow \ldots) \land (T \rightarrow x' = 0) \]

holds.

Example (Con’t):

Consider \( \{ \varphi \} \tau_{\ell_2} \{ \varphi \} : \)

\[ (at_{-\ell_0} \rightarrow x = 1) \land (at_{-\ell_1} \rightarrow x = 0) \land \]

\[ move(\ell_2, \ell_3) \land x' = x + 1 \]

\[ \xrightarrow{\rho_{\tau_{\ell_2}}} \]

\[ \rightarrow (at'_{-\ell_0} \rightarrow x' = 1) \land (at'_{-\ell_1} \rightarrow x' = 0) \]

\[ \xrightarrow{\varphi'} \]

\[ move(\ell_2, \ell_3) \text{ implies } \ell_3 \in \pi' \]

and by CONFLICT invariants \( \ell_0, \ell_1 \not\in \pi' \).

Therefore

\[ \ldots \land \ldots \rightarrow (F \rightarrow x' = 1) \land (F \rightarrow x' = 0) \]

holds.

\( \{ \varphi \} \tau_{\ell_2} \{ \varphi \} \) is not state-valid, but it is \( P \)-state valid. Why?
Strategy 2: Incremental proof

Use previously proven invariances $\chi$ to exclude parts of the state space from consideration.

Example:

To show $/BC(at_{-\ell_1} \rightarrow x = 0)$, prove first (separately) by rule B-INV

$$\Box(at_{-\ell_0} \rightarrow x = 1),$$

then show $\Box(at_{-\ell_1} \rightarrow x = 0)$ by rule B-INV, but add the conjunct

$$at_{-\ell_0} \rightarrow x = 1$$

to the antecedent of all verification conditions.

(Example continues...)
**Strategy 2: Incremental proof (Con’t)**

**Example: (cont’d)**

E.g., to show \( \{ \chi \land q \} \tau_{\ell_0} \{ q \} \), prove

\[
\begin{align*}
&\text{at}_{-\ell_0} \rightarrow x = 1 \land \text{at}_{-\ell_1} \rightarrow x = 0 \land \\
&\text{move}(\ell_0, \ell_1) \land x > 0 \land x' = x - 1 \\
&\rho_{\tau_{\ell_0}} \\
&\rightarrow \text{at}'_{-\ell_1} \rightarrow x' = 0
\end{align*}
\]

---

**Strategy 2: Incremental proof (Con’t)**

In an incremental proof we use previously proven properties to eliminate parts of the state space (non \( P \)-accessible states) from consideration, relying on the following rules:

**Rule sv-psv:** from state validities to \( P \)-state validities

For assertions \( q_1, q_2 \) and \( \chi \),

\[
\begin{align*}
P &\not\models \Box \chi \\
P &\not\models \chi \land q_1 \rightarrow q_2 \\
P &\models \Box (q_1 \rightarrow q_2)
\end{align*}
\]

**Rule i-con:** Conjunction

For assertions \( q_1 \) and \( q_2 \),

\[
\begin{align*}
P &\models \Box q_1 \\
P &\models \Box q_2 \\
P &\models \Box (q_1 \land q_2)
\end{align*}
\]
Strategy 2: Incremental proof (Con’t)

Example: Program MUX-SEM
(mutual exclusion by semaphores)

local y: integer where y = 1

\[\begin{align*}
P_1 &:: [\ell_0: \text{loop forever do} \left[ \ell_1: \text{noncritical} \right] \quad \ell_2: \text{request y} \quad \ell_3: \text{critical} \quad \ell_4: \text{release y} \right] \quad || \quad P_2:: [m_0: \text{loop forever do} \left[ m_1: \text{noncritical} \right] \quad m_2: \text{request y} \quad m_3: \text{critical} \quad m_4: \text{release y} \right]
\end{align*}\]

Prove mutual exclusion
\[\square \neg (at_{\ell_3} \land at_{m_3})\]

Program MUX-SEM (Con’t)

3 steps: \(\square (y \geq 0)\)
\[\varphi_1\]
\[\square \left( at_{-\ell_3,4} + at_{-m_3,4} + y = 1 \right) \varphi_2\]
\[\square \neg (at_{-\ell_3} \land at_{-m_3}) \quad p\]

where \(F = 0, T = 1\).

