Proof Rules for Invariance and Liveness Properties

Thai Son Hoang

Chair of Information Security, Department of Computer Science
Swiss Federal Institute of Technology Zürich (ETH Zürich)

3rd February 2011, Ascona Meeting

Executions and Traces (of States)

Executions
\[ \alpha = S_0 \xrightarrow{a_0} S_1 \xrightarrow{a_1} S_2 \xrightarrow{a_2} S_3 \xrightarrow{a_3} \ldots \]

Traces
\[ \sigma = S_0, S_1, S_2, S_3, \ldots \]

\[ T(S) \] denotes the set of all traces of system \( S \).

Example

<table>
<thead>
<tr>
<th>system</th>
<th>Counter</th>
<th>events</th>
</tr>
</thead>
<tbody>
<tr>
<td>variables ( c \in \mathbb{Z} )</td>
<td>inc ( \triangleq c \neq 5 \rightarrow c := c + 1 )</td>
<td></td>
</tr>
<tr>
<td>initially ( c = 0 )</td>
<td>dec ( \triangleq c &gt; 3 \rightarrow c := c - 1 )</td>
<td></td>
</tr>
</tbody>
</table>

\( \sigma_{\text{Counter}} : \langle 0 \rangle, \langle 1 \rangle, \langle 2 \rangle, \langle 3 \rangle, \langle 4 \rangle, \langle 5 \rangle, \langle 3 \rangle, \langle 4 \rangle, \langle 5 \rangle, \ldots \)

Outline

1. The Language of Temporal Logic
2. Proof Principles
3. Example. Reader and Writer
4. Conclusions

Discrete Transition Systems (Recall)

Given the following transition system \( S \)

\[ \begin{align*}
\text{system } S & \in \mathcal{T} \\
\text{variables } \mathbf{v} & \in \mathcal{V} \\
\text{initially } \text{init}(\mathbf{v}) & \\
\text{events } \text{evt}_i & \triangleq \text{G}_i(\mathbf{v}) \rightarrow \mathbf{v} := f_i(\mathbf{v})
\end{align*} \]

- \( \mathbf{v} \) denotes the vector of variables \( v_1, \ldots, v_n \).
- \( \text{init}(\mathbf{v}) \) is the initialisation.
- \( \text{G}_i(\mathbf{v}) \) is the guard of event \( \text{evt}_i \).
- \( \text{evt}_i \) is said to be enabled in some state \( s \) if \( \text{G}_i(\mathbf{v}) \) holds in \( s \).

\[ \mathbf{v} : = \tilde{f}_i(\mathbf{v}) \] is the action of event \( \text{evt}_i \).
Temporal formulas to be interpreted over traces.

- A (basic) state formula $Q(\sigma)$ is any first-order logic formula, e.g. $0 \leq c$, $\neg(c = 0) \land c < 2$, $\forall m \cdot m \neq 0 \Rightarrow m \leq c$.
- The basic formulas can be extended by combining the Boolean operators ($\neg$, $\land$, $\lor$, $\Rightarrow$) with temporal operators:
  - $\square$: always
  - $\Diamond$: eventually
  - $\mathcal{U}$: until
- Example of extended formulas:
  - $\square c \in [0, 5]$  
  - $c \leq 6$
  - $\Diamond c = 2$
  - $\square \Diamond c = 2$

Example

$\sigma \models \phi$ means that a trace $\sigma$ satisfies formula $\phi$.

