Transition Invariants

[A. Podelski, A. Rybalchenko, LICS '04]

Goal: Prove program termination, ideally automatically.

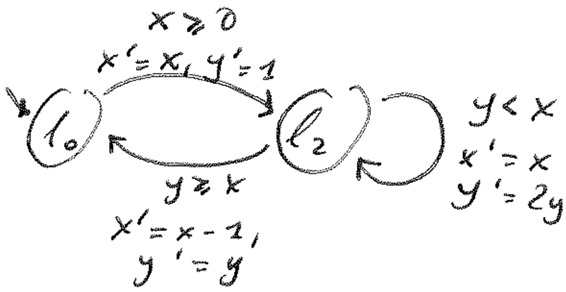
Approach: Use transition invariants, binary relations that include the transitive closure of the transition relation.

Needed: An inductiveness principle that identifies a given relation as a transition invariant.

Beyond: The technique also applies to liveness properties: the liveness property holds iff there is a disjunctively well-founded transition invariant.

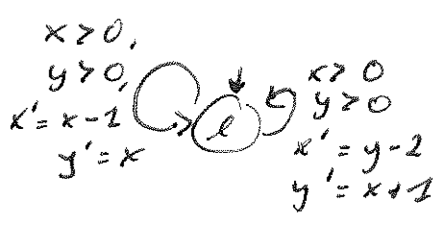
Examples: LOOPS init: $x = n \geq 0$

1) l_0 : while $x \geq 0$ do
   - l_1 : $y := 1$;
   - l_2 : while $y < x$ do
     - l_3 : $y := 2y$;
   - l_4 : $x := x - 1$



Towards a termination argument: Loop $l_0 - l_2$: decrease x; Loop l_2 : decrease $x - y$ } both non-negative.

2) CHOICE: while $x > 0 \wedge y > 0$ do
   - l^a : $(x, y) := (x - 1, x)$
   - or
   - l^b : $(x, y) := (y - 2, x + 1)$



Towards a termination argument:

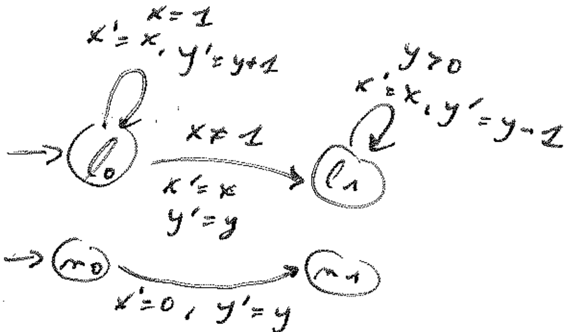
either decrease x or y or x+y.

$$(x,y) \xrightarrow{l_3} (y-2, x+1) \xrightarrow{l_a} (y-3, y-2)$$

3) ANY-DOWN

$x = 1$, $y = 0$

l_0: **while** $x = 1$ **do**	m_0: $x := 0$
$\quad l_1$: $y := y+1$	m_1:
l_2: **while** $y > 0$ **do**	
$\quad l_3$: $y := y-1$	



Does not terminate.

However: the non-terminating computation

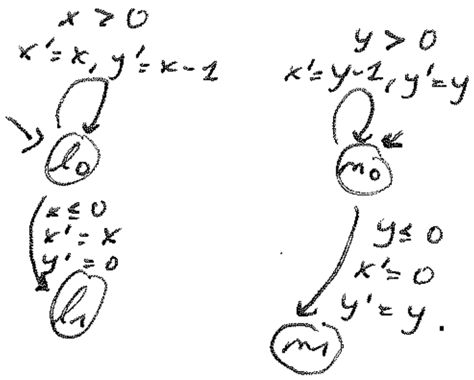
$$(l_0, m_0, 1, 0) \to (l_1, m_0, 1, 0) \to (l_0, m_0, 1, 1) \to \dots$$

is not fair:

the second thread is continuously enabled but never executed.

4) CONC-WHILES

l_0: **while** $x > 0$ **do**	m_0: **while** $y > 0$ **do**
$\quad l_1$: $y := x-1$	$\quad m_1$: $x := y-1$
l_2: $y := 0$	m_2: $x := 0$



Needs a more complicated fairness condition:

every thread has to be scheduled infinitely often.

$$(x,y) \xrightarrow{l_1} (x, x-1) \xrightarrow{l_2} (x-2, x-1) \to \dots$$

Notation:

Program $P = (W, I, R)$ with

W = set of states,

$I \subseteq W$ = set of initial states,

$R \subseteq W \times W$ = transition relation.

Computation = maximal sequence of states $s_1, s_2, \dots$

with
- $s_1 \in I$
- $(s_i, s_{i+1}) \in R$ f.a. i.

$\underline{Acc} \subseteq W$ = accessible states that appear in some computation (reachable states in our terminology).

Computation segment = infix $s_i, s_{i+1}, \dots, s_j$ of a computation.

Transition Invariants:

Definition:
A transition invariant of $P = (W, I, R)$ is a relation $T \subseteq W \times W$ with

$$R^+ \cap Acc \times Acc \subseteq T.$$

Idea: For every computation segment $s_i, s_{i+1}, \dots, s_j$, we have $(s_i, s_j) \in T$.

