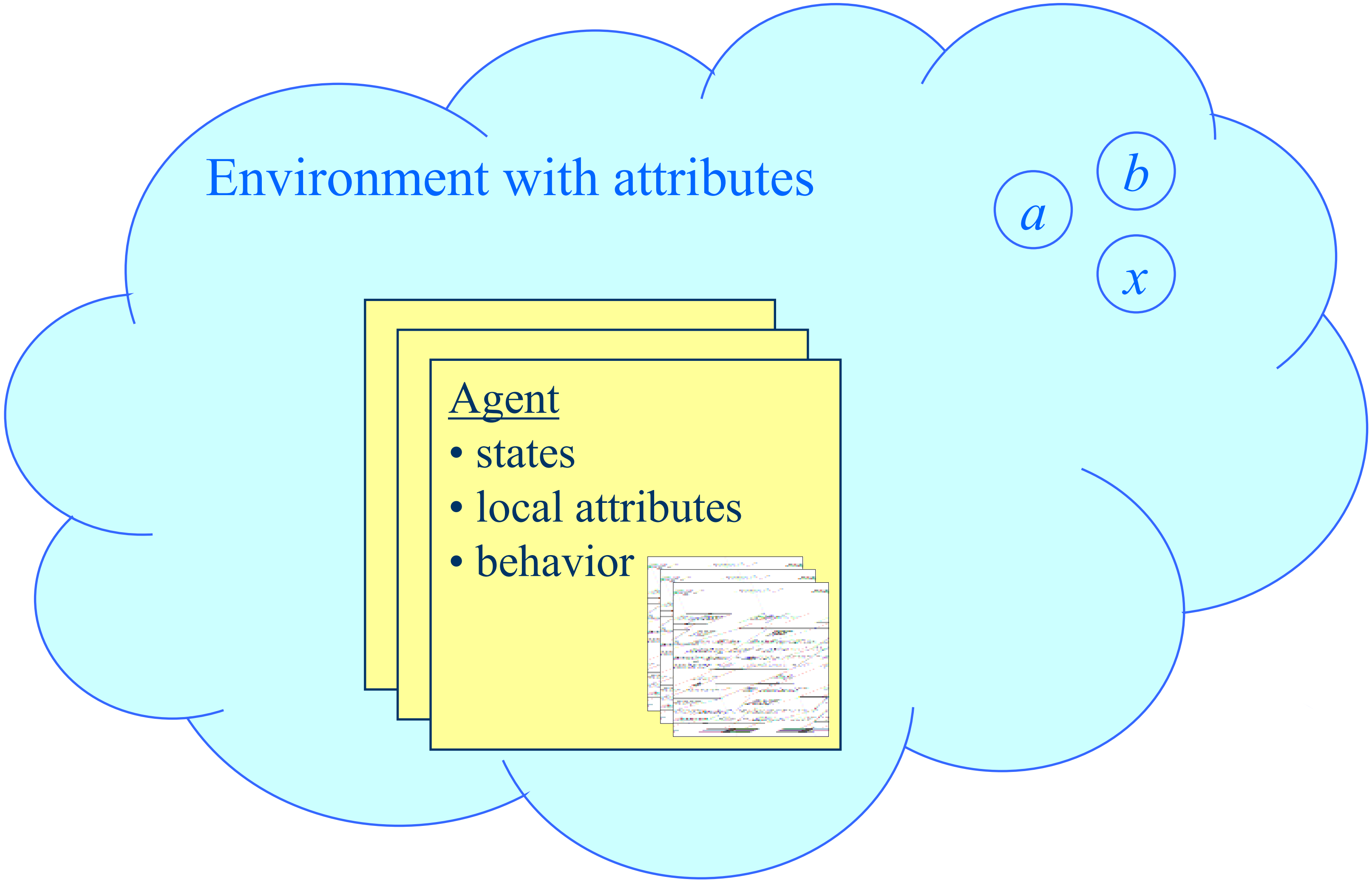
Static Requirements Checking and Reachability Problem

