Verification

Lecture 25

Bernd Finkbeiner
Peter Faymonville
Michael Gerke

UNIVERSITÄT
DES
SAARLANDES

Invariant generation: forward propagation

let ForwardPropagate F_{pre} $\mathcal{L}$ =
  $S := \{L_0\}$;
  $\mu(L_0) := F_{\text{pre}}$;
  $\mu(L) := \bot$ for $L \in \mathcal{L} \setminus \{L_0\}$;
  while $S \neq \emptyset$ do
    let L_j = choose S in
    $S := S \setminus \{L_j\}$;
    foreach $L_k \in \text{succ}(L_j)$ do [$L_k \in \text{succ}(L_j)$ is a **successor** of L_j if there is a basic path from L_j to L_k]
      let F = $\text{sp}(\mu(L_j),\ S_j; \ldots; S_k)$ in
      if $F \not\Rightarrow \mu(L_k)$
      then $\mu(L_k) := \mu(L_k) \lor F$;
          $S := S \cup \{L_k\}$;
    done;
  done;
  μ

Problem: algorithm may not terminate
Solution: Abstraction

A state s is reachable for program P if it appears in some computation of P.

The problem is that ForwardPropagate computes the exact set of reachable states.

Inductive annotations usually over-approximate the set of reachable states: every reachable state s satisfies the annotation, but other unreachable states can also satisfy the annotation.

Abstract interpretation cleverly over-approximate the reachable state set to guarantee termination.

Abstract interpretation is constructed in 6 steps.

Step 1: Choose an abstract domain D.

The abstract domain D is a syntactic class of Σ-formulae of some theory T.

- interval abstract domain D_{I} consists of conjunctions of $\Sigma_{\mathbb{Q}}$-literals of the forms

  $$c \leq v \quad \text{and} \quad v \leq c \,,$$

  for constant c and program variable v.
  Useful representation: intervals $[l, u]$ with interval arithmetic.
- Karr's abstract domain D_{K} consist of conjunctions of $\Sigma_{\mathbb{Q}}$-literals of the form

  $$c_0 + c_1 x_1 + \cdots + c_n x_n = 0 \,,$$

  for constants $c_0, c_1, \ldots, c_n$ and variables $x_1, \ldots, x_n$.

Step 2: Construct a map from FOL formulae to D.

Define

$$\nu_D : \text{FOL} \rightarrow D$$

to map a FOL formula F to element $\nu_D(F)$ of D, with the property that for any F,

$$F \;\Rightarrow\; \nu_D(F)\;.$$

Example:

$@L_0 : i = 0 \;\wedge\; n \geq 0;$
while
$@L_1 : ?$
$(i < n)\ \{$
$i := i + 1;$
$\}$

Abstraction of $F:\; i = 0 \;\wedge\; n \geq 0$ at L_0 in the interval abstract domain:

$$\nu_{D_I}(F):\; 0 \leq i \;\wedge\; i \leq 0 \;\wedge\; 0 \leq n$$

Step 3: Define an abstract sp.

Define an abstract strongest postcondition $\overline{\mathrm{sp}}_D$ for assumption and assignment statements such that

$$\mathrm{sp}(F, S) \Rightarrow \overline{\mathrm{sp}}_D(F, S) \quad \text{and} \quad \overline{\mathrm{sp}}_D(F, S) \in D$$

for statement S and $F \in D$.

- statement `assume` c:

  $$\mathrm{sp}(F, \texttt{assume}\ c) \Leftrightarrow c \land F\,.$$

  Define abstract conjunction $\sqcap_D$, such that

  $$F_1 \land F_2 \Rightarrow F_1 \sqcap_D F_2 \quad \text{and} \quad F_1 \sqcap_D F_2 \in D$$

  for $F_1, F_2 \in D$. Then if $F \in D$,

  $$\overline{\mathrm{sp}}_D(F, \texttt{assume}\ c) \Leftrightarrow \nu_D(c) \sqcap_D F\,.$$

  If the abstract domain D consists of conjunctions of literals, $\sqcap_D$ is just $\land$. For example, in the interval domain,

  $$\overline{\mathrm{sp}}_{D_I}(F, \texttt{assume}\ c) \Leftrightarrow \nu_{D_I}(c) \land F\,.$$

- assignment statements:

$$\mathrm{sp}(F[v],\ v := e[v]) \Leftrightarrow \exists v^0.\ v = e[v^0] \wedge F[v^0],$$

Avoid quantification whenever possible. For example, in the interval domain, use the interval evaluation $[l, u]$ of $e[v]$ to define

$$\mathrm{sp}(F[v],\ v := e[v]) \Leftrightarrow l \leq v \wedge v \leq u \wedge G$$

where G is the conjunction of literals in F except those referring to v.

Step 4: Define abstract disjunction.

Disjunction is applied in ForwardPropagate

$$\mu(L_k) := F \vee \mu(L_k)$$

Define abstract disjunction $\sqcup_D$ for this purpose, such that

$$F_1 \vee F_2 \Rightarrow F_1 \sqcup_D F_2 \quad \text{and} \quad F_1 \sqcup_D F_2 \in D$$

for $F_1, F_2 \in D$.
In the interval domain, use interval hull:

$$[l_1, u_1] \sqcup [l_2, u_2] = [\min(l_1, l_2), \max(u1, u2)]$$

Step 5: Define abstract implication checking.

On each iteration of the inner loop of ForwardPropagate, validity of the implication

$$F \Rightarrow \mu(L_k)$$

is checked to determine whether $\mu(L_k)$ has changed. A proper selection of D ensures that this validity check is decidable.

In the interval domain,
let F assert that $x_i \in [l_i, u_i]$ and G assert that $x_i \in [m_i, n_i]$, then

$$F \Rightarrow G \qquad \text{iff} \qquad m_i \leq l_i \wedge u_i \leq n_i \text{ for all } i$$

Step 6: Define widening.

Defining an abstraction is not sufficient to guarantee termination in general. Thus, abstractions that do not guarantee termination are equipped with a widening operator $\triangledown_D$.

A widening operator $\triangledown_D$ is a binary function

$$\triangledown_D : D \times D \to D$$

such that

$$F_1 \vee F_2 \;\Rightarrow\; F_1 \triangledown_D F_2$$

for $F_1, F_2 \in D$. It obeys the following property. Let $F_1, F_2, F_3, \ldots$ be an infinite sequence of elements $F_i \in D$ such that for each i,

$$F_i \;\Rightarrow\; F_{i+1} \,.$$

Define the sequence

$$G_1 = F_1 \quad \text{and} \quad G_{i+1} = G_i \triangledown_D F_{i+1} \,.$$

For some i^* and for all $i \geq i^*$,

$$G_i \;\Leftrightarrow\; G_{i+1} \,.$$

That is, the sequence G_i converges even if the sequence F_i does not converge. A proper strategy of applying widening guarantees that the forward propagation procedure terminates.