Let \(\pi_\ell: \pi \cap \{\ell_0, \ldots, \ell_4\}\)
\(\pi_m: \pi \cap \{m_0, \ldots, m_4\}\)

By control invariants (CONFLICT, SOMEWHERE and PARALLEL)

\(|\pi_\ell| = |\pi_m| = 1|
Program MUX-SEM (Con’t)

Step 1: $\Box (y \geq 0)
\psi_1$
by rule B-INV

B1. $\pi = \{\ell_0, m_0\} \land y = 1 \rightarrow y \geq 0
\psi_1$

B2. $\rho \land y \geq 0 \rightarrow y' \geq 0$

check only $\ell_2, \ell_4, m_2, m_4$
(“$y$-modifiable transitions”)

Program MUX-SEM (Con’t)

$\ell_2$: $\text{move}(\ell_2, \ell_3) \land y > 0 \land y' = y-1 \land y \geq 0
\rho_r \land \psi \rightarrow y' \geq 0
\psi'$
holds since $y > 0 \rightarrow y-1 \geq 0$

$\ell_4$: $\text{move}(\ell_4, \ell_0) \land y' = y+1 \land y \geq 0 \rightarrow y' \geq 0
\rho_r \land \psi \rightarrow \psi'$
holds since $y \geq 0 \rightarrow y+1 \geq 0$.

Similarly for $m_2, m_4$. 
Program MUX-SEM (Con’t)

Step 2:

\(\square(at_\ell 3,4 + at_m 3,4 + y = 1)\)

by rule B-INV

B1. \(\pi = \{\ell_0, m_0\} \land y = 1 \rightarrow \)

\(\varphi_2\)

B2. \(\rho_\tau \land \varphi_2 \rightarrow \varphi'_2\)

\(\rho_{\ell_0} \land 0 + at_m 3,4 + y = 1 \rightarrow \)

\(0 + at_m 3,4 + y = 1\)

\(\rho_{\ell_1} \land 0 + at_m 3,4 + y = 1 \rightarrow \)

\(0 + at_m 3,4 + y = 1\)

\(\rho_{\ell_2} \land 0 + at_m 3,4 + y = 1 \rightarrow \)

\(1 + at_m 3,4 + (y-1) = 1\)

\(\rho_{\ell_3} \land 1 + at_m 3,4 + y = 1 \rightarrow \)

\(1 + at_m 3,4 + y = 1\)

\(\rho_{\ell_4} \land 1 + at_m 3,4 + y = 1 \rightarrow \)

\(0\)

\(at'_{\ell 3,4} + at'_m 3,4 + (y+1) = 1\)

6-38
Program MUX-SEM (Con’t)

**Step 3:** Show $P \vDash \Box \neg (\text{at}_-\ell_{3} \land \text{at}_-m_{3})$

- **By i-con**

  $P \vDash \Box \varphi_1, P \vDash \Box \varphi_2$

  \[ P \vDash \Box (\varphi_1 \land \varphi_2) \]

- **By mon-i**

  $P \vDash \Box (\varphi_1 \land \varphi_2)$

  \[ P \vDash y \geq 0 \land \varphi_1 \land \varphi_2 \]

  \[ P \vDash \varphi_1 \land \varphi_2 \land (\text{at}_-\ell_{3,4} + \text{at}_-m_{3,4} + y = 1) \]

  \[ \rightarrow \neg (\text{at}_-\ell_{3} \land \text{at}_-m_{3}) \]

  \[ P \vDash \Box \neg (\text{at}_-\ell_{3} \land \text{at}_-m_{3}) \]

6-40