Oleksandr Letychevskyi,

Stepan Potiyenko

May-08

Agents and Environment



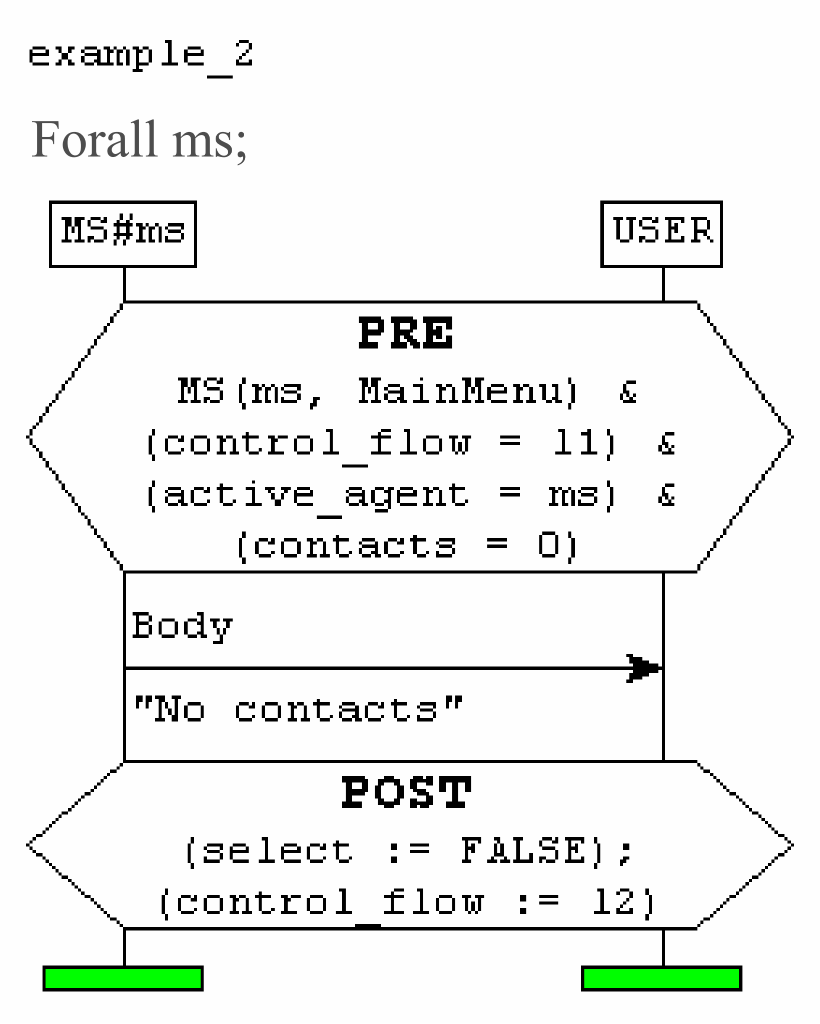
Basic Protocols

Basic Protocol is a triple

$$\forall x(\alpha \rightarrow < u > \beta)$$

where:

- x is a list of parameters,
- α – is a precondition,
- u – process (action),
- β – post condition



Transition consistency

- Consistent system has deterministic behavior.
- For each agent state preconditions should not intersect.

$$\forall(i,j)\,(\,i \neq j \wedge S_i = S_j \;\rightarrow\; \neg(\alpha_i \wedge \alpha_j)\,)$$

where:

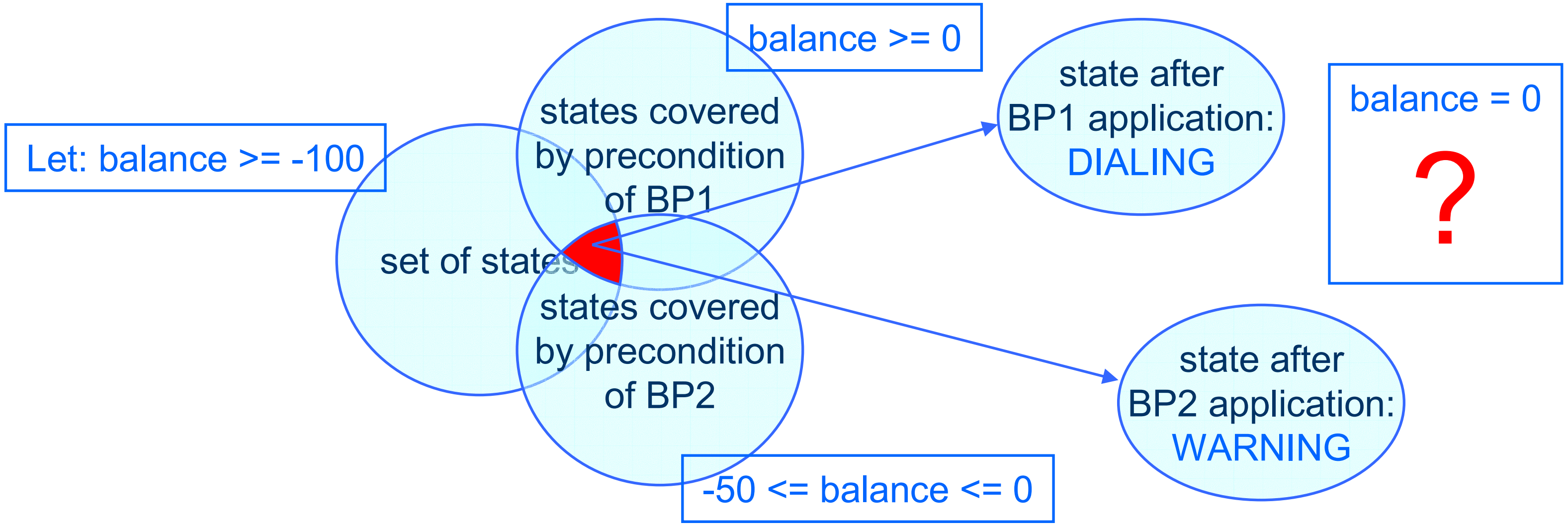
- S_n – agent state in precondition of basic protocol n.
- α_n – precondition of basic protocol n.

Transition consistency

- Example of inconsistency

precondition:
phone(p, dial);
balance >= 0

precondition:
phone(p, dial);
-50 <= balance <= 0



Transition completeness

- Complete system never comes to deadlock.
- For each agent state disjunction of all preconditions should be true (taking into account restrictions).

$$\forall i \exists s \, (S_i = s \;\rightarrow\; R_s \vee \bigvee_i \alpha_i \,)$$

where

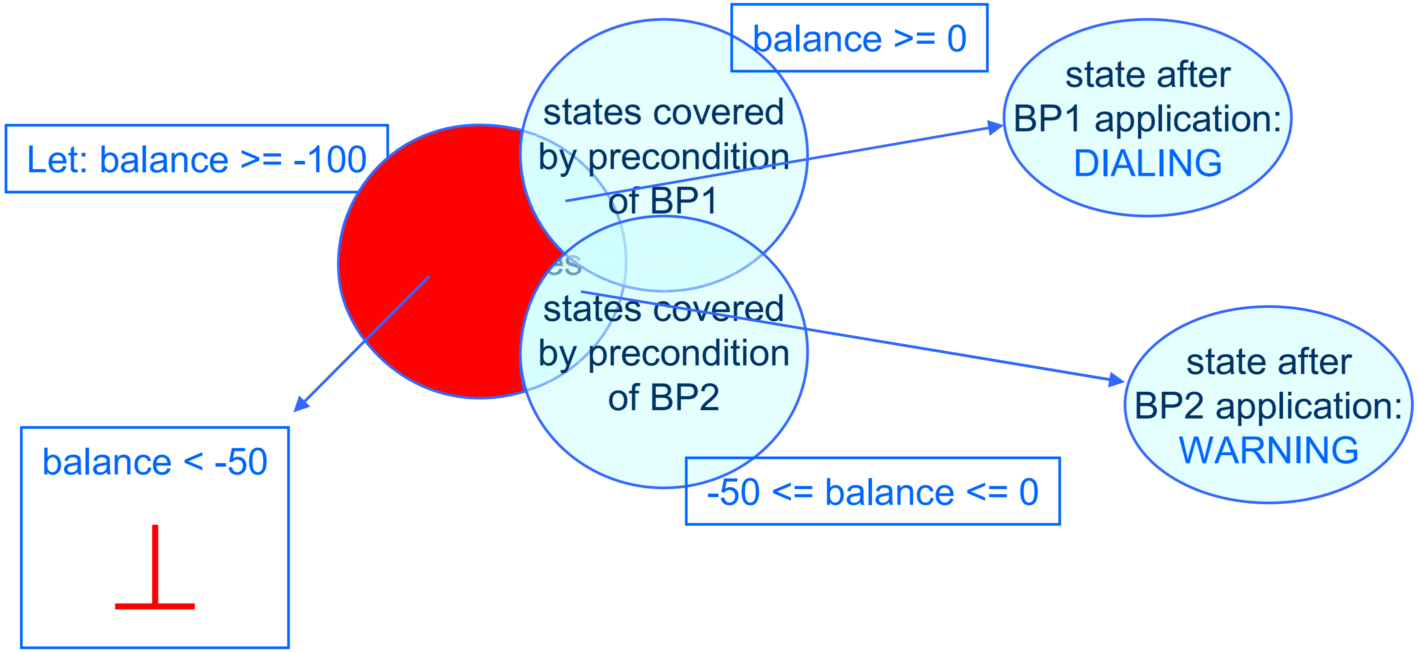
- S_n – agent state in precondition of basic protocol n.
- R_s – restriction for state s.
- α_n – precondition of basic protocol n.