Interval analysis does not naturally terminate

Example:

$@L_0 : i = 0 \ \wedge \ n \geq 0;$

`while`

$@L_1 : ?$

$(i < n)$ {

$i := i + 1;$

}

Widening:

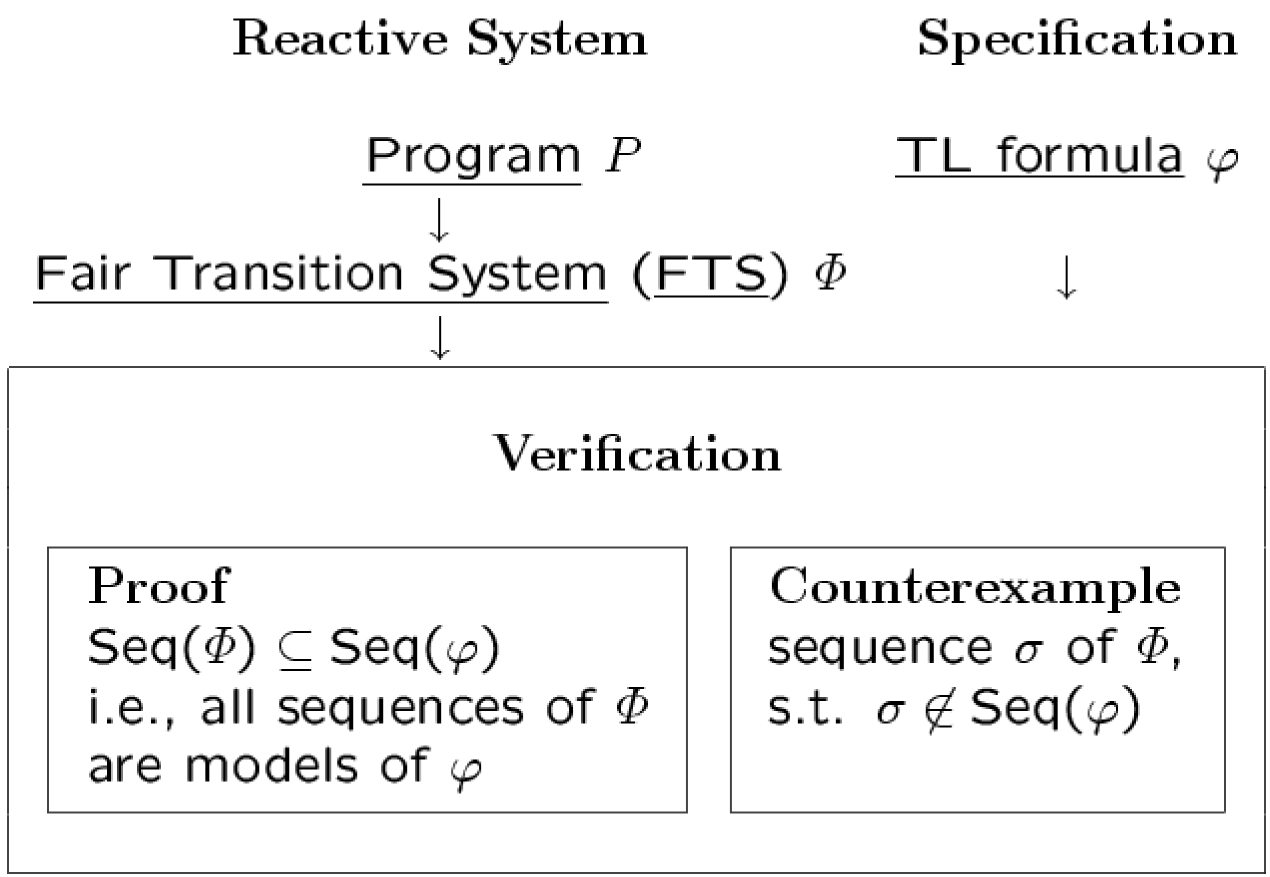
Suppose F asserts $x \in [l_1, u_1]$ and G asserts that $x \in [l_2, u_2]$, then $F \bigtriangledown_{D_I} G$ asserts $x \in [l, u]$ where

- $l = -\infty$ if $l_2 < l_1$, otherwise $l = l_1$
- $u = \infty$ if $u_2 > u_1$, otherwise $u = u_1$.

```
let AbstractForwardPropagate P F_pre 𝓛 =
  S := {L_0};
  μ(L_0) := ν_D(F_pre);
  μ(L) := ⊥ for L ∈ 𝓛 ∖ {L_0};
  while S ≠ ∅ do
    let L_j = choose S in
    S := S ∖ {L_j};
    foreach L_k ∈ succ(L_j) do
      let F = sp‾_D(μ(L_j), S_j; ... ; S_k) in
      if F ⇏ μ(L_k)
      then if Widen()
             then μ(L_k) := μ(L_k) ∇_D (μ(L_k) ⊔_D F);
             else μ(L_k) := μ(L_k) ⊔_D F;
             S := S ∪ {L_k};
    done;
  done;
  μ
```

Deductive Verification of Reactive Systems

Deductive verification of reactive systems



Symbolic Transition Systems

- A (finite) set of variables $V \subseteq \mathcal{V}$
  System variables: data variables + control variables

- Initial condition θ
  first-order assertion over V
  that characterizes all initial states

- A (finite) set of transitions $\mathcal{T}$

For each $\tau \in$: τ: $\Sigma \mapsto 2^{\Sigma}$

τ is represented by the transition relation $\rho(\tau)$
(next-state relation)

Enabled/Disabled/Taken Transitions

- A transition τ
  - is enabled on s if $\tau(s) \neq \{\}$
  - is disabled on s if $\tau(s) = \{\}$
- For an infinite sequence of states

  $\sigma: s_0, s_1, s_2, \ldots$

  a transition τ
  - is enabled at position k if it is enabled on s_k
  - is taken at position k if s_{k+1} is a τ-successor of s_k

Fair Transition Systems

$$\Phi = (V, \theta, \mathcal{T}, \mathcal{J}, \mathcal{C})$$

- $\mathcal{J} \subseteq \mathcal{T}$: set of just (weakly fair) transitions
- $\mathcal{C} \subseteq \mathcal{T}$: set of compassionate (strongly fair) transitions

- Justice: for each just transition it is not the case that the transition is continually enabled but only taken at finitely many positions.
- Compassion: for each compassionate transition it is not the case that the transition is enabled at infinitely many positions but only taken at finitely many positions.

Example

- V : {x,y: integer}
- θ : $x=0 \wedge y=0$
- $\mathcal{T}$: $\{\tau_I, \tau_x, \tau_y\}$
- $\mathcal{J}$: $\{\tau_x\}$
- $\mathcal{C}$: $\{\tau_y\}$
- $\rho(\tau_x)$: $x' = x+1 \bmod 2$
- $\rho(\tau_y)$: $x=1 \wedge y' = y+1$

- s_0=<x=0, y=0>
  (satisfies the initial condition)
- s_1=<x=1, y=0>
  (τ_x taken)
- s_2=<x=0, y=0>
  (τ_x taken)
- s_3=<x=1, y=0>
  (τ_x taken)
- …

Justice: YES

Compassion: NO (τ_y is infinitely often enabled but never taken.)