Example:
1) $W \times W$ is a transition invariant.
2) Every over-approximation $R^+ \subseteq O$ is a transition invariant.

Definition:

A state invariant is a set $Inv \subseteq W$ with

$$Acc \subseteq Inv$$

Lemma: Let $P = (W, I, R)$.

1) If T is a transition invariant, then

$$I \cup \{s' \mid s \in I \wedge (s, s') \in T\}$$

is a state invariant.

2) If Inv is a state invariant and T a transition invariant, then

$$T \cap Inv \times Inv$$

is a transition invariant.

Definition:

A program terminates, if every computation is finite.

Fact:

$P = (W, I, R)$ terminates iff

$R \cap Acc \times Acc$ is well-founded

(no infinite sequence $s R s' R s'' R \ldots$)

Goal: Obtain well-foundedness of the transition relation through a weaker property of its transition invariants.

Definition:

A relation T is *disjunctively well-founded*,

if it is a finite union

$$T = T_1 \cup \ldots \cup T_n$$

of well-founded relations.

Lemma:

- R well-founded $\Rightarrow$ R disjunctively well-founded.
- T disjunctively well-founded $\wedge$ $R \subseteq T$ $\not\Rightarrow$ R well-founded.

Counterexample: $\{(\text{ack}, \text{req})\} \cup \{(\text{req}, \text{ack})\}$

- $\text{ack} \longrightarrow \text{req} \;\cup\; \text{req} \longrightarrow \text{ack}$ disjunctively well-founded.
- $\text{ack} \rightleftarrows \text{req}$ not well-founded.

$$\{(\text{ack}, \text{req}), (\text{req}, \text{ack})\}.$$

- T disjunctively well-founded $\wedge$ $R^+ \subseteq T$ $\Rightarrow$ R well-founded.
  ..... transition invariant (even stronger)

The proof of the last point
is part of the following main result.

Theorem (Termination):

Program P terminates iff $\exists$ disjunctively well-founded transition invariant for P.

Proof:

"$\Leftarrow$" Let

$$T = T_1 \cup \ldots \cup T_n$$

be a disjunctively well-founded transition invariant.

Towards a contradiction, assume P does not terminate.

Then there is an infinite computation

$$\sigma = s_1 . s_2 . s_3 \ldots$$

We define a coloring of the infinite complete graph $K_{\mathbb{N}}$.

$$f : \mathbb{N} \times \mathbb{N} \to \{T_1, \ldots, T_n\}$$

$$f(k, l) = T_i \ , \ \text{if} \ (s_k, s_l) \in T_i \ .$$
with $k < l$

Because T is a transition invariant,
we know that T_i exists.
The coloring is finite.
Hence, Ramsey's theorem applies
and yields an infinite complete subgraph
that is colored by a single color, say T_i.
Let the nodes be

$$s_{i_1} \ T_i \ s_{i_2} \ T_i \ s_{i_3} \ T_i \ldots$$

The sequence contradicts the fact that T_i is well-founded,
so the assumption that P does not terminate is false.

"$\Rightarrow$" Assume P terminates.

Define $\quad T = R^+ \cap Acc \times Acc.$

Now T is a transition invariant by definition.

Assume

$$\sigma = s^1 . s^2 \ldots$$

is an infinite sequence of states

with $(s_i, s_{i+1}) \in T$ f.a. i.

Since
- s^1 is accessible and
- there is a non-empty computation segment leading from s^i to s^{i+1}, for all i (meaning $(s_i, s_{i+1}) \in R^+$)

there is an infinite computation

$$s_1 \ldots s^1 \ldots s^2 \ldots s^3 \ldots$$

This contradicts the fact that P terminates.

Hence, T is (disjunctively) well-founded. $\square$

<u>Fact</u>:

We cannot drop the finiteness requirement in the definition of disjunctive well-foundedness.

The relation

$$R = \{(i, i+1) \mid i \geq 1\}$$

has a transition invariant

$$T = T_1 \cup T_2 \cup \ldots$$

that is the union of well-founded relations

$$T_i = \{(i, i+j) \mid j \geq 1\}.$$

However, R is not well-founded.

Termination:

Goal: Devise disjunctively well-founded transition invariants for the first two examples from above.

LOOPS:

$$T = T_1 \cup T_2 \cup \bigcup_{\substack{i \neq j,\\ i,j \in \{0,\dots,4\}}} T_{ij}$$

with

$$T_1 = x \geq 0 \wedge x' < x$$

$$T_2 = x - y > 0 \wedge x' - y' < x - y$$

$$T_{ij} = pc = l_i \wedge pc' = l_j.$$

We give an intuitive argument why T is a transition invariant.

A proof technique follows.

There are three kinds of computation segments that lead a state s to s':

↳ $(s, s') \in R^+$ via l_0 : then $(s, s') \in T_1$.

↳ $(s, s') \in R^+$ via l_1 but <u>not l_0</u> : then $(s, s') \in T_2$.

↳ $(s, s') \in R^+$ with different location labels : then $(s, s') \in T_{ij}$ for some i, j.

CHOICE:

$$T = T_1 \cup T_2 \cup T_3$$

with

$$T_1 = x > 0 \wedge x' < x$$

$$T_2 = x + y > 0 \wedge x' + y' < x + y$$

$$T_3 = y > 0 \wedge y' < y.$$