Transition completeness

- Example of incompleteness

precondition:
phone(p, dial);
balance >= 0

precondition:
phone(p, dial);
-50 <= balance <= 0



Safety

- First, initial state is checked

$$\forall x \, (\, I(x) \rightarrow Q(x) \,)$$

where:

- Q – safety condition.
- I – initial state.
- x – a set of attributes.

Safety

- Second, each applicable basic protocol should be invariant relatively to safety condition.

$$\forall i \, (\alpha_i \wedge Q)$$

$$\forall i \, (Pt(\alpha_i \wedge Q, \beta_i) \rightarrow Q)$$

where:

- Q – safety condition.
- $Pt(X, \beta_i)$ – environment state X transformed by postcondition β_i.
- α_i – precondition of basic protocol i.
- β_i – postcondition of basic protocol i.

Safety

- Example of safety violation

safety condition:
balance > 0

precondition:
phone(p, dial); balance > 0
postcondition:
phone(p, dialing); balance := balance - 1

precondition:
phone(p, dial); -50 <= balance <= 0
postcondition:
phone(p, warning)

- BP1 breaks safety
- Counter-example:
  - Let balance = 1;
  - Apply BP1;
  - balance = 0.

- Precondition of BP2 breaks safety
- What does it mean?
  - If safety is correct it always should be true, consequently, BP2 will never be applied – unreachable protocol;
  - Otherwise, safety is incorrect and should be revised

Reachability

- Found inconsistency, incompleteness and safety violations prove existence of “bad” states where system has nondeterministic behavior, deadlock or breaks safety conditions.

- Problem: Are these states reachable?

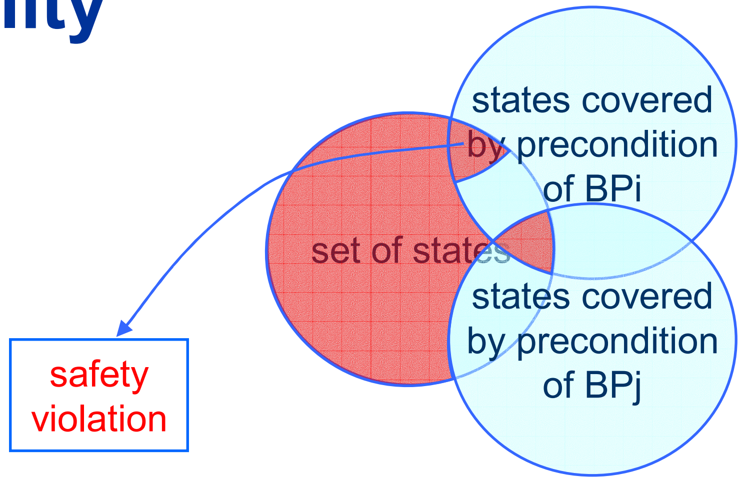
- Let’s build formulas describing “bad” states.

Reachability

precondition: $\alpha_i(x)$
postcondition: $\beta_i(x)$

precondition: $\alpha_j(x)$

x – a set of attributes



- Let protocols i and j are inconsistent. It means that the following formula is true: $\exists(x)\ (\ \alpha_i(x) \wedge \alpha_j(x)\ )$
  It is a “bad” state.
- Let protocols i and j cover incomplete state. It means that the following formula is true: $\exists(x)\ \neg(\ \alpha_i(x) \vee \alpha_j(x)\ )$
  It is a “bad” state.
- Let protocol i breaks safety Q. It means that some state $X(x)$ exists such as the following formula is true:
  $\exists(x)\ (\ \alpha_i(x) \wedge Q(x) \wedge X(x) \wedge \neg(\ Pt(X(x),\ \beta_i(x)) \rightarrow Q(x)\ )\ )$
  X is a “bad” state.