- For state formula $\phi$, $\sigma \models \phi$ if all only if $s_0$ satisfies $\phi$.
- Boolean operators are interpreted in the natural way, e.g.
  $\sigma \models \phi_1 \land \phi_2$ if and only if $\sigma \models \phi_1$ and $\sigma \models \phi_2$.
- Temporal operators are interpreted as follows:
  $\sigma \models \square \phi$ if and only if $\forall k \cdot 0 \leq k < l(\sigma)$, $\sigma^{(k)} \models \phi$
  $\sigma \models \Diamond \phi$ if and only if $\exists k \cdot 0 \leq k < l(\sigma)$, $\sigma^{(k)} \models \phi$
  $\sigma \models \phi_1 \mathcal{U} \phi_2$ if and only if $\exists k \cdot 0 \leq k < l(\sigma)$ such that $\sigma^{(k)} \models \phi_2$ and
  $\forall i \cdot 0 \leq i < k$, $\sigma^{(i)} \models \phi_1$

Length and Suffixes of Traces

Let $\sigma : s_0, s_1, \ldots$ be any non-empty trace.

- The length of $\sigma$ denoted by $l(\sigma)$.
  - Finite trace $\sigma : s_0, \ldots, s_k$: $l(\sigma) = k + 1$.
  - Infinite trace: $l(\sigma) = \infty$.
- For $0 \leq k < l(\sigma)$, $k$-suffix of $\sigma$ is defined as $\sigma^{(k)} = s_k, s_{k+1}, \ldots$.
Safety v.s. Liveness

- Safety properties: *something (bad) will never happen.*
  - Example: invariance properties.
  - Typically expressed by a temporal formula: $\square \phi$ or $\phi_1 \Rightarrow \square \phi_2$.

- Liveness properties: *something (good) will happen.*
  - Example: termination, responsiveness.
  - Typically expressed by a temporal formula:
    $\diamond \phi$ or $\square (\phi_1 \Rightarrow \diamond \phi_2)$.
  - Extended: $\phi_1 U \phi_2$ or $\square (\phi_1 \Rightarrow \diamond \phi_2 U \phi_3)$.

System Properties

- A system S satisfying property $\phi$ if all its traces satisfy $\phi$.
  $S \models \phi$ if and only if $\sigma \models \phi$, for all $\sigma \in T(S)$.

- $S \models \phi$ states that $S \models \phi$ is provable.

Invariance Rules (1/2)

$\vdash \text{inv}() \Rightarrow \phi$

$\vdash S \text{ leads from } \phi \text{ to } \phi$

$\text{INV}_{\text{induct}}$

Counter

$\vdash \square c \in 0..5$

<table>
<thead>
<tr>
<th>System</th>
<th>Counter</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>variables $c \in \mathbb{Z}$</td>
<td>inc $\equiv c \neq 5 \rightarrow c := c + 1$</td>
<td>initially $c = 0$</td>
</tr>
<tr>
<td></td>
<td>dec $\equiv c &gt; 3 \rightarrow c := c - 1$</td>
<td>dec: $c \in 0..5 \wedge c &gt; 3 \Rightarrow c - 1 \in 0..5$</td>
</tr>
</tbody>
</table>

- Initialisation: $\vdash c = 0 \Rightarrow c \in 0..5$
- $\text{inc: } c \in 0..5 \wedge c \neq 5 \Rightarrow c + 1 \in 0..5$
- $\text{dec: } c \in 0..5 \wedge c > 3 \Rightarrow c - 1 \in 0..5$
The Language of Temporal Logic

Proof Principles

Example. Reader and Writer

Conclusions

Invariance Rules (2/2)

\[ \vdash \phi_2 \Rightarrow \phi_1 \]

\[ S \vdash \Box \phi_2 \]

\[ S \vdash \Box \phi_1 \]

INVtheorem

Counter \[ \vdash \Box c \leq 6 \]

Choose \( \phi_2 \) to be \( c \in [0..5] \).