Computations

An infinite sequence of states

$$\sigma: s_0, s_1, s_2, \ldots$$

is a computation of a fair transition system, if it satisfies:

- Initiality
- Consecution
- Justice
- Compassion

Fairness = Justice + Compassion

Computation = Run + Fairness

Inductive Assertions

For assertion q,

$$\begin{array}{ll} \text{B1.} & P \Vdash \Theta \rightarrow q \\ \text{B2.} & P \Vdash \{q\}\ \mathcal{T}\ \{q\} \\ \hline & P \vDash \Box q \end{array} \quad \text{B-INV}$$

- 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

$$\begin{array}{l} \textbf{local } x\text{: } \textbf{integer} \textbf{ where } x = 1 \\ \left[\begin{array}{ll} \ell_0: & \textbf{request } x \\ \ell_1: & \textbf{critical} \\ \ell_2: & \textbf{release } x \\ \ell_3: & \end{array}\right] \end{array}$$

B1: $\underbrace{x = 1 \wedge \pi = \{\ell_0\}}_{\Theta} \rightarrow \underbrace{at\_\ell_1 \rightarrow x = 0}_{q}$

holds since $\pi = \{\ell_0\} \rightarrow at\_\ell_1 = \mathrm{F}$

B2: $\{q\}\tau_{\ell_0}\{q\}$

$$\underbrace{at\_\ell_1 \rightarrow x = 0}_{q} \wedge \underbrace{move(\ell_0, \ell_1) \wedge x > 0 \wedge x' = x - 1}_{\rho_{\tau_{\ell_0}}}$$

$$\rightarrow \underbrace{at'\_\ell_1 \rightarrow x' = 0}_{q'}$$

we have $move(\ell_0, \ell_1) \rightarrow at'\_\ell_1 = \mathrm{T}$

BUT

$(at\_\ell_1 \rightarrow x = 0) \wedge x > 0 \wedge x' = x - 1 \rightarrow x' = 0$

Cannot prove: not state-valid

Rules for Strengthening

For assertions q_1, q_2,

$$\frac{P \models \Box q_1 \qquad P \Vdash q_1 \rightarrow q_2}{P \models \Box q_2}$$

MON-I

For assertions q, φ

$$\frac{\begin{array}{ll} \text{I1.} & P \Vdash \varphi \rightarrow q \\ \text{I2.} & P \Vdash \Theta \rightarrow \varphi \\ \text{I3.} & P \Vdash \{\varphi\}\ \mathcal{T}\ \{\varphi\} \end{array}}{P \models \Box q}$$

INV

Example

$\textbf{local} \quad x\colon \quad \textbf{integer} \quad \textbf{where } x = 1$

$$\left[\begin{array}{ll} \ell_0: & \textbf{request } x \\ \ell_1: & \textbf{critical} \\ \ell_2: & \textbf{release } x \\ \ell_3: & \end{array}\right]$$

$$\boxed{P \models \Box \underbrace{(at\_\ell_1 \rightarrow x = 0)}_{q}}$$

inductive assertion that implies

$$q: \; at\_\ell_1 \rightarrow x = 0$$

is

$$\varphi: \; (at\_\ell_1 \rightarrow x = 0) \land (at\_\ell_0 \rightarrow x = 1)$$

Example (cont'd)

$$\text{local } x\text{: integer where } x = 1$$

$$\left[\begin{array}{ll} \ell_0: & \text{request } x \\ \ell_1: & \text{critical} \\ \ell_2: & \text{release } x \\ \ell_3: & \end{array}\right]$$

Consider $\{\varphi\}\ \tau_{\ell_0}\ \{\varphi\}$:

$$\underbrace{(at\_\ell_0 \rightarrow x = 1) \ \wedge \ (at\_\ell_1 \rightarrow x = 0)}_{\varphi} \ \wedge$$

$$\underbrace{move(\ell_0, \ell_1) \wedge x > 0 \wedge x' = x - 1}_{\rho_{\tau_{\ell_0}}}$$

$$\rightarrow \ \underbrace{(at'\_\ell_0 \rightarrow x' = 1) \ \wedge \ (at'\_\ell_1 \rightarrow x' = 0)}_{\varphi'}$$

$move(\ell_0, \ell_1)$ implies $\ell_0 \in \pi, \ell_0 \notin \pi', \ell_1 \in \pi'$

Therefore

$$(\mathrm{T} \rightarrow x = 1) \wedge \ldots \wedge x' = x - 1 \wedge x > 0$$

$$\rightarrow \ (\mathrm{F} \rightarrow \ldots) \wedge (\mathrm{T} \rightarrow x' = 0)$$

holds.

Example (cont‘d)

Consider $\{\varphi\}\ \tau_{\ell_2}\ \{\varphi\}$:

$$\underbrace{(at\_\ell_0 \rightarrow x = 1)\ \wedge\ (at\_\ell_1 \rightarrow x = 0)}_{\varphi}\ \wedge$$

$$\underbrace{move(\ell_2, \ell_3) \wedge x > 0 \wedge x' = x - 1}_{\rho_{\tau_{\ell_2}}}$$

$$\rightarrow\ \underbrace{(at'\_\ell_0 \rightarrow x' = 1)\ \wedge\ (at'\_\ell_1 \rightarrow x' = 0)}_{\varphi'}$$

$move(\ell_2, \ell_3)$ implies $\ell_2 \in \pi, \ell_2 \notin \pi', \ell_3 \in \pi'$
and by CONFLICT invariants $\ell_0, \ell_1 \notin \pi'$.

local x: integer where $x = 1$		
ℓ_0 :	request x	
ℓ_1 :	critical	
ℓ_2 :	release x	
ℓ_3 :		

Therefore

$$\ldots \wedge \ldots\ \rightarrow\ (\mathrm{F} \rightarrow x' = 1) \wedge (\mathrm{F} \rightarrow x' = 0)$$

holds.