Approaches to reachability problem solution

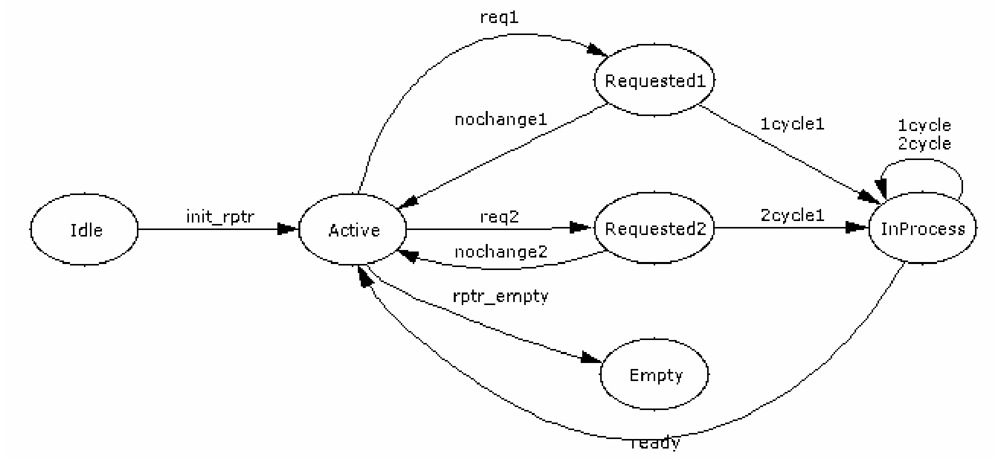
- Approach 1. To implement some static filtering to avoid unreachable “bad” states:
- Let’s consider negation of every “bad” state B as safety condition: $\neg B$
- If the system never breaks condition $\neg B$ then “bad” state B is unreachable – it can be omitted.
- Otherwise – the problem remains opened.

- Specification of user-defined restricted states is another method of filtering.
- Let R – formula specifying restricted states.
- If formula $R \rightarrow B$ is true then “bad” state B is restricted and can be filtered.

Approaches to reachability problem solution

- Approach 2. Based on using model checking and symbolic modeling.
- Consider simplified behavior of the model and build directed graph of transitions using attributes subset. Let's choose agent state as a subset. It satisfies the following properties in any basic protocols system:
  - occurs in precondition of each protocol;
  - always has concrete value.

Approaches to reachability problem solution



- Graph nodes u_i are formulas which specify value of chosen subset of attributes (in this case – names of agent states);
- Graph edges r_j are the names of basic protocols.

- Consider “bad” state B and find nodes such as: $u_i \wedge B \neq 0$
- Build all paths from the node specifying initial state of the system to found nodes.
- This set of paths is complete but redundant. These paths can be used for guided search in model checking or symbolic modeling as forward as backward.

Symbolic modeling

- Here we consider methods of state space generation in symbolic modeling.
- Pre- and postcondition of any basic protocol can be represented in the following form:
  - precondition: $A(r,l,s,z)$;
  - postcondition: $B(r,l,s,z) = (\ r := t(r,l,s,z)\ ) \wedge U(l,r,s,z) \wedge C(r,l,s)$.
- Here:
  - l – the vector of list attributes;
  - r,s,z – vectors of attribute expressions of numeric and symbolic types;
  - $A(r,s,z)$ and $C(r,l,s)$ – basic languages formulas;
  - $U(l,r,s,z)$ – conjunction of list updating operators (l – updated lists);
  - $r := t(r,s,z)$ – conjunction of assignments for attributes r: $(r_1 := t_1(r,s,z)) \wedge (r_2 := t_2(r,l,s,z)) \wedge \ldots$;
  - z – the vector of attributes which occur in assignments and list updating operators but absent in formula $C(r,l,s)$.
  - all lists l are excluded from precondition A and assignments in postcondition because list access operators can be substituted by corresponding expressions (first or last element of a list).

