T-UPPAAL:
Real-Time Online Testing Tool

Marius Mikucionis, Kim G. Larsen, Brian Nielsen

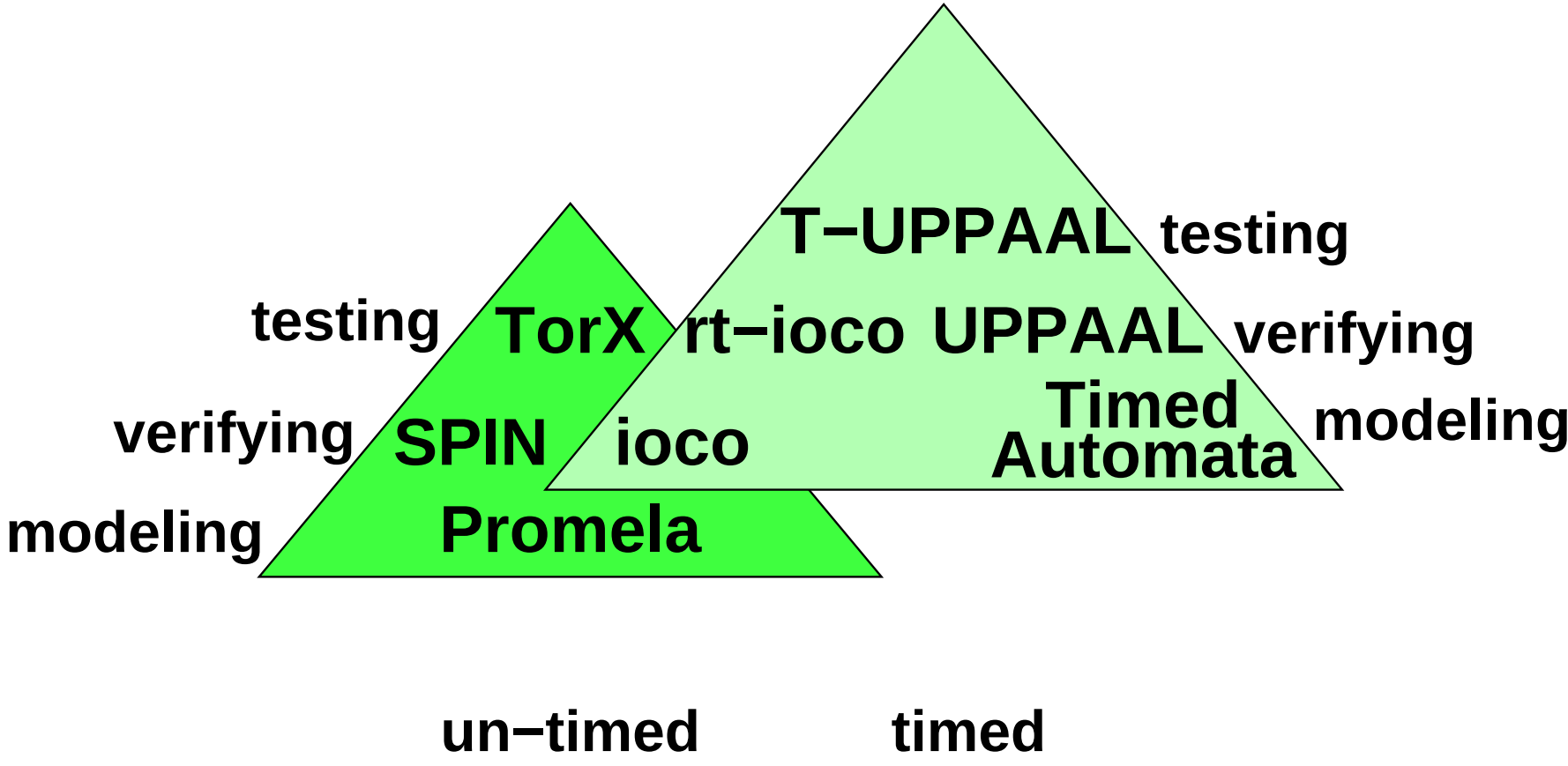
{ marius, kgl, bnielsen } @cs.auc.dk.