Example: Peterson's Mutex-Algorithm

$$\text{local} \quad y_1, y_2 : \text{boolean} \quad \text{where } y_1 = \text{F}, y_2 = \text{F}$$
$$s : \text{integer} \quad \text{where } s = 1$$

$$P_1 :: \quad \ell_0 : \text{loop forever do} \begin{bmatrix} \ell_1 : & \text{noncritical} \\ \ell_2 : & (y_1, s) := (\text{T}, 1) \\ \ell_3 : & \text{await } (\neg y_2) \vee (s = 2) \\ \ell_4 : & \text{critical} \\ \ell_5 : & y_1 := \text{F} \end{bmatrix}$$

$$||$$

$$P_2 :: \quad m_0 : \text{loop forever do} \begin{bmatrix} m_1 : & \text{noncritical} \\ m_2 : & (y_2, s) := (\text{T}, 2) \\ m_3 : & \text{await } (\neg y_1) \vee (s = 1) \\ m_4 : & \text{critical} \\ m_5 : & y_2 := \text{F} \end{bmatrix}$$

Goal:

Mutual exclusion for Peterson‘s algorithm:

$$\Box \underbrace{\neg(at\_\ell_4 \ \wedge \ at\_m_4)}_{\psi}$$

Bottom-up invariants:

φ_0: $s = 1 \ \vee \ s = 2$

φ_1: $y_1 \ \leftrightarrow \ at\_\ell_{3..5}$

φ_2: $y_2 \ \leftrightarrow \ at\_m_{3..5}$

Example (cont‘d)

$$\begin{array}{llll} \textbf{local} & y_1, y_2: & \textbf{boolean} & \textbf{where } y_1 = \text{F}, y_2 = \text{F} \\ & s \quad : & \textbf{integer} & \textbf{where } s = 1 \end{array}$$

$$\ell_0: \quad \textbf{loop forever do}$$

$$P_1 :: \quad \left[\begin{array}{ll} \ell_1: & \textbf{noncritical} \\ \ell_2: & (y_1,\ s) := (\text{T},\ 1) \\ \ell_3: & \textbf{await } (\neg y_2) \vee (s = 2) \\ \ell_4: & \textbf{critical} \\ \ell_5: & y_1 := \text{F} \end{array}\right]$$

$$||$$

$$m_0: \quad \textbf{loop forever do}$$

$$P_2 :: \quad \left[\begin{array}{ll} m_1: & \textbf{noncritical} \\ m_2: & (y_2,\ s) := (\text{T},\ 2) \\ m_3: & \textbf{await } (\neg y_1) \vee (s = 1) \\ m_4: & \textbf{critical} \\ m_5: & y_2 := \text{F} \end{array}\right]$$

$$\Box \underbrace{\neg(at\_\ell_4 \ \wedge \ at\_m_4)}_{\psi}$$

Problem:

The verification conditions

$$\{\varphi_0 \ \wedge \ \varphi_1 \ \wedge \ \varphi_2 \ \wedge \ \psi\} \ \ell_3 \ \{\psi\}$$

$$\{\varphi_0 \ \wedge \ \varphi_1 \ \wedge \ \varphi_2 \ \wedge \ \psi\} \ \ m_3 \ \{\psi\}$$

are not state-valid.

Example (cont'd)

$$pre(\tau_{\ell_3}, \psi)\colon\ \forall \pi'\colon\ \underbrace{move(\ell_3, \ell_4)\ \wedge\ (\neg y_2\ \vee\ s \neq 1)}_{\rho_{\ell_3}}\ \rightarrow$$

$$\underbrace{\neg(at'\_\ell_4\ \wedge\ at'\_m_4)}_{\psi'}$$

$pre(\tau_{\ell_3}, \psi)$ simplifies to:

$$at\_\ell_3\ \wedge\ (\neg y_2\ \vee\ s \neq 1)\ \rightarrow\ \neg at\_m_4$$

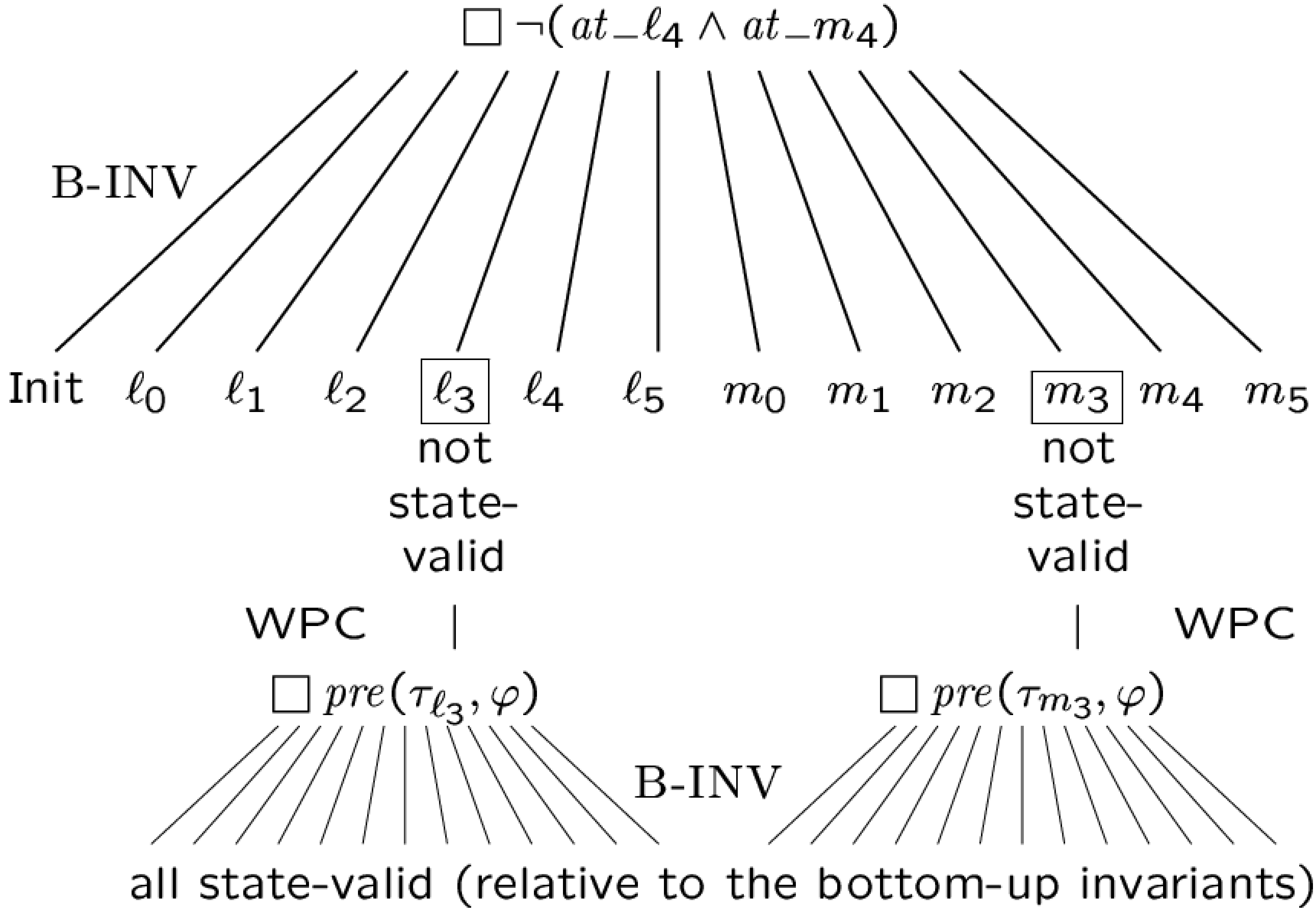
$$\boxed{\varphi_3\colon\ at\_\ell_3\ \wedge\ at\_m_4\ \rightarrow\ y_2\ \wedge\ s = 1}$$