- \( \vdash c \in [0..5] \Rightarrow c \leq 6 \)
- \( \text{Counter} \vdash \Box c \in [0..5] \)

Proof Tools (2 of 3)

Convergence

- Let \( \phi \) be a state formula.
- A trace is said to be convergent when \( \phi \) holds if it does not end with an infinite sequences of states satisfying \( \phi \).
- System \( S \) is said to be convergent when \( \phi \) holds if all its traces are convergent when \( \phi \) holds.
- When the above fact is provable, we denote it as \( \vdash S \downarrow \phi \)

Technique

- For system \( S \) with events \( ev \) \[ \equiv G(v) \rightarrow v \rightleftharpoons \bar{f}(v) \]
- Give a integer variant \( V(v) \)
- \( S \) converges when \( \phi \) holds if for all events \( ev \) of \( S \)
  - \( \phi(v) \land G_i(v) \Rightarrow V(v) \in \mathbb{N} \)
  - \( \phi(v) \land G_i(v) \Rightarrow V(\bar{f}(v)) < V(v) \)

Proof Tools (3 of 3)

Deadlock-freeness

- Let \( \phi \) be a state formula.
- System \( S \) is deadlock-free when \( \phi \) holds if there exists an enabled event of \( S \) when \( \phi \) holds.
- When the above fact is provable, we denote it as \( \vdash S \downarrow \neg \phi \)

This is guaranteed by proving the following.

\[ \phi(v) \Rightarrow G_1(v) \lor \ldots \lor G_n(v) \]

Counter \[ \vdash \Box \Diamond c \geq 2 \]

Choose \( \phi_2 \) to be \( c \in [0..5] \).

- \( \vdash c \in [0..5] \Rightarrow c \leq 6 \)
- \( \text{Counter} \vdash \Box c \in [0..5] \)

Liveness Rules (1/3)

Always Eventually

\[ \vdash S \downarrow \neg \phi \]

\[ \vdash S \text{ is deadlock-free when } \neg \phi \text{ holds} \]

\[ S \vdash \Box \Diamond \phi \]

LIVE\( \Diamond \phi \)

Counter \[ \vdash \Box \Diamond c \geq 2 \]

- Convergence: Using variant \( V \equiv 5 - c \).
  - 5 - c \in \mathbb{N} (using invariant \( c \in [0..5] \))
  - inc: \( -c \geq 2 \land c \neq 5 \Rightarrow 5 - (c + 1) < 5 - c \)
  - dec: \( -c \geq 2 \land c > 3 \Rightarrow 5 - (c - 1) < 5 - c \)
- Deadlock-free: \( -c \geq 2 \Rightarrow c \neq 5 \lor c > 3 \)
Liveness Rules (2/3)

Until

\[ \vdash S \text{ leads from } \phi_1 \land \lnot \phi_2 \text{ to } \phi_1 \lor \phi_2 \]

\[ S \vdash \Box \Diamond (\lnot \phi_1 \lor \phi_2) \]

\[ S \vdash (\phi_1 \Rightarrow \phi_1 \lor \phi_2) \]

LIVE

Counter \[ \vdash \Box (c < 2 \Rightarrow (c < 2 \lor c = 2)) \]

- \textit{Counter} leads from \( c < 2 \land \lnot c = 2 \) to \( c < 2 \lor c = 2 \),
  equivalently \textit{Counter} leads from \( c < 2 \) to \( c < 2 \lor c = 2 \)
  - \textit{inc}: \( c < 2 \land c \neq 5 \Rightarrow c + 1 \leq 2 \)
  - \textit{dec}: \( c < 2 \land c > 3 \Rightarrow c - 1 \leq 2 \)
- Eventually: \( \Box \Diamond (\lnot c < 2 \lor c = 2) \), equivalent to \( \Box \Diamond c \geq 2 \)

Liveness Rules (3/3)

Response

\[ \vdash S \models (\phi_1 \Rightarrow \phi_3) \]

\[ \vdash S \models (\phi_3 \Rightarrow \phi_2 \lor \phi_2) \]

\[ \vdash (\phi_1 \Rightarrow \Diamond \phi_2) \]

LIVE

Counter \[ \vdash \Box (c = 0 \Rightarrow \Diamond c = 2) \]

Choose \( \phi_2 \equiv c < 2 \)