Department of Computer Science

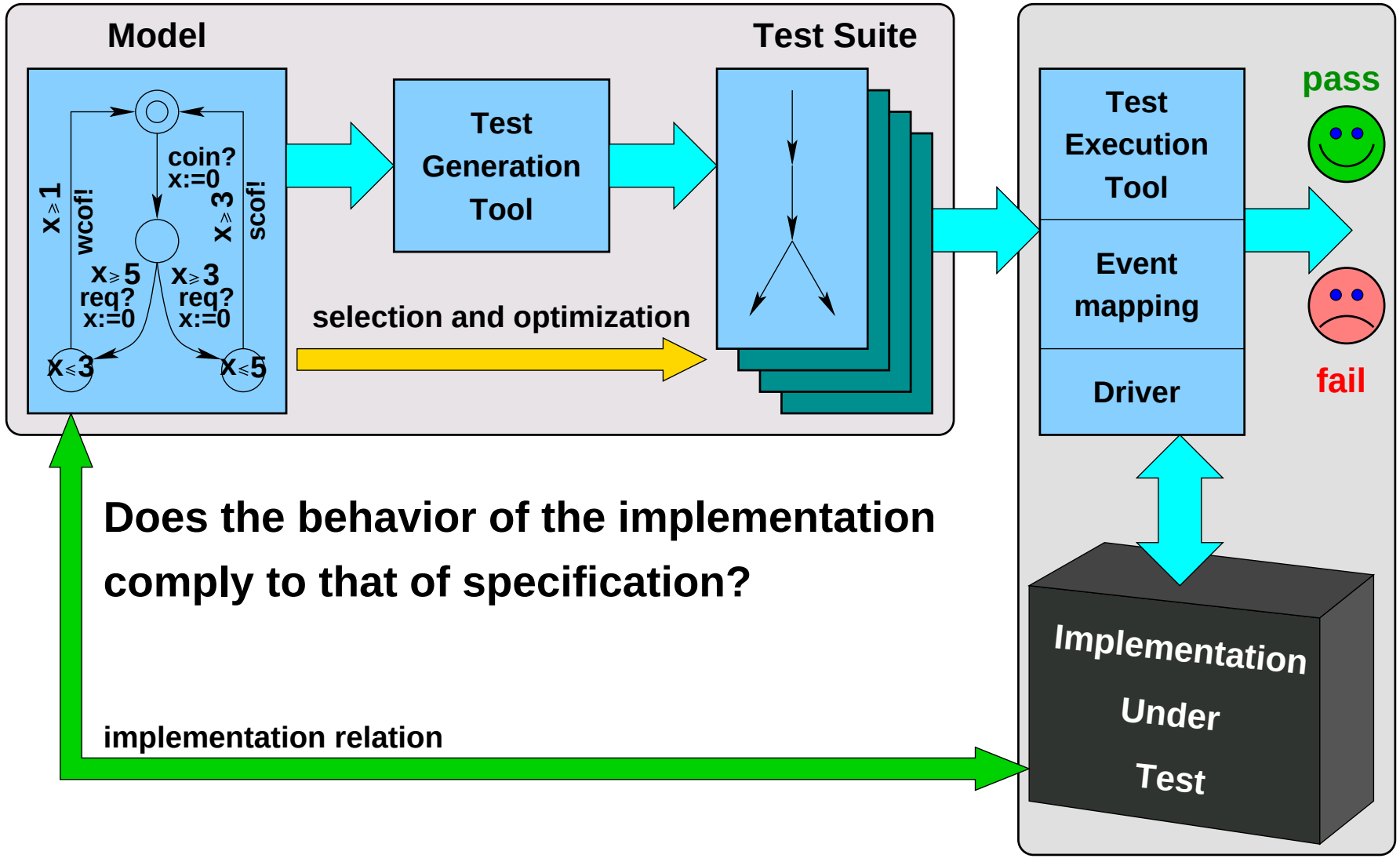
Aalborg University

T-UPPAAL Context

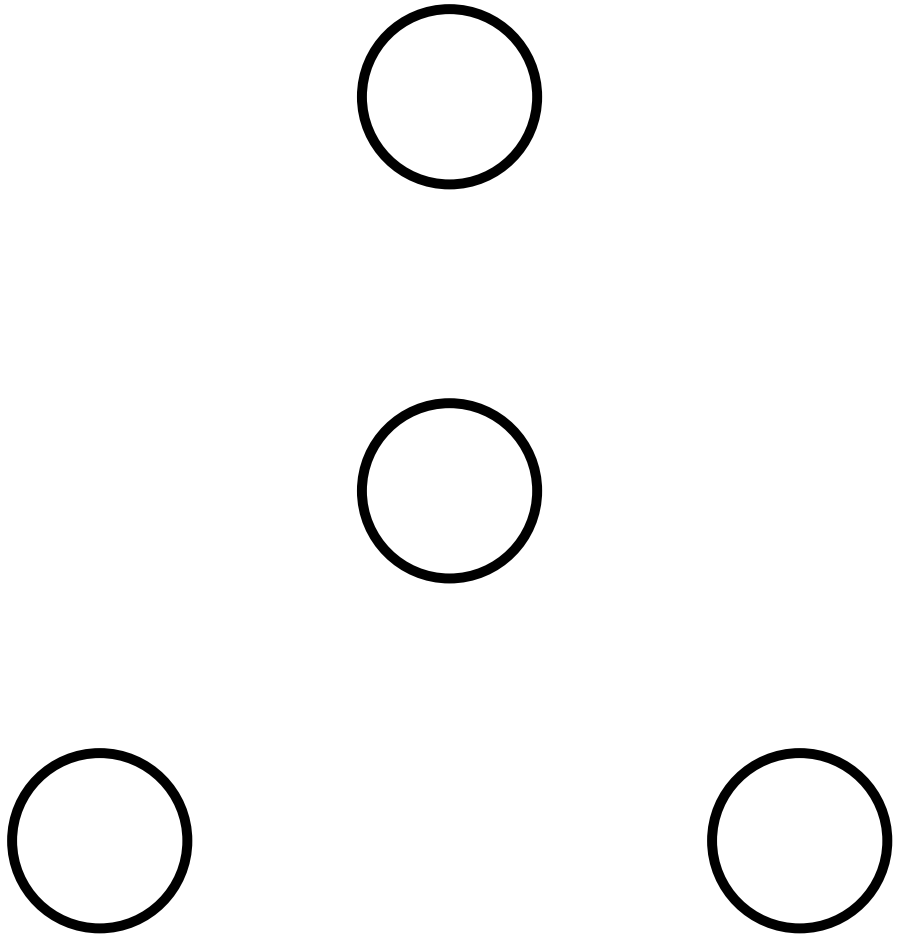
- The work is relying on the following giants:



Black-box Testing Framework



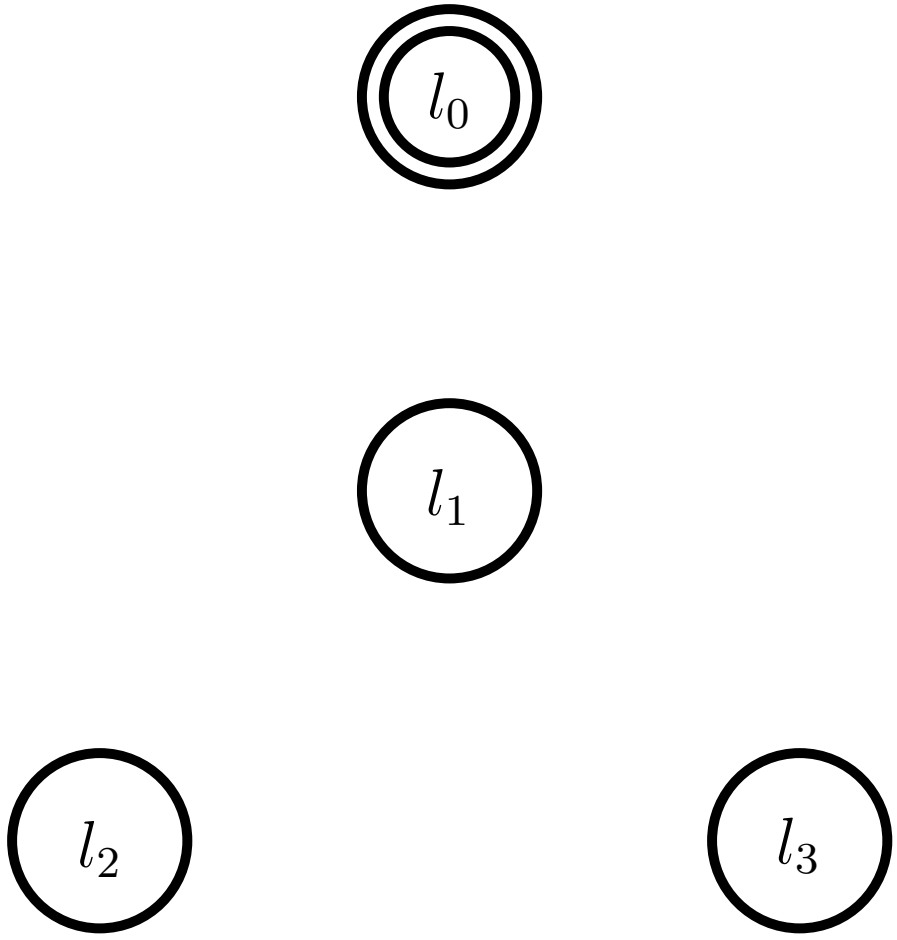
System Model Specification



Timed automaton over actions A is a tuple (L, l_0, X, D, E, I):

- L is set of *locations*,

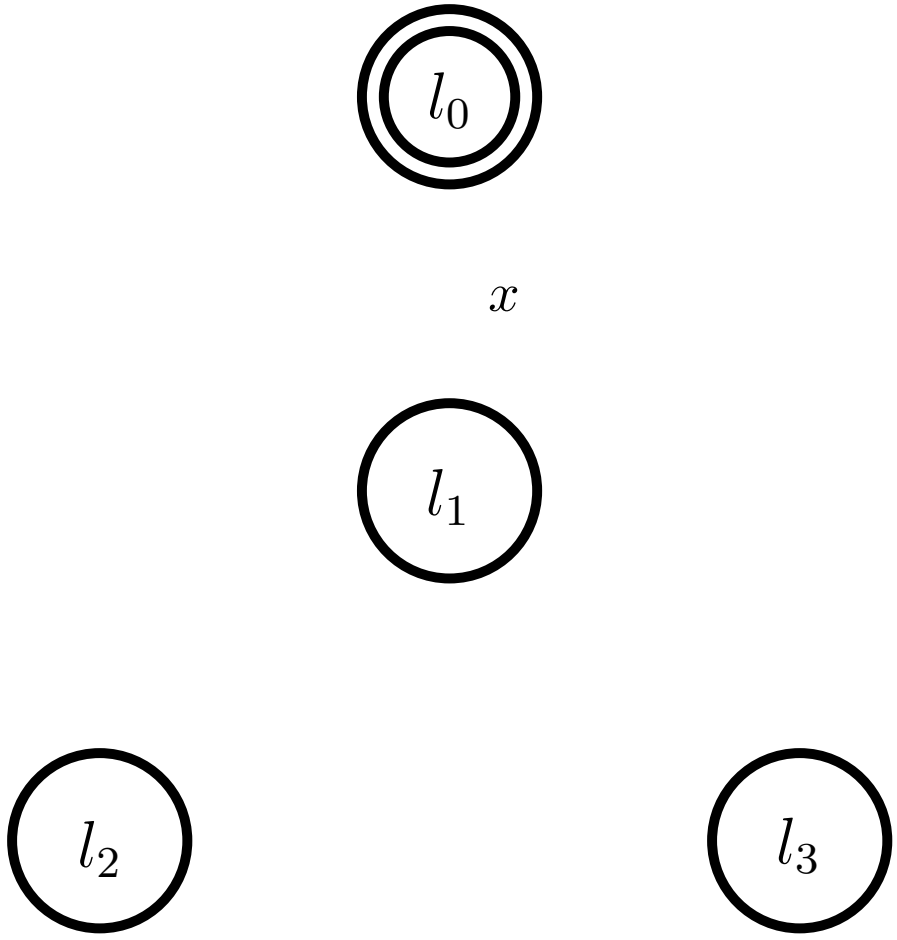
System Model Specification



Timed automaton over actions A is a tuple (L, l_0, X, D, E, I):

- L is set of *locations*,
- $l_0 \in L$ is the *initial* location,

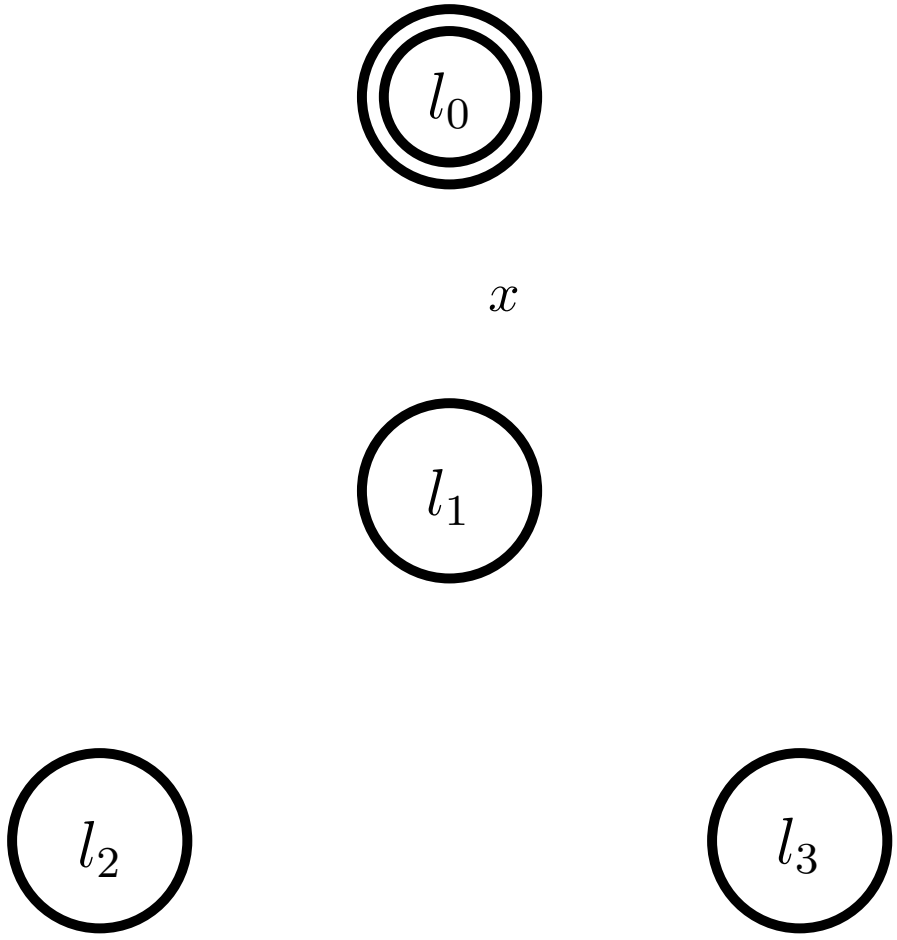
System Model Specification



Timed automaton over actions A is a tuple (L, l_0, X, D, E, I):