$pre(\tau_{m_3}, \psi)$: $\forall \pi' \ldots\ldots$

simplifies to:

$$\boxed{\varphi_4\colon\ at\_\ell_4\ \wedge\ at\_m_3\ \rightarrow\ y_1\ \wedge\ s = 2}$$

Example (cont'd)



Precedence Properties

are of the form

$$p \;\Rightarrow\; q_m \;\mathcal{W}\; (q_{m-1} \;\cdots\; (q_1 \;\mathcal{W}\; q_0) \ldots)$$

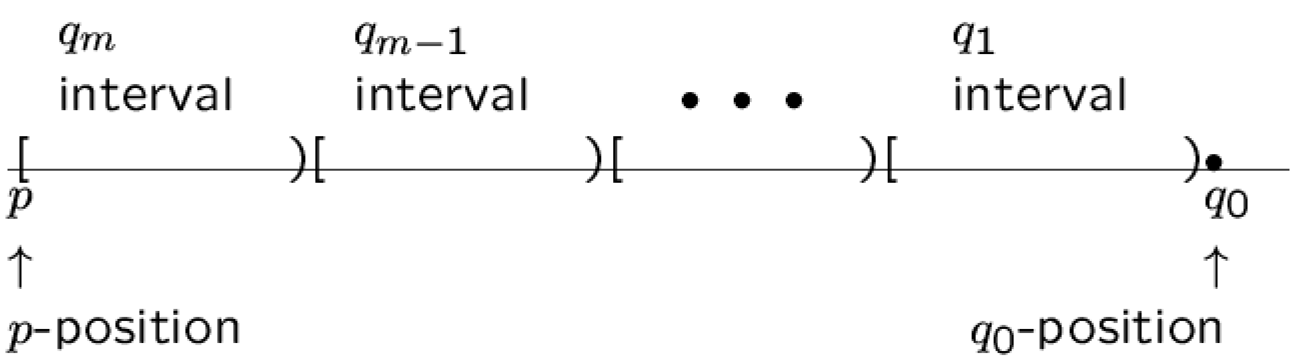
also written

$$\boxed{p \;\Rightarrow\; q_m \;\mathcal{W}\; q_{m-1} \;\cdots\; q_1 \;\mathcal{W}\; q_0}$$

for assertions $p, q_0, q_1, \ldots, q_m$.

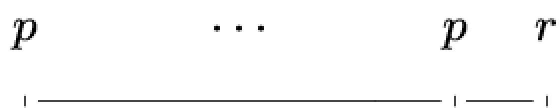
Each interval may be empty, may extend to infinity.

Models that satisfy these formulas



Simple Precedence

$$\boxed{p \;\Rightarrow\; p \,\mathcal{W}\, r}$$



can be reduced to first-order VCs by verification rule WAIT-B:

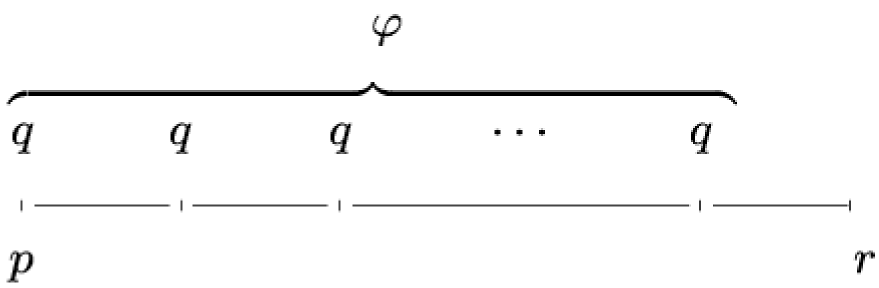
Rule WAIT-B (basic waiting-for)

For assertions p, r,

$$\frac{P \;\Vdash\; \{p\}\; \mathcal{T}\; \{p \vee r\}}{P \;\vDash\; p \Rightarrow p \,\mathcal{W}\, r}$$

General Waiting-For

$$\boxed{p \ \Rightarrow \ q \, \mathcal{W} \, r}$$



Rule WAIT (general waiting-for)

For assertions p, q, r, φ

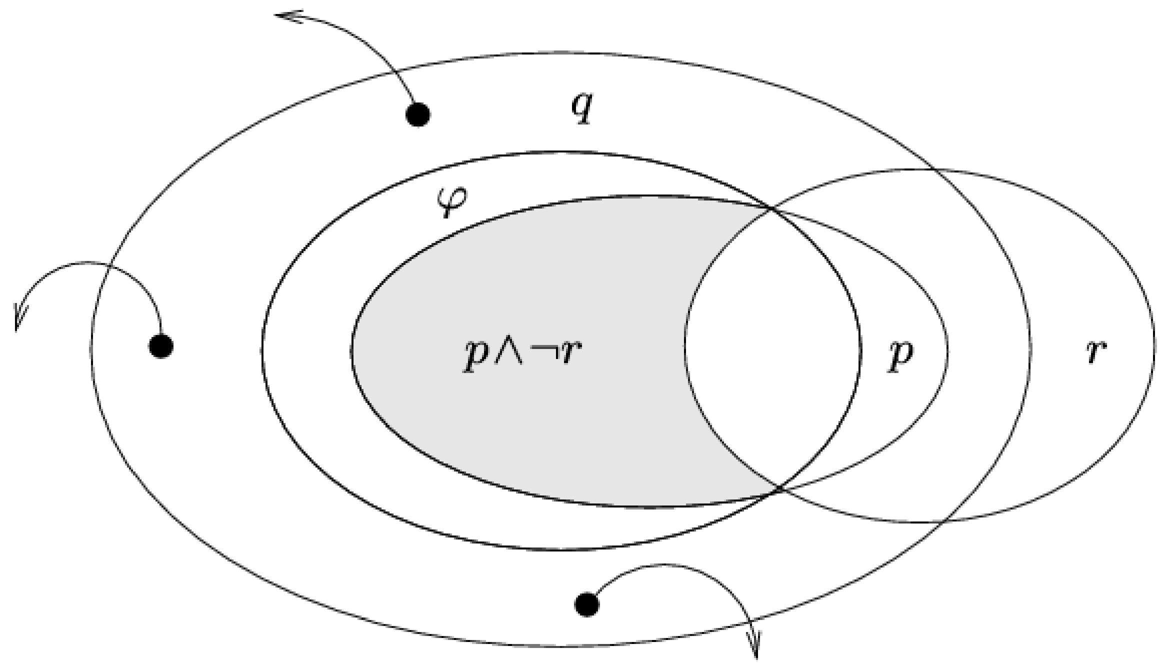
W1. $p \ \rightarrow \ \varphi \vee r$

W2. $\varphi \ \rightarrow \ q$

W3. $\{\varphi\}\mathcal{T}\{\varphi \vee r\}$

$$p \ \Rightarrow \ q \, \mathcal{W} \, r$$

Strengthening & Weakening



$\varphi \rightarrow q$ "φ strengthens q"

$p \rightarrow \varphi \vee r$, i.e., $p \wedge \neg r \rightarrow \varphi$ "φ weakens $p \wedge \neg r$"