Symbolic modeling

- Basic protocol is applicable on state class E if formula $E \wedge A(r,l,s,z)$ is true. Applicable protocol makes transition:
  $E \rightarrow E'$
- Here E and E' are formulas that specify state classes. They are represented as:
  - $E = F(r,s,z) \wedge L(r,l,s,z)$
  - $E' = F'(r,s,z) \wedge L'(r,l,s,z)$
- Where:
  - $F(r,s,z)$, $F'(r,s,z)$ – basic language formulas;
  - $L(r,l,s,z)$, $L'(r,l,s,z)$ – list equalities:
    - $(l_1 = list(head_1(r,s,z), \ldots, tail_1(r,s,z))) \wedge (l_2 = list(head_2(r,s,z), \ldots, tail_2(r,s,z))) \wedge \ldots$;
    - $head_i(r,s,z)$ and $tail_i(r,s,z)$ – sequences of expressions, can be empty;
    - $\ldots$ - abstract (unknown) part of the list; it's absent in lists with concrete length.
- Transition from given state to the next one is made by predicate transformers defined as functions of formulas deduction.

Forward predicate transformer

$$E' = pt(\ E \wedge A(r,l,s,z),\ B(r,l,s,z)\ )$$

$$E' = pt(\ F(r,s,z) \wedge L(l,r,s,z)\ \wedge A(r,s,z),\ B(r,l,s,z)\ )$$

$$E' = E_1 \vee E_2 \vee \ldots$$

$$E_i = \exists(u,v)\ (\ F(u,v,\xi_i) \wedge A(u,v,\xi_i) \wedge T(r,u,v,\xi_i) \wedge L(l,u,v,\xi_i) \wedge P_i(u,v,r,s)\ ) \wedge C(r,l,s)$$

$$T(r,u,v,\xi_i) = (\ (r_1 = t_1(u,v,\xi_i)) \wedge (r_2 = t_2(u,v,\xi_i)) \wedge \ldots\ )$$

$$L(l,u,v,\xi_i) = (\ (l_1 = list(head_1(u,v,\xi_i),\ \ldots,\ tail_1(u,v,\xi_i))) \wedge (l_2 = list(head_2(u,v,\xi_i),\ \ldots,\ tail_2(u,v,\xi_i))) \wedge \ldots\ )$$

- Here u,v – vectors of new variables introduced for signing old values of attributes r,s. $L(l,u,v,z)$ contains updated lists after operators $U(l,r,s,z)$ application. If one attribute of functional type occurs in postcondition more than once we should consider all possible identifications of its arguments. Formula $P_i(u,v,r,s)$ specifies one of such possibilities. ξ_i derived from vector z taking into account $P_i(u,v,r,s)$.

Backward predicate transformer

$$E = pt^{-1}(\ E',\ A(r,l,s,z),\ B(r,l,s,z)\ )$$

– Let $(\ r = t(u,s,z) \wedge C(r,l,s)\ ) \neq 0$ (valid postcondition).

$$F(r,s,z) = \exists v\ (\ F'(t(r,v,z),v,z)\ ) \wedge A(r,l,s,z) \wedge P(r,s,z)$$

– If formula $F'(r,s,z)$ is false then given basic protocol could not be applied and corresponding behavior branch is not considered.

– List updating operators $U(r,l,s,z)$ change list equalities in the environment state. $U(r,l,s,z)$ contains operators:

- *add_to_tail(l, f(r,s,z))*
- *add_to_head(l, f(r,s,z))*
- *remove_from_tail(l, f(r,s,z))*
- *remove_from_head(l, f(r,s,z))*

– Updating of the lists and generation of list equalities $L(l,r,s,z)$ is made by inverse operators to $U(r,l,s,z)$.

Demo

- CDMA (target site) was checked by SRC.
- One safety condition formulated:
  - (SDU tsdu.SdfMsHHoCmpltT >= 0) & (SDU tsdu.SdfMsHHoCmpltT < 2)
  - It means that timer never started twice and never stopped twice.

Tool	inconsistency / nondeterminism	incompleteness / deadlock	safety violation
SRC	118 pairs of protocols	6 classes of states	4 protocols
CTG with Maxtraces=10000 (28 protocols were not applied)	0 concrete states	110 concrete states	3 protocols