- \( \Box (c = 0 \Rightarrow c < 2) \)
- \( \Box (c < 2 \Rightarrow (c < 2 \lor c = 2)) \)

Example. Reader and Writer

system \( RdWr \) events
variables \( r, w \in \mathbb{Z}, \mathbb{Z} \)
read \( \equiv \ r \neq w \rightarrow r := r + 1 \)
write \( \equiv \ w < r + 3 \rightarrow w := w + 1 \)

An execution
\( (0, 0) \rightarrow (0, 1) \rightarrow (0, 2) \rightarrow (0, 3) \rightarrow (1, 3) \rightarrow (2, 3) \rightarrow (3, 3) \rightarrow (3, 4) \rightarrow (3, 5) \rightarrow (4, 5) \rightarrow (4, 6) \rightarrow (5, 6) \rightarrow (5, 7) \ldots \)

A Progress Properties

Reader’s progress

The Reader will \textbf{eventually read} the data that the Writer wrote.

Formalisation. First attempt

- Can we prove \( RdWr \models \Box \Diamond r = w \)?
  - \textit{No, the Reader might always be behind} the Writer (despite progressing).

Formalisation. Second attempt

\[ RdWr \models (w = K \Rightarrow \Diamond r = K) \]

Counter \[ \vdash \Box (c < 2 \Rightarrow (c < 2 \lor c = 2)) \]

Choose \( \phi_2 \equiv c < 2 \)

- \( \Box (c < 2 \Rightarrow c < 2) \)
- \( \Box (c < 2 \Rightarrow (c < 2 \lor c = 2)) \)

Counter \[ \vdash \Box (c = 0 \Rightarrow \Diamond c = 2) \]

Choose \( \phi_2 \equiv c < 2 \)

- \( \Box (c = 0 \Rightarrow c < 2) \)
- \( \Box (c < 2 \Rightarrow (c < 2 \lor c = 2)) \)
A Proof (1/6)

**system RdWr :**

- **events**
  - read \( \equiv r \neq w \rightarrow r := r + 1 \)
  - write \( \equiv w < r + 3 \rightarrow w := w + 1 \)

- **variables** \( r, w \in \mathbb{Z} \)
- **initially** \( r = 0 \land w = 0 \)

\[ \vdash □(w = K \Rightarrow \Diamond r = K) \]

\[ \begin{align*}
\text{LIVE}_{\text{response}} & \quad \forall \phi \phi_3 \Rightarrow (\phi_3 \U r = K) \\
\text{RdWr} \vdash □(w = K \Rightarrow \Diamond r = K) \\
\text{RdWr} \vdash □(\phi_3 \Rightarrow (\phi_3 \U r = K))
\end{align*} \tag{1} \]

\[ S \vdash □(\phi_1 \Rightarrow \phi_3) \]

\[ S \vdash □(\phi_2 \Rightarrow (\phi_3 \U r = K)) \]

\[ S \vdash □(\phi_1 \Rightarrow \phi_3 \U r = K) \]

\[ \text{LIVE}_{\text{response}} \]

A Proof (2/6)

**system RdWr :**

- **events**
  - read \( \equiv r \neq w \rightarrow r := r + 1 \)
  - write \( \equiv w < r + 3 \rightarrow w := w + 1 \)

- **variables** \( r, w \in \mathbb{Z} \)
- **initially** \( r = 0 \land w = 0 \)

\[ \vdash □(w = K \Rightarrow r \leq K) \]

\[ \begin{align*}
\text{INV}_{\text{induct}} & \quad \forall \phi \phi_2 \Rightarrow \phi_1 \\
S \vdash □ \phi_2 \\
S \vdash □ \phi_1
\end{align*} \]

\[ \begin{align*}
\text{INV}_{\text{theoem}} & \quad \forall \phi \phi_2 \Rightarrow \phi_1 \\
\vdash □(w = K \Rightarrow r \leq K)
\end{align*} \]

- **INV_{induct}** fails, hence apply **INV_{theoem}** with \( \phi_2 \) to be \( r \leq w \).

A Proof (3/6)

**system RdWr :**

- **events**
  - read \( \equiv r \neq w \rightarrow r := r + 1 \)
  - write \( \equiv w < r + 3 \rightarrow w := w + 1 \)

- **variables** \( r, w \in \mathbb{Z} \)
- **initially** \( r = 0 \land w = 0 \)

\[ \vdash \phi \equiv \phi_1 \land \lnot \phi_2 \rightarrow \phi_1 \lor \phi_2 \]

\[ S \vdash □(\lnot \phi_1 \lor \phi_2) \]

\[ S \vdash □(\phi_1 \Rightarrow \phi_1 \lor \phi_2) \]

\[ \text{LIVE}_{\text{induction}} \]

A Proof (4/6)

**system RdWr :**

- **events**
  - read \( \equiv r \neq w \rightarrow r := r + 1 \)
  - write \( \equiv w < r + 3 \rightarrow w := w + 1 \)

- **variables** \( r, w \in \mathbb{Z} \)
- **initially** \( r = 0 \land w = 0 \)

\[ \vdash □(r \leq K \Rightarrow r \leq K \lor w = K) \]

\[ \text{RdWr} \vdash □(r \leq K \Rightarrow w = K) \]

\[ \text{logic} \]

\[ \vdash □(r \leq K \Rightarrow r \leq K \lor w = K) \]

\[ \vdash □(r \leq K \Rightarrow r \leq K) \]

\[ \vdash □(r \leq K \Rightarrow w = K) \]

\[ \vdash □(r < K \land w = K + 1 \Rightarrow r \leq K) \]

\[ \vdash □(r < K \land w = K + 1 \Rightarrow r \leq K) \]

\[ \vdash □(r < K \land w = K + 1 \Rightarrow r \leq K) \]

\[ \vdash □(r < K \land w = K + 1 \Rightarrow r \leq K) \]
The Language of Temporal Logic

Proof Principles

Example. Reader and Writer

Conclusions

A Proof (5/6)

\[ \vdash S \Downarrow \neg \phi \]

\[ \vdash S \text{ is deadlock-free when } \neg \phi \text{ holds} \]

\[ S \vdash \Box \Diamond \phi \]

\[ (2.2) \]

 Logic

\[ \vdash \Box \phi (\neg \rho \leq K \lor r = K) \]

\[ \vdash \Box \rho \geq K \]

\[ \vdash \Box \phi \text{ is deadlock-free when } \neg \rho \geq K \text{ holds} \]

\[ (2.2.1) \]

 Use variant:

\[ (K - r) \times 2 + (r + 3 - w) \]

\[ (2.2.2) \]

 Definition

\[ \vdash \Box \phi \text{ is deadlock-free when } \neg \rho \geq K \text{ holds} \]

\[ \vdash \Box \phi \text{ is deadlock-free when } \neg \rho \geq K \text{ holds} \]

\[ \neg \rho \geq K \Rightarrow \rho \neq w \lor w < 3 + r \]

Further Directions

Proof rules for certain classes of invariance and liveness properties.

The proof rules based on the reasoning about:

- the system leads from \( \phi_1 \) to \( \phi_2 \)
- the system is convergence when \( \phi \) holds
- the system is deadlock-free when \( \phi \) holds.

Proofs become tedious when the system becomes large.

Refinement helps to reduce the complexity.

- Invariance properties are maintained.
- How about liveness?

Concurrent systems: fairness assumptions.

- Expect some weaker rules.
- Interaction with refinement?
Zohar Manna and Amir Pnueli.
Adequate Proof Principles for Invariance and Liveness Properties of Concurrent Programs. 

Zohar Manna and Amir Pnueli.
Completing the Temporal Picture. 