Example

$$
\begin{array}{llll}
\textbf{local} & y_1, y_2: & \textbf{boolean} & \textbf{where } y_1 = \text{F}, y_2 = \text{F} \\
 & s \quad : & \textbf{integer} & \textbf{where } s = 1
\end{array}
$$

$$
P_1 :: \quad \ell_0 : \ \textbf{loop forever do} \begin{bmatrix} \ell_1 : & \textbf{noncritical} \\ \ell_2 : & (y_1,\ s) := (\text{T},\ 1) \\ \ell_3 : & \textbf{await } (\neg y_2) \vee (s = 2) \\ \ell_4 : & \textbf{critical} \\ \ell_5 : & y_1 := \text{F} \end{bmatrix}
$$

$$||$$

$$
P_2 :: \quad m_0 : \ \textbf{loop forever do} \begin{bmatrix} m_1 : & \textbf{noncritical} \\ m_2 : & (y_2,\ s) := (\text{T},\ 2) \\ m_3 : & \textbf{await } (\neg y_1) \vee (s = 1) \\ m_4 : & \textbf{critical} \\ m_5 : & y_2 := \text{F} \end{bmatrix}
$$

We proved mutual exclusion

$$\psi_1: \quad \neg(at\_\ell_4 \ \wedge \ at\_m_4)$$

Using invariants

$$\varphi_0: \quad s = 1 \ \vee \ s = 2$$

$$\varphi_1: \quad y_1 \ \leftrightarrow \ at\_\ell_{3..5}$$

$$\varphi_2: \quad y_2 \ \leftrightarrow \ at\_m_{3..5}$$

$$\varphi_3: \quad at\_\ell_3 \ \wedge \ at\_m_4 \ \rightarrow \ y_2 \ \wedge \ s = 1$$

$$\varphi_4: \quad at\_\ell_4 \ \wedge \ at\_m_3 \ \rightarrow \ y_1 \ \wedge \ s = 2$$

$$\psi_2: \quad \underbrace{at\_\ell_3 \ \wedge \ at\_m_{0..2}}_{p} \ \Rightarrow \ \underbrace{\neg at\_m_4}_{q} \ \mathcal{W} \ \underbrace{at\_\ell_4}_{r}$$

Proof Attempt

$$\varphi = p \wedge \neg r : at\_\ell_3 \wedge at\_m_{0..2}$$

W1, W2 hold.
For W3:

$$\{\underbrace{at\_\ell_3 \wedge at\_m_{0..2}}_{p}\}\mathcal{T}\{\underbrace{(at\_\ell_3 \wedge at\_m_{0..2})}_{p} \vee \underbrace{at\_\ell_4}_{r}\}$$

we only need to consider the enabled transitions:

ℓ_3: establishes at_ℓ_4

m_0: leads to m_1

m_1: leads to m_2

m_2: ... does not lead to $(at\_\ell_3 \wedge at\_m_{0..2}) \vee at\_\ell_4$

$$\rho_{m_2} \wedge \underbrace{at\_\ell_3 \wedge at\_m_{0..2}}_{p} \rightarrow \underbrace{at'\_\ell_3 \wedge at'\_m_{0..2}}_{p'} \vee \underbrace{at'\_\ell_4}_{r'}$$

FAILS

(ρ_{m_2} neither preserves p nor achieves r')

Weakening & Strengthening

Let

$$\boxed{\varphi : at\_\ell_3 \wedge at\_m_{0..3}}$$

We cannot weaken φ to include at_m_4 because of premise

W2: $\varphi \;\rightarrow\; \neg \underbrace{at\_m_4}_{q}$

We weakened φ too much;
so we have to strengthen it back.

Let

$$\boxed{\varphi : at\_\ell_3 \wedge (at\_m_{0..2} \vee (at\_m_3 \wedge s = 2))}$$

Check:

ℓ_3: OK
m_0: OK
m_1: OK
m_2: OK

But m_3 leads to m_4.

Example (cont'd)

W1: $$\underbrace{at\_\ell_3 \;\wedge\; at\_m_{0..2}}_{p} \;\rightarrow$$
$$\underbrace{at\_\ell_3 \;\wedge\; (at\_m_{0..2} \;\vee\; \cdots)}_{\varphi} \;\vee\; \underbrace{\cdots}_{r}$$

W2: $$\underbrace{\cdots \;\wedge\; (at\_m_{0..2} \;\vee\; (at\_m_3 \;\wedge\; \cdots))}_{\varphi} \;\rightarrow\; \underbrace{\neg at\_m_4}_{q}$$

W3: $$\rho_\tau \wedge \underbrace{at\_\ell_3 \;\wedge\; (at\_m_{0..2} \;\vee\; (at\_m_3 \;\wedge\; s = 2))}_{\varphi} \;\rightarrow$$

$$\underbrace{at'\_\ell_3 \;\wedge\; (at'\_m_{0..2} \;\vee\; (at'\_m_3 \;\wedge\; s' = 2))}_{\varphi'} \;\vee\; \underbrace{at'\_\ell_4}_{r'}$$

Check:

ℓ_3, m_2: OK

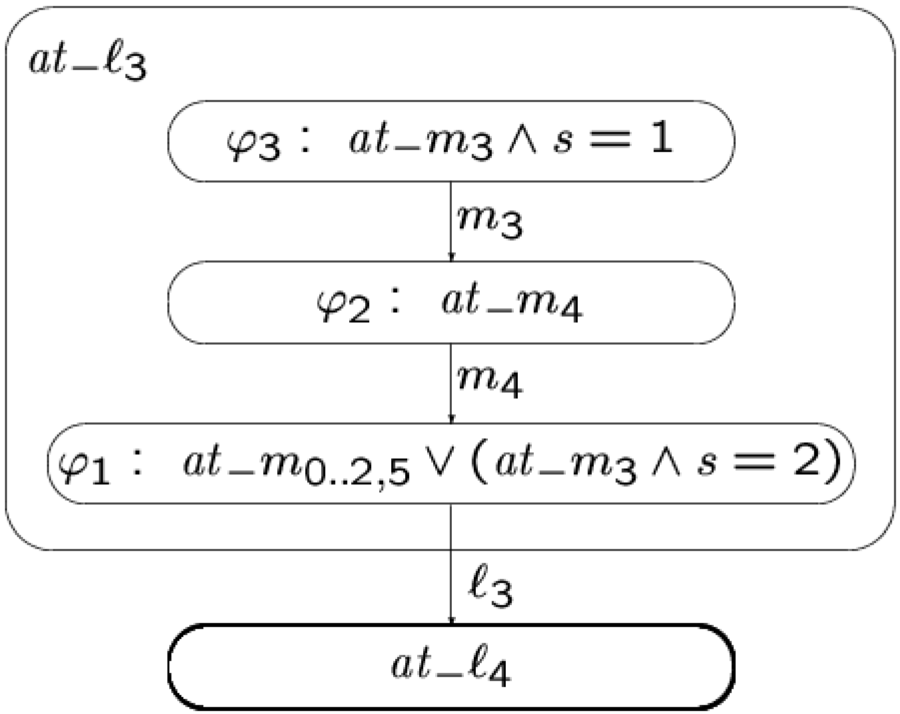
m_3: disabled (with the help of the invariant

$at\_\ell_{3..5} \leftrightarrow y_1$, we have $y_1 = \mathrm{T}$).