- L is set of *locations*,
- $l_0 \in L$ is the *initial* location,
- X is set of *real-valued clocks*,

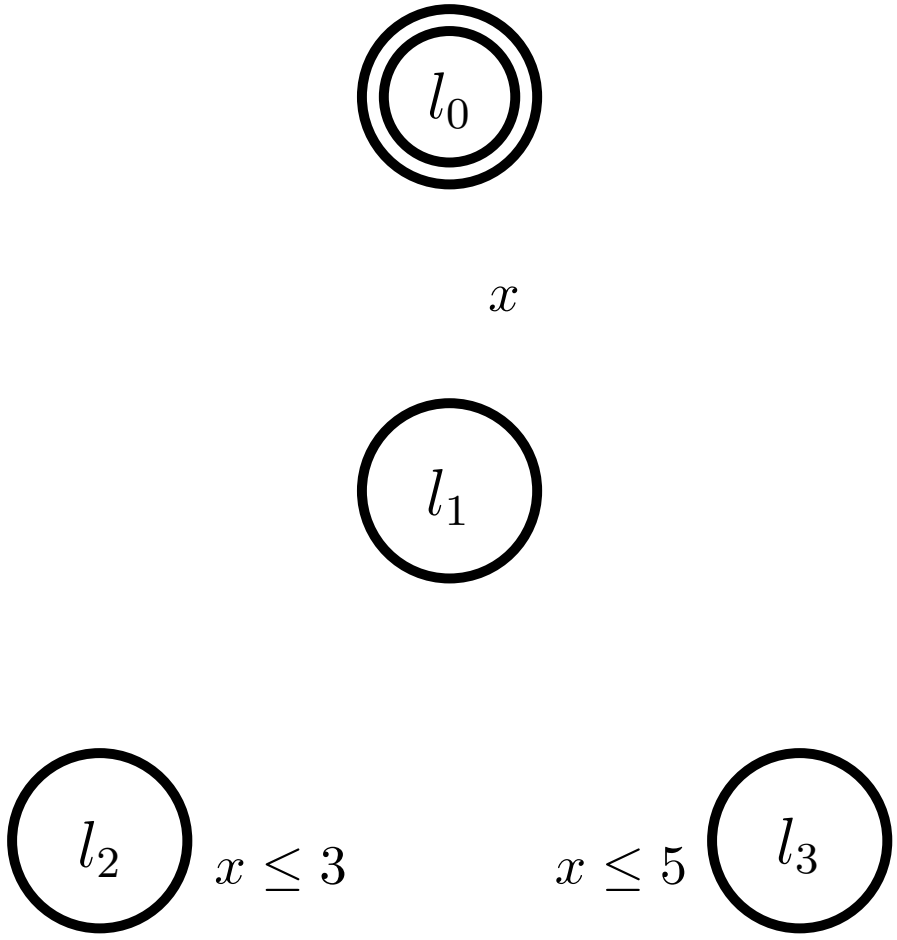
System Model Specification



Timed automaton over actions A is a tuple (L, l_0, X, D, E, I):

- L is set of *locations*,
- $l_0 \in L$ is the *initial* location,
- X is set of *real-valued clocks*,
- D - bounded integer *variables*,

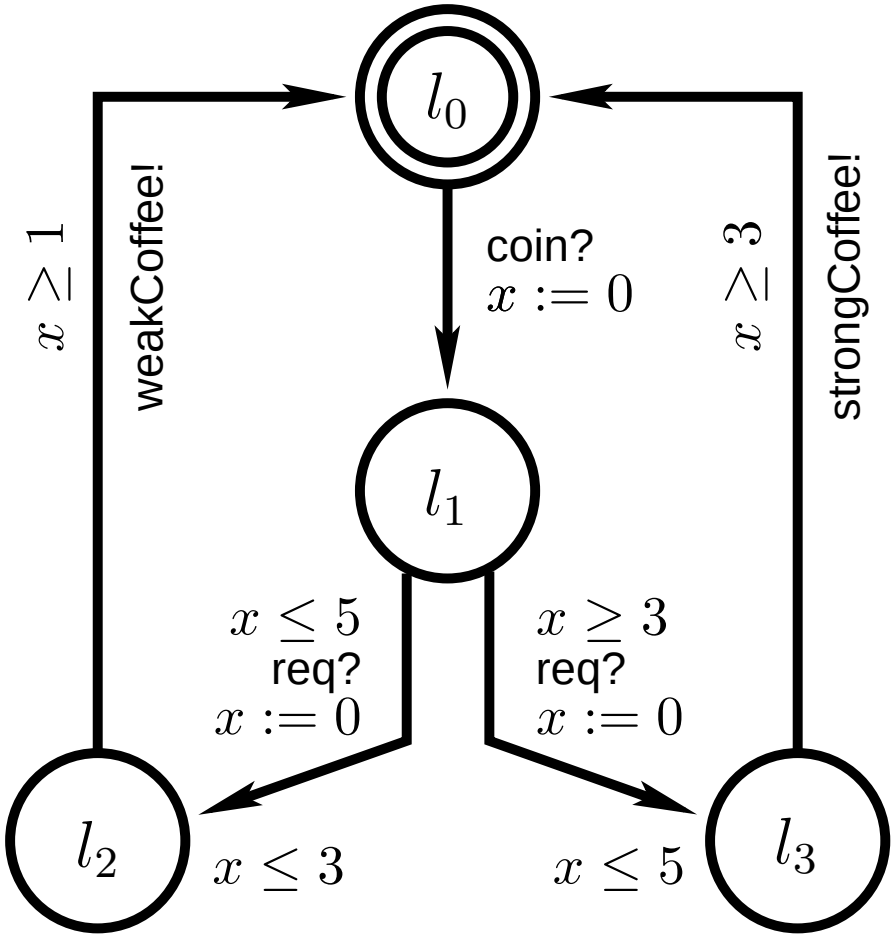
System Model Specification



Timed automaton over actions A is a tuple (L, l_0, X, D, E, I):

- L is set of *locations*,
- $l_0 \in L$ is the *initial* location,
- X is set of *real-valued clocks*,
- D - bounded integer *variables*,
- $I : l \mapsto G(X)$ is the location *invariant* mapping,

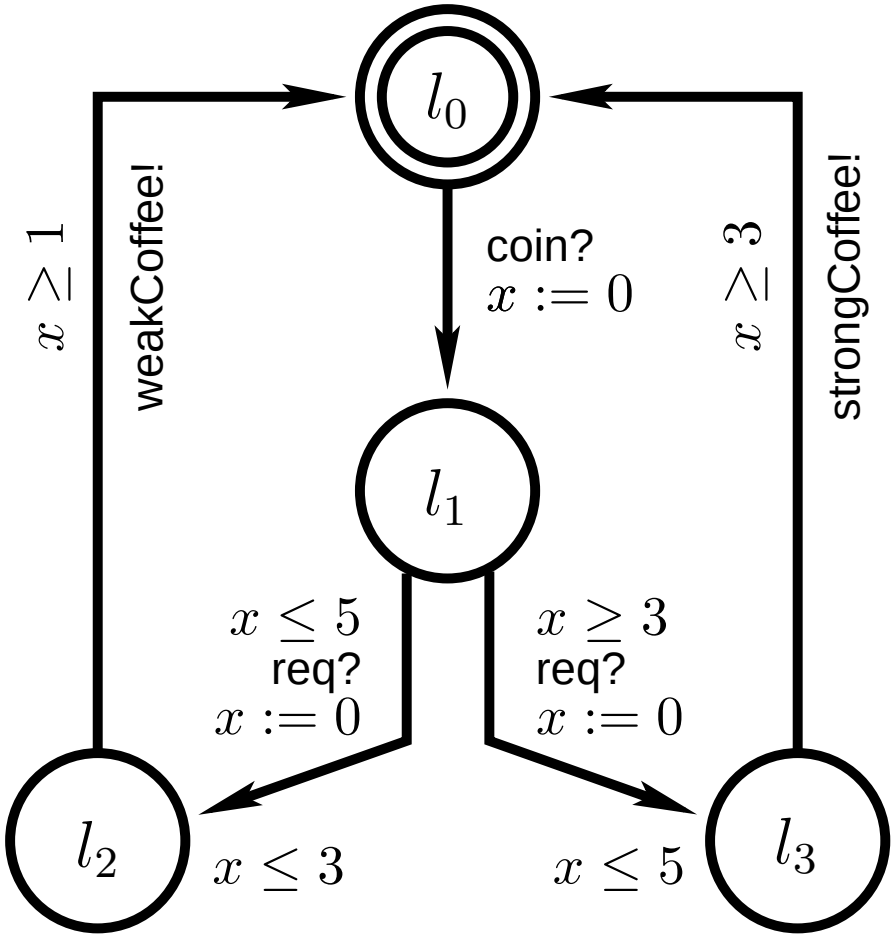
System Model Specification



Timed automaton over actions A is a tuple (L, l_0, X, D, E, I):

- L is set of *locations*,
- $l_0 \in L$ is the *initial* location,
- X is set of *real-valued clocks*,
- D - bounded integer *variables*,
- $I : l \mapsto G(X)$ is the location *invariant* mapping,
- $E \subseteq L \times G(X) \times A \times 2^{R(X)} \times L$ is a superset of directed *edges*: $l \xrightarrow{g,a,r} l'$ iff $\langle l, g, a, r, l' \rangle \in E$.

System Model Specification



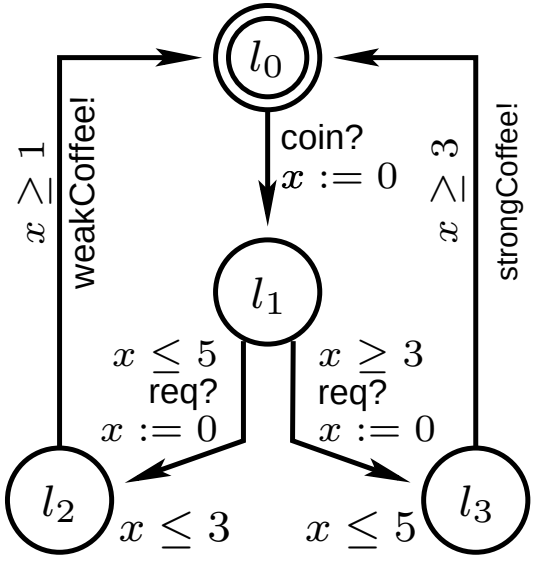
Timed automaton over actions A is a tuple (L, l_0, X, D, E, I):

- L is set of *locations*,
- $l_0 \in L$ is the *initial* location,
- X is set of *real-valued clocks*,
- D - bounded integer *variables*,
- $I : l \mapsto G(X)$ is the location *invariant* mapping,
- $E \subseteq L \times G(X) \times A \times 2^{R(X)} \times L$ is a superset of directed *edges*: $l \xrightarrow{g,a,r} l'$ iff $\langle l, g, a, r, l' \rangle \in E$.

- *Labeled Transition System (LTS)* semantics.

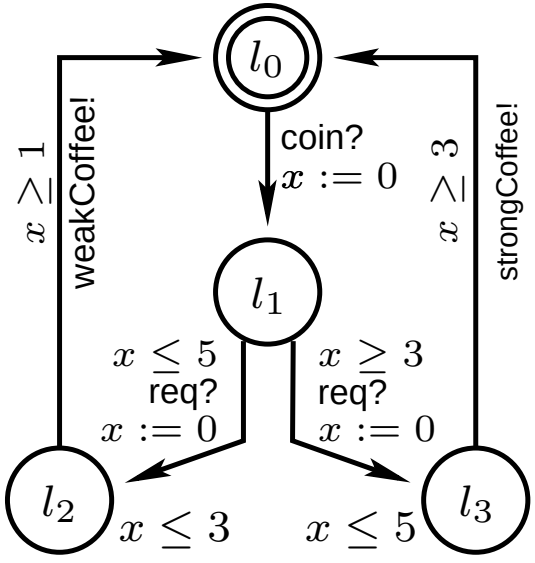
Implementation Relation: rt-ioco

- Timed trace eg.: $\sigma = coin? \cdot 5 \cdot req? \cdot 2 \cdot weakCoffee! \cdot 9 \cdot coin?$



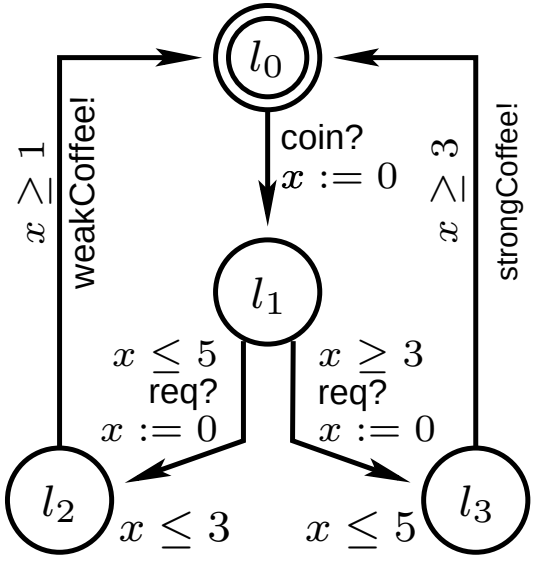
Implementation Relation: rt-ioco

- Timed trace eg.: $\sigma = coin? \cdot 5 \cdot req? \cdot 2 \cdot weakCoffee! \cdot 9 \cdot coin?$
- $\mathbf{ttraces}(s)$ – set of *timed traces* from s: $\{\sigma \in (A \cup \mathbb{R}_+)^* \mid s \stackrel{\sigma}{\Rightarrow}\}$



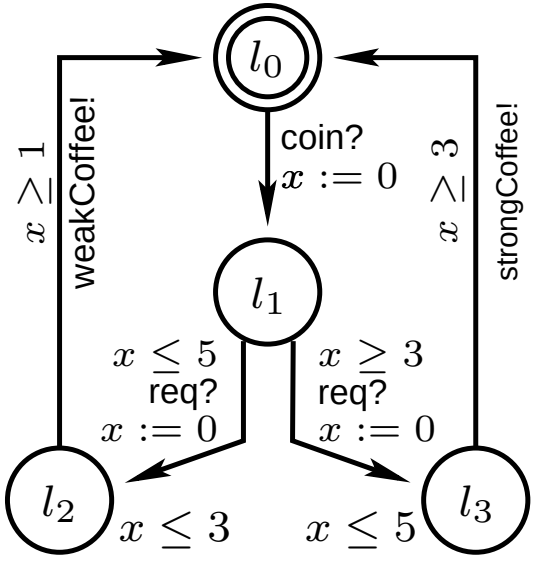
Implementation Relation: rt-ioco

- Timed trace eg.: $\sigma = coin? \cdot 5 \cdot req? \cdot 2 \cdot weakCoffee! \cdot 9 \cdot coin?$
- $\mathbf{ttraces}(s)$ – set of *timed traces* from s: $\{\sigma \in (A \cup \mathbb{R}_+)^* \mid s \stackrel{\sigma}{\Rightarrow}\}$
- Timed trace *inclusion* as conf. relation: $\mathbf{ttraces}(i) \subseteq \mathbf{ttraces}(s)$



Implementation Relation: rt-ioco

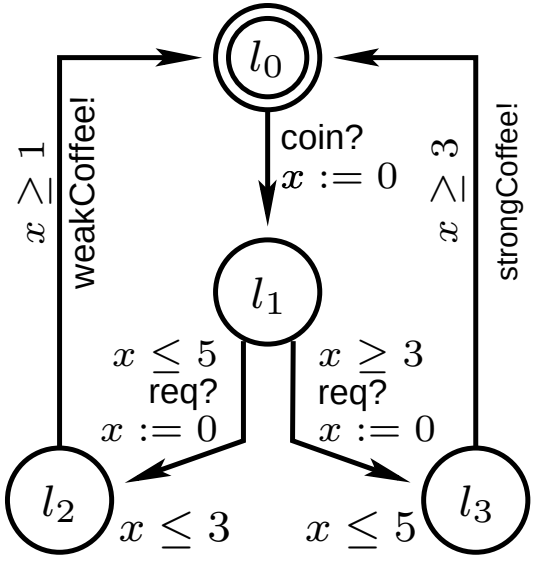
- Timed trace eg.: $\sigma = coin? \cdot 5 \cdot req? \cdot 2 \cdot weakCoffee! \cdot 9 \cdot coin?$
- $\mathbf{ttraces}(s)$ – set of *timed traces* from s: $\{\sigma \in (A \cup \mathbb{R}_+)^* \mid s \stackrel{\sigma}{\Rightarrow}\}$
- Timed trace *inclusion* as conf. relation: $\mathbf{ttraces}(i) \subseteq \mathbf{ttraces}(s)$



- no illegal output is produced and

Implementation Relation: rt-ioco

- Timed trace eg.: $\sigma = coin? \cdot 5 \cdot req? \cdot 2 \cdot weakCoffee! \cdot 9 \cdot coin?$
- $\mathbf{ttraces}(s)$ – set of *timed traces* from s: $\{\sigma \in (A \cup \mathbb{R}_+)^* \mid s \stackrel{\sigma}{\Rightarrow}\}$
- Timed trace *inclusion* as conf. relation: $\mathbf{ttraces}(i) \subseteq \mathbf{ttraces}(s)$



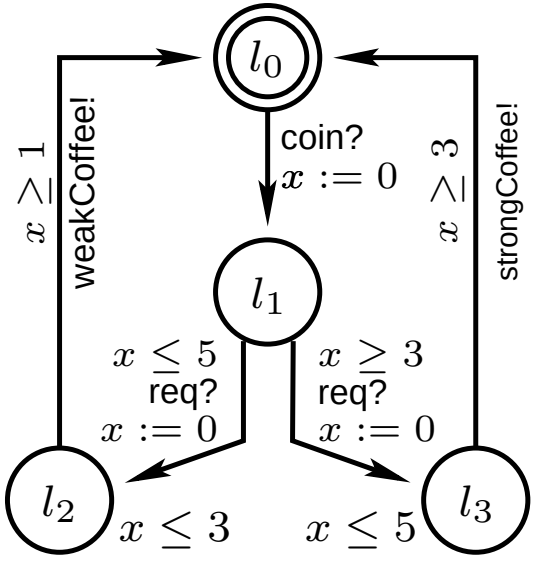
- no illegal output is produced and
- required output is observed (at right time):

$i \,\mathbf{rt\text{-}ioco}\, s =_{def}$
$\forall \sigma \in \mathbf{ttraces}(s) \,.out(i \,\mathbf{after}\, \sigma) \subseteq out(s \,\mathbf{after}\, \sigma)$

$i \,\mathbf{rt\text{-}ioco}\, s \stackrel{?}{\Leftrightarrow} \mathbf{ttraces}(i) \subseteq \mathbf{ttraces}(s)$

Implementation Relation: rt-ioco

- Timed trace eg.: $\sigma = coin? \cdot 5 \cdot req? \cdot 2 \cdot weakCoffee! \cdot 9 \cdot coin?$
- $\mathbf{ttraces}(s)$ – set of *timed traces* from s: $\{\sigma \in (A \cup \mathbb{R}_+)^* \mid s \stackrel{\sigma}{\Rightarrow}\}$
- Timed trace *inclusion* as conf. relation: $\mathbf{ttraces}(i) \subseteq \mathbf{ttraces}(s)$



- no illegal output is produced and
- required output is observed (at right time):

$i \,\mathbf{rt\text{-}ioco}\, s =_{def}$
$\forall \sigma \in \mathbf{ttraces}(s) \,.\, out(i \,\mathbf{after}\, \sigma) \subseteq out(s \,\mathbf{after}\, \sigma)$

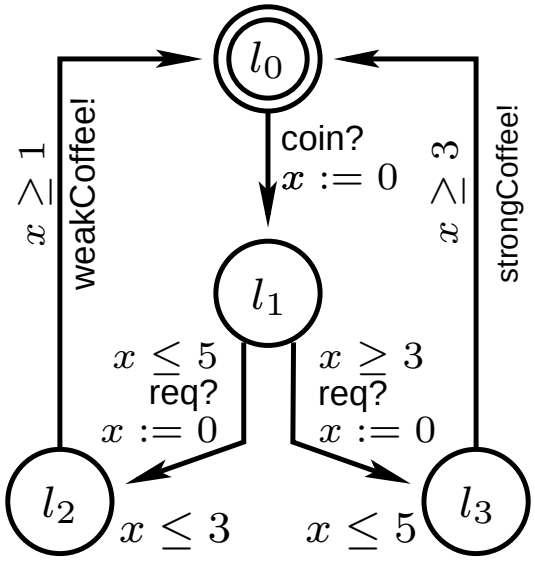
$i \,\mathbf{rt\text{-}ioco}\, s \stackrel{?}{\Leftrightarrow} \mathbf{ttraces}(i) \subseteq \mathbf{ttraces}(s)$

- $\Rightarrow$ Reachability algorithms:
  - $i \,\mathbf{after}\, \sigma$ – set of states reachable after σ
  - $out(P)$ – possible outputs from state set P:

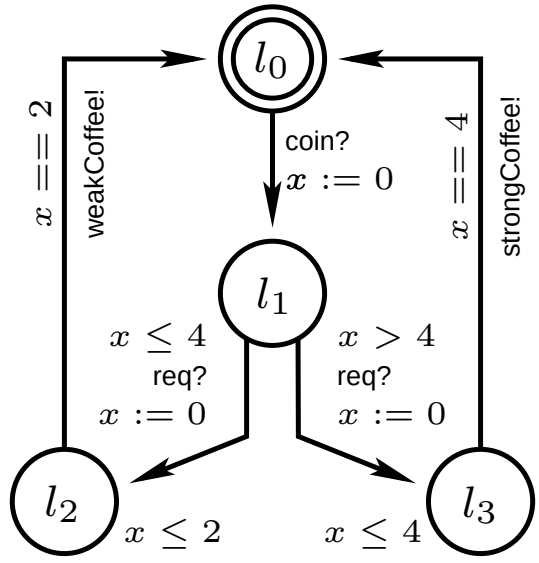
$out(P) =_{def} \bigcup \{\, \alpha \in A_{out} \mid p \in P.\, p \stackrel{\alpha}{\rightarrow} \}$

Implementation Relation Example

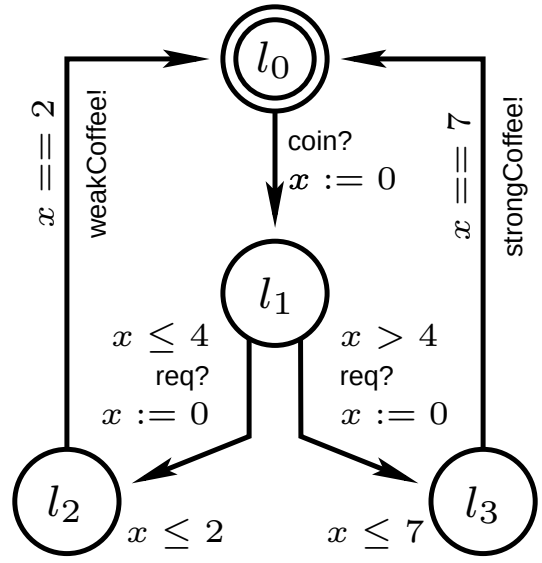
Specification



Implementation 1



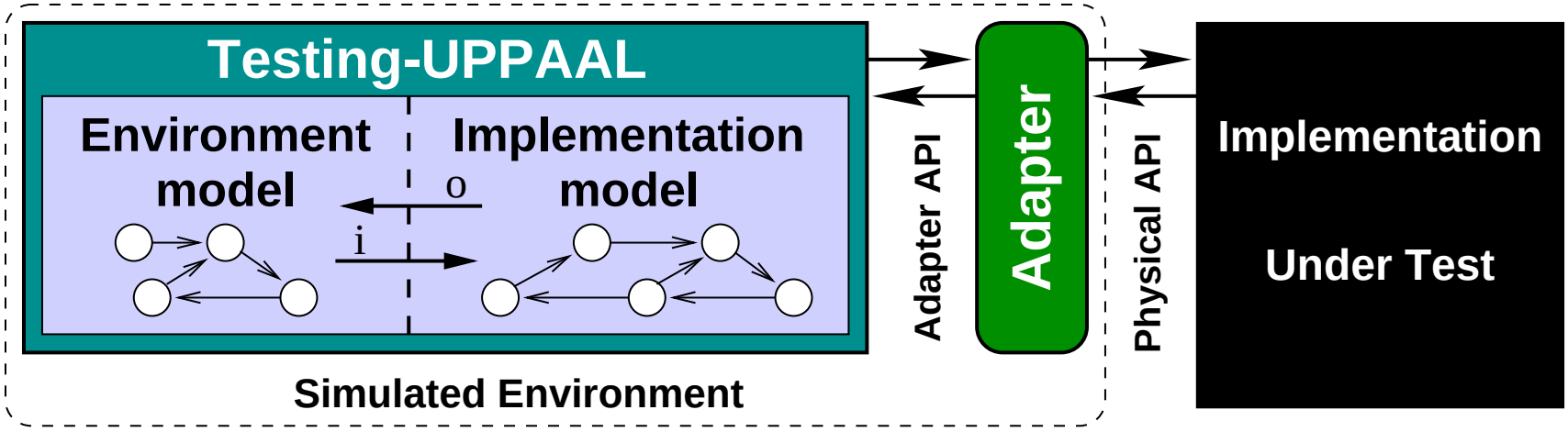
Implementation 2



Trace, σ	$out(i\mathbf{after}\sigma)$	allowed?
2	$\emptyset$	Yes
$c? \cdot 2$	$\emptyset$	Yes
$c? \cdot 2 \cdot r? \cdot 2$	weakC	Yes
$c? \cdot 5 \cdot r? \cdot 4$	strongC	Yes

Trace, σ	$out(i\mathbf{after}\sigma)$	allowed?
2	$\emptyset$	Yes
$c? \cdot 2$	$\emptyset$	Yes
$c? \cdot 2 \cdot r? \cdot 2$	weakC	Yes
$c? \cdot 5 \cdot r? \cdot 7$	strongC	No

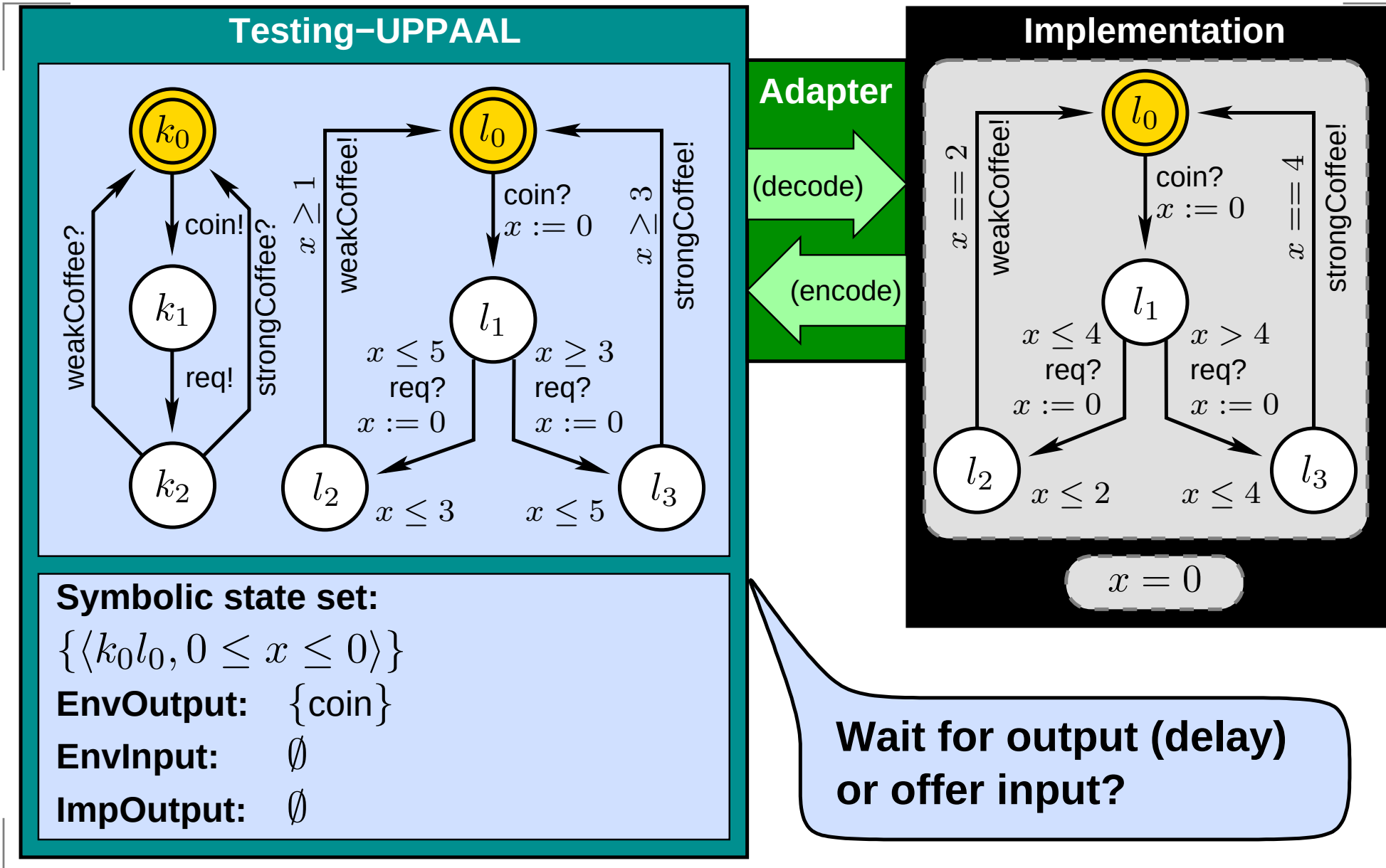
Test Setup



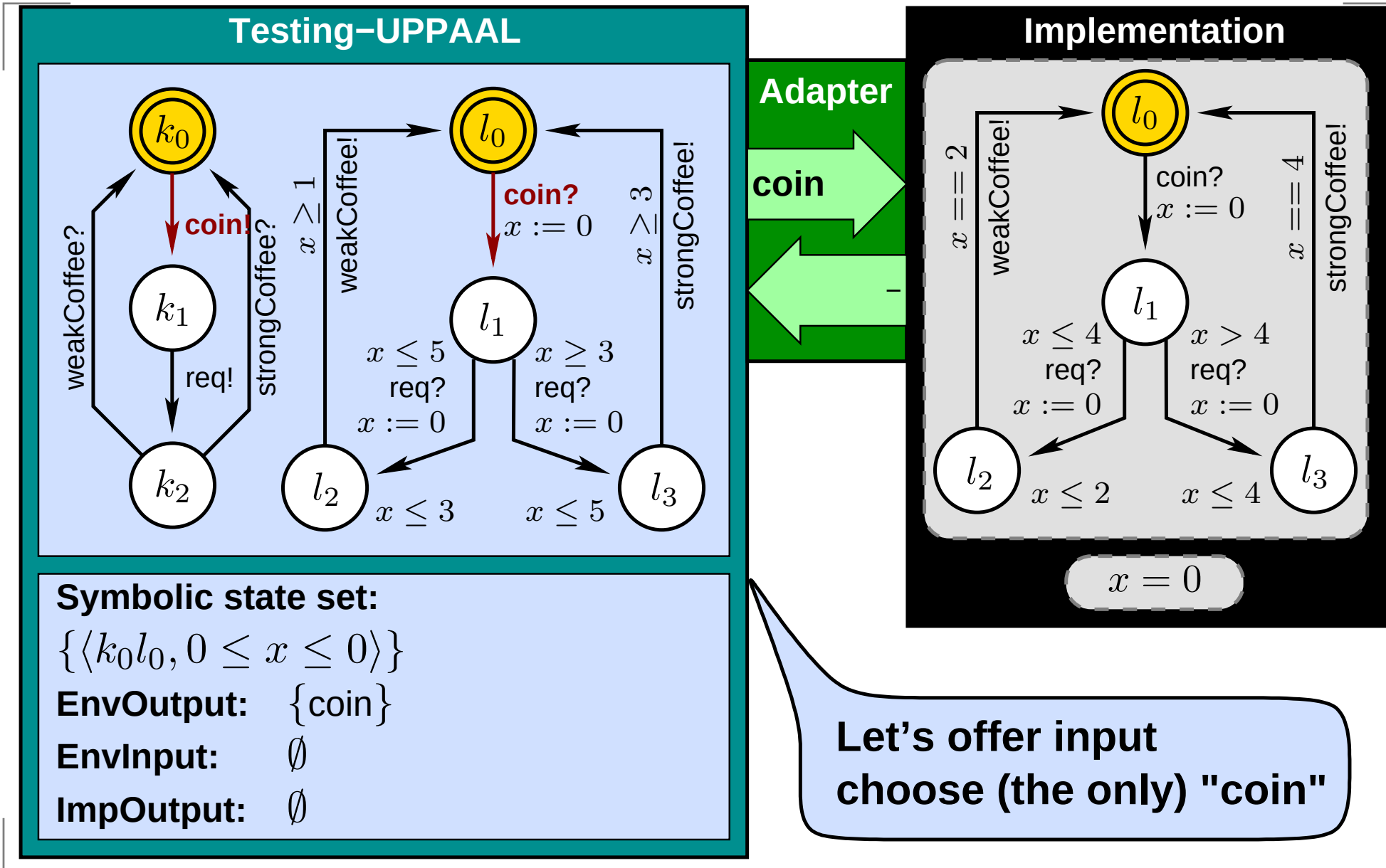
Test specification supplied by user:

- IUT model which is *(weakly) input enabled*.
- Model of *Environment* (not necessarily input enabled).
- $\Rightarrow$ *Closed Network* partitioned into *Env* and *IUT*.
- Designate *observable input* and *output* actions.
- Associate data-variables for *value passing* (future work).
- Specify amount of real time per one time-unit in model.

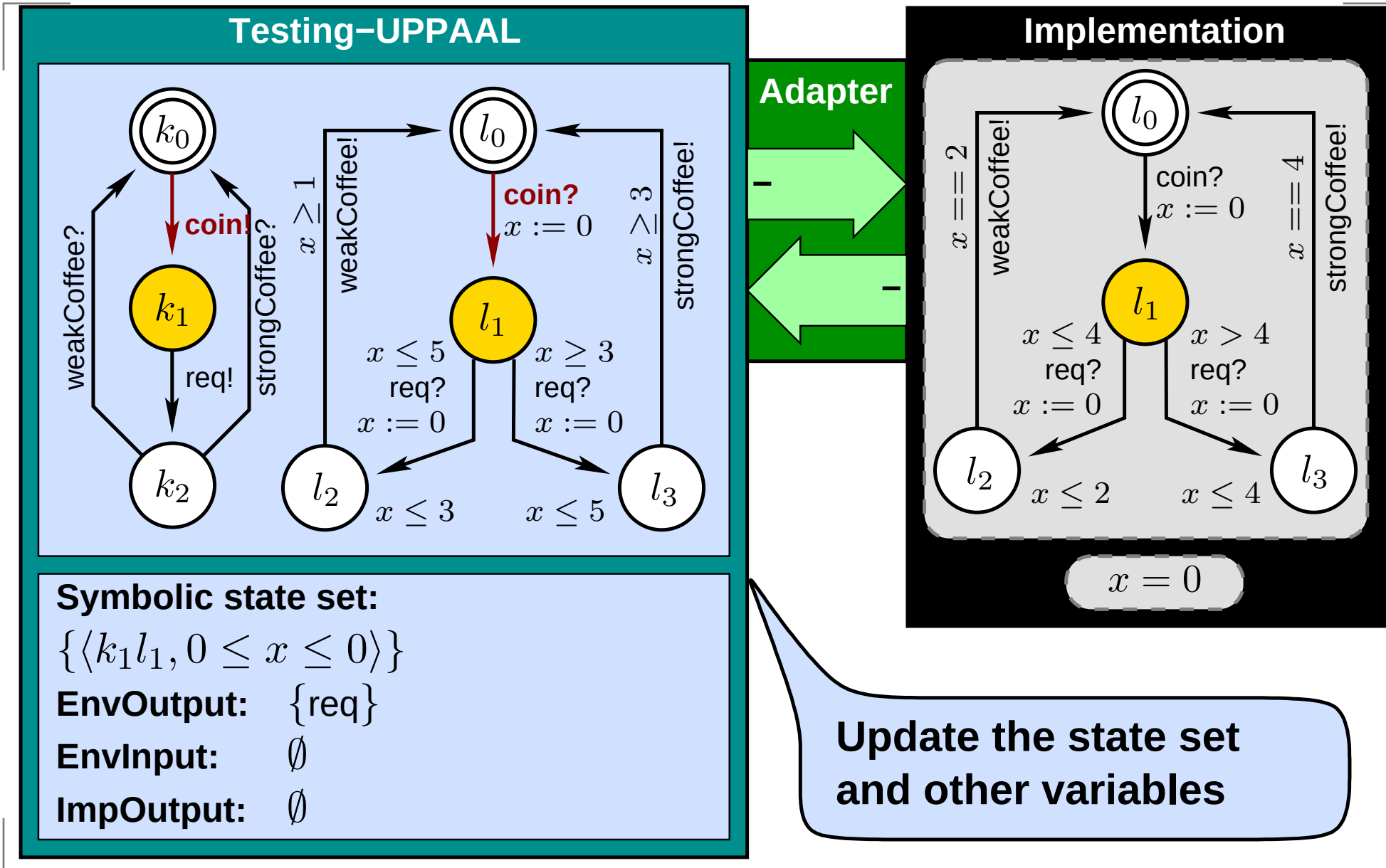
Testing Online



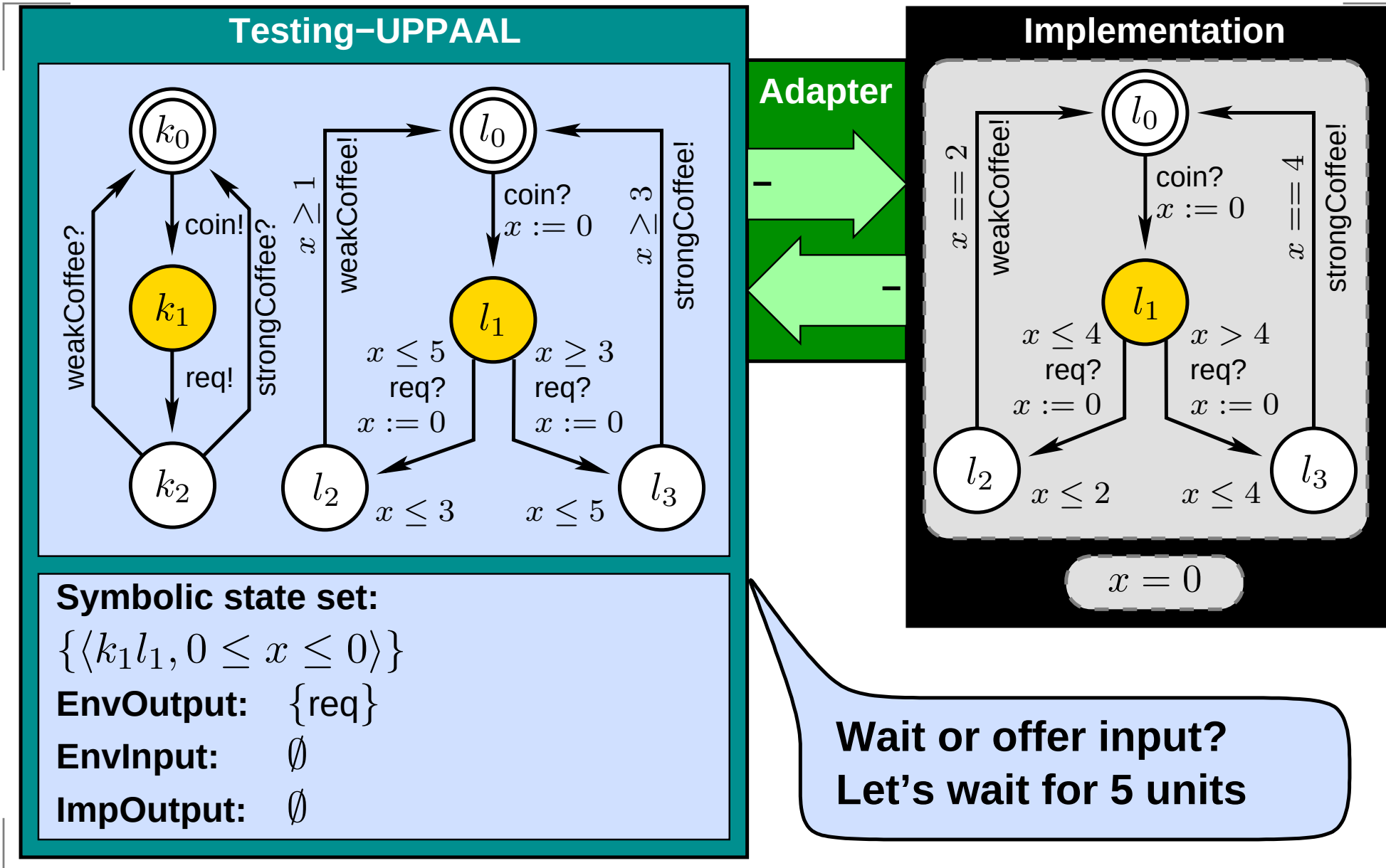
Testing Online



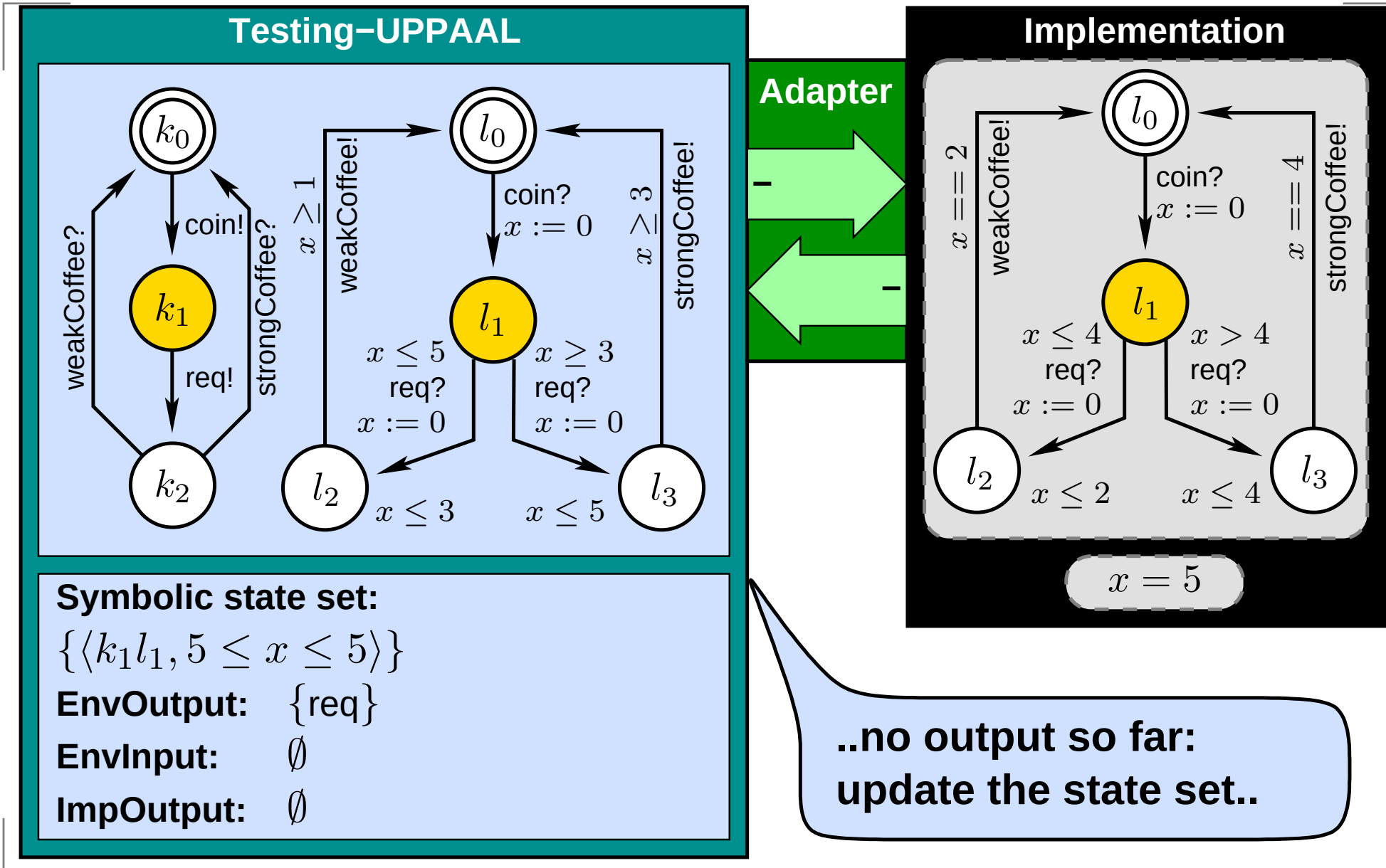
Testing Online



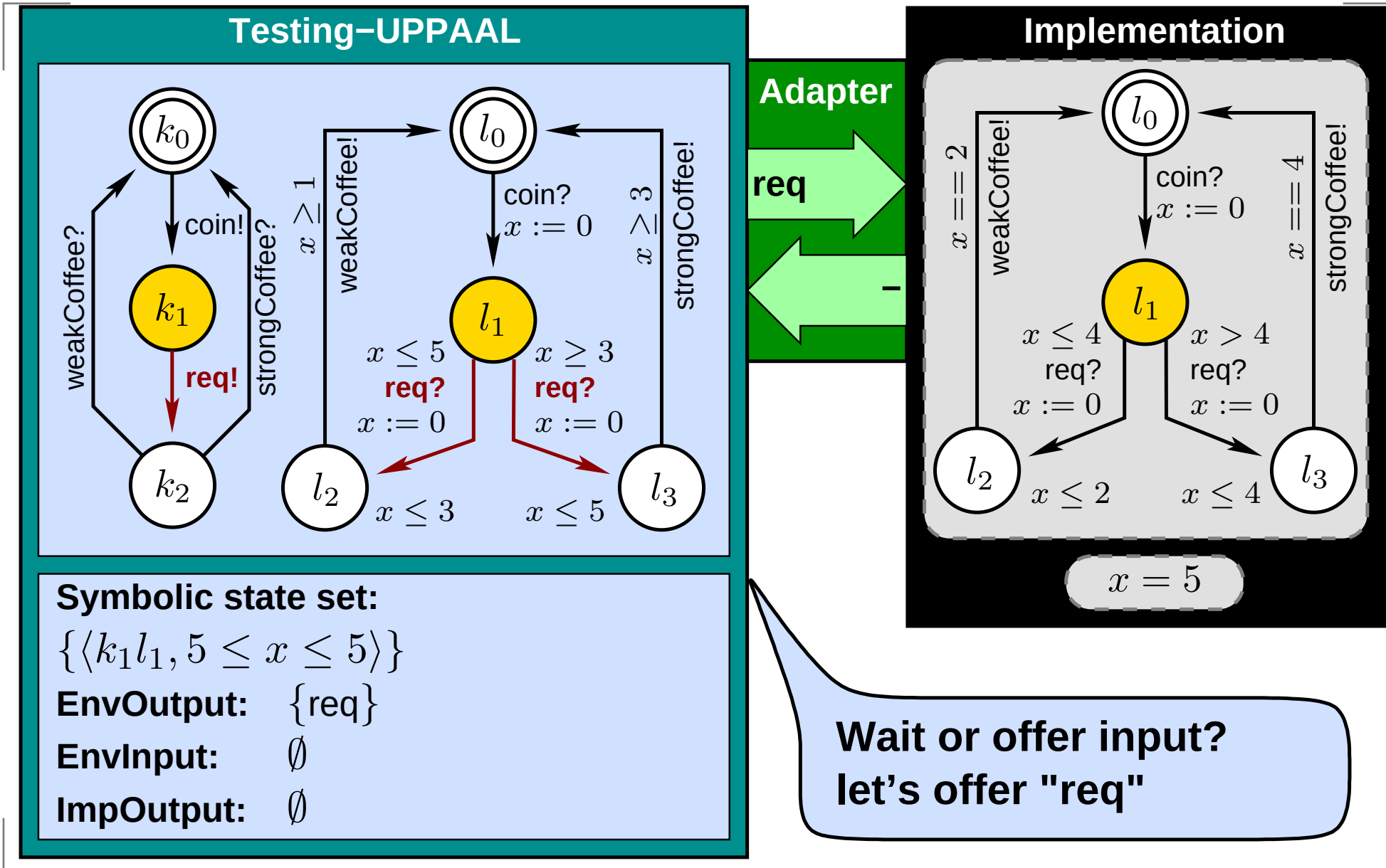
Testing Online



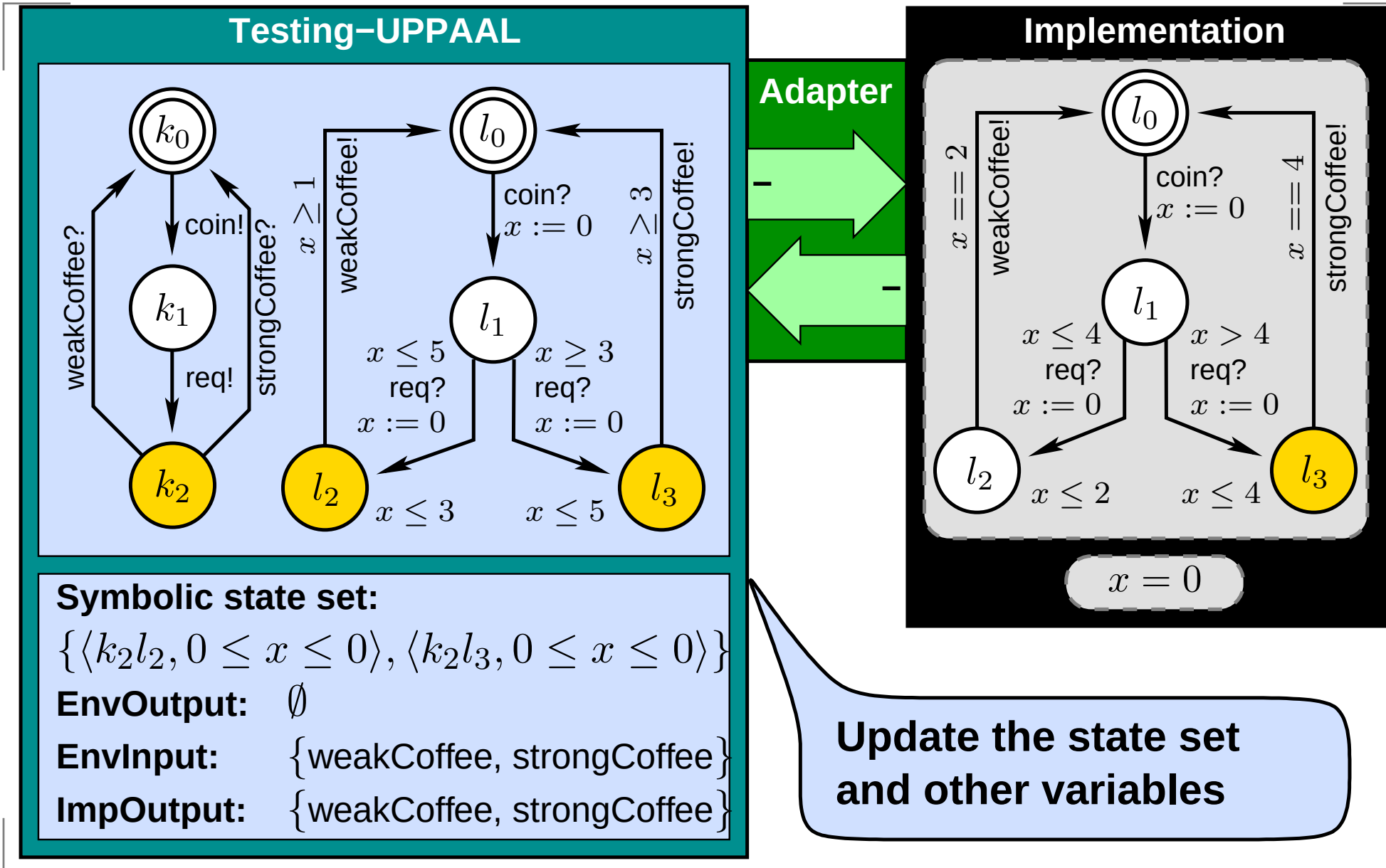
Testing Online