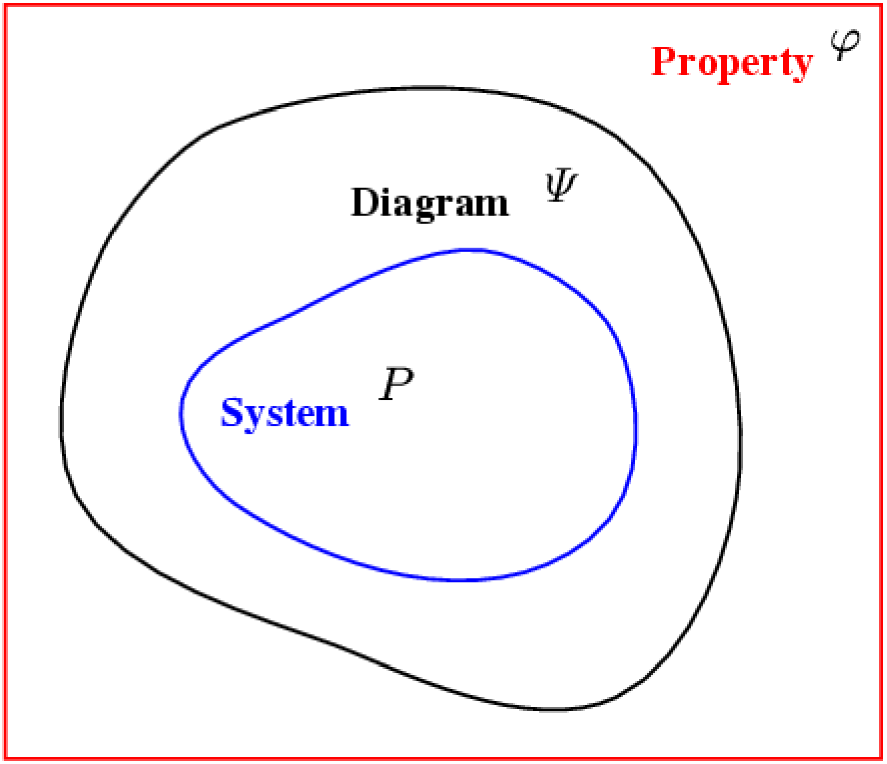
Verification Diagrams

Verification diagrams allow a graphical representation of a proof of a temporal property.

Example:



Idea



$\mathcal{L}(P) \subseteq \mathcal{L}(\Psi)$ proved by verification conditions.

$\mathcal{L}(\Psi) \subseteq \mathcal{L}(\varphi)$ follows from well-formedness of diagram

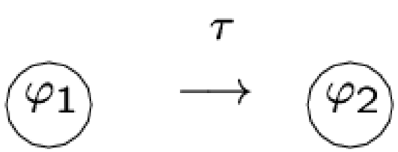
P-Valid Verification Diagrams

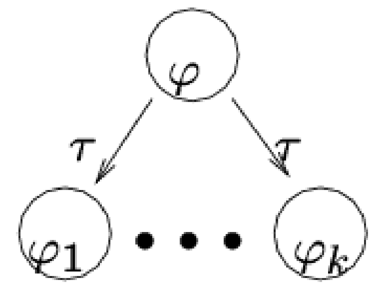
Directed labeled graph with

Verification conditions

Nodes – labeled by assertions

Edges – labeled by names of transitions





$\Rightarrow \{\varphi\}\ \tau\ \{\varphi \vee \varphi_1 \vee \ldots \vee \varphi_k\}$

Terminal Node ("goal") – no edges depart from it

Definition: VD is P-valid iff all VCs associated with nodes in the diagram are P-state valid

Wait Diagrams

VDs with nodes $\varphi_m, \ldots, \varphi_0$ such that:

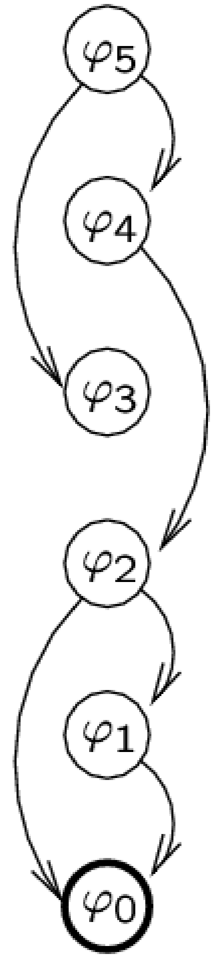
- weakly acyclic, i.e.,

  if $\varphi_i \longrightarrow \varphi_j$

  then $i \geq j$

- φ_0 is a terminal node

  φ_0



Proofs with Wait Diagrams

A P-valid WAIT diagram establishes that

$$\bigvee_{j=0}^{m} \varphi_j \;\Rightarrow\; \varphi_m \,\mathcal{W}\, \varphi_{m-1} \;\cdots\; \varphi_1 \,\mathcal{W}\, \varphi_0$$

is P-valid.

If, <u>in addition</u>,

(N1) $\quad p \;\rightarrow\; \bigvee_{j=0}^{m} \varphi_j$

(N2) $\quad \varphi_i \;\rightarrow\; q_i \quad$ for $\quad i = 0, 1, \ldots, m$

are P-state valid, then

$$p \;\Rightarrow\; q_m \,\mathcal{W}\, q_{m-1} \;\cdots\; q_1 \,\mathcal{W}\, q_0$$

is P-valid.

Example

local y_1, y_2: boolean where $y_1 = \text{F}, y_2 = \text{F}$
s : integer where $s = 1$

P_1 ::

ℓ_0 : loop forever do

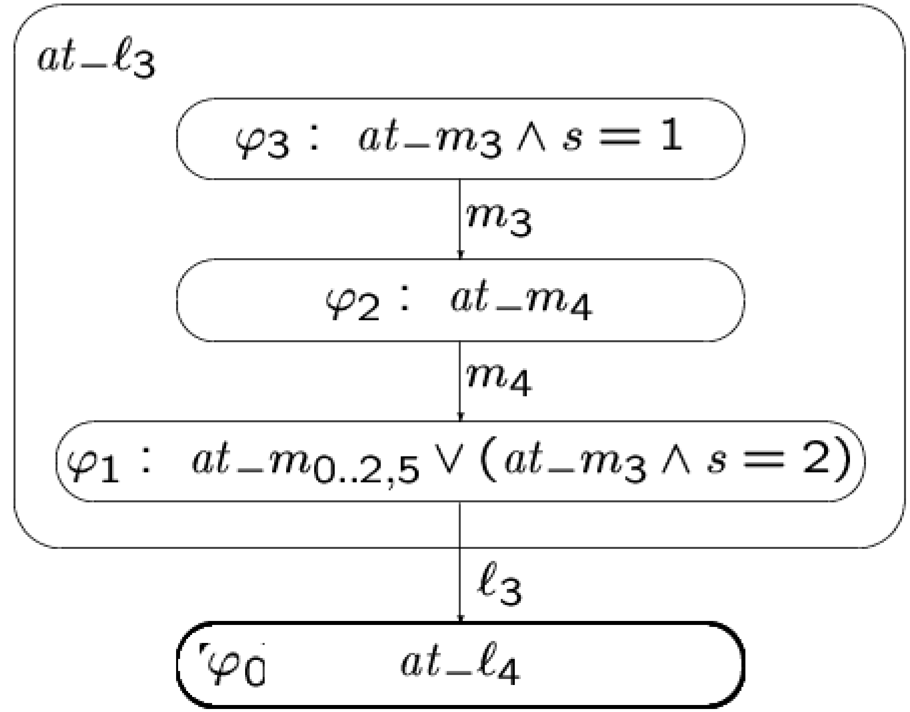
ℓ_1 : noncritical
ℓ_2 : $(y_1, s) := (\text{T}, 1)$
ℓ_3 : await $(\neg y_2) \vee (s = 2)$
ℓ_4 : critical
ℓ_5 : $y_1 := \text{F}$

||

P_2 ::

m_0 : loop forever do

m_1 : noncritical
m_2 : $(y_2, s) := (\text{T}, 2)$
m_3 : await $(\neg y_1) \vee (s = 1)$
m_4 : critical
m_5 : $y_2 := \text{F}$



Associated VCs

- From φ_3

  $\{\varphi_3\}\ m_3\ \{\varphi_3 \vee \varphi_2\}$ $\quad$ $\{\varphi_3\}\ \overline{m_3}\ \{\varphi_3\}$

- From φ_2

  $\{\varphi_2\}\ m_4\ \{\varphi_2 \vee \varphi_1\}$ $\quad$ $\{\varphi_2\}\ \overline{m_4}\ \{\varphi_2\}$

- From φ_1

  $\{\varphi_1\}\ \ell_3\ \{\varphi_1 \vee \varphi_0\}$ $\quad$ $\{\varphi_1\}\ \overline{\ell_3}\ \{\varphi_1\}$

Example (cont'd)

- $\underbrace{at\_\ell_3}_{p} \rightarrow \bigvee_{j=0}^{3} \varphi_j$

$$\varphi_0 \rightarrow \underbrace{at\_\ell_4}_{q_0} \qquad \varphi_1 \rightarrow \underbrace{\neg at\_m_4}_{q_1}$$

$$\varphi_2 \rightarrow \underbrace{at\_m_4}_{q_2} \qquad \varphi_3 \rightarrow \underbrace{\neg at\_m_4}_{q_3}$$

are state-valid.

Therefore,

$$\psi\colon \underbrace{at\_\ell_3}_{p} \Rightarrow$$

$$\underbrace{(\neg at\_m_4)}_{q_3} \; \mathcal{W} \; \underbrace{at\_m_4}_{q_2} \; \mathcal{W} \; \underbrace{(\neg at\_m_4)}_{q_1} \; \mathcal{W} \; \underbrace{at\_\ell_4}_{q_0}$$

Invariance Diagrams

VDs with no terminal nodes (cycles OK)

Claim (invariance diagram):

A P-valid INVARIANCE diagram establishes that

$$\bigvee_{j=1}^{m} \varphi_j \quad \Rightarrow \quad \square\left(\bigvee_{j=1}^{m} \varphi_j\right)$$

is P-valid.

If, in addition,

$$\text{(I1)} \quad \bigvee_{j=1}^{m} \varphi_j \rightarrow q$$

$$\text{(I2)} \quad \Theta \rightarrow \bigvee_{j=1}^{m} \varphi_j$$

are P-state valid, then

$$\square\, q \quad \text{is } P\text{-valid}$$

Example

local y_1, y_2: boolean where $y_1 = \text{F}, y_2 = \text{F}$
s : integer where $s = 1$

ℓ_0 : loop forever do

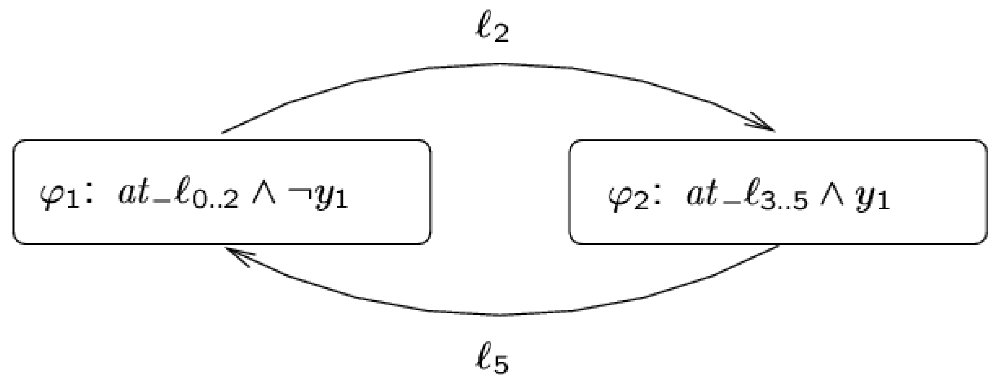
P_1 ::
$\left[\begin{array}{ll} \ell_1 : & \text{noncritical} \\ \ell_2 : & (y_1, s) := (\text{T}, 1) \\ \ell_3 : & \text{await } (\neg y_2) \vee (s = 2) \\ \ell_4 : & \text{critical} \\ \ell_5 : & y_1 := \text{F} \end{array}\right]$

||

m_0 : loop forever do

P_2 ::
$\left[\begin{array}{ll} m_1 : & \text{noncritical} \\ m_2 : & (y_2, s) := (\text{T}, 2) \\ m_3 : & \text{await } (\neg y_1) \vee (s = 1) \\ m_4 : & \text{critical} \\ m_5 : & y_2 := \text{F} \end{array}\right]$

- INVARIANCE diagram
  valid for program MUX-PET1



- Also,

(I2) $\underbrace{at\_\ell_0 \wedge \neg y_1 \wedge \cdots}_{\Theta} \rightarrow \underbrace{at\_\ell_{0..2} \wedge \neg y_1}_{\varphi_1} \vee \underbrace{\cdots}_{\varphi_2}$

(I1) $\underbrace{at\_\ell_{0..2} \wedge \neg y_1}_{\varphi_1} \rightarrow \underbrace{y_1 \leftrightarrow at\_\ell_{3..5}}_{q}$

$\underbrace{at\_\ell_{3..5} \wedge y_1}_{\varphi_2} \rightarrow \underbrace{y_1 \leftrightarrow at\_\ell_{3..6}}_{q}$

Therefore

$$\Box(y_1 \leftrightarrow at\_\ell_{3..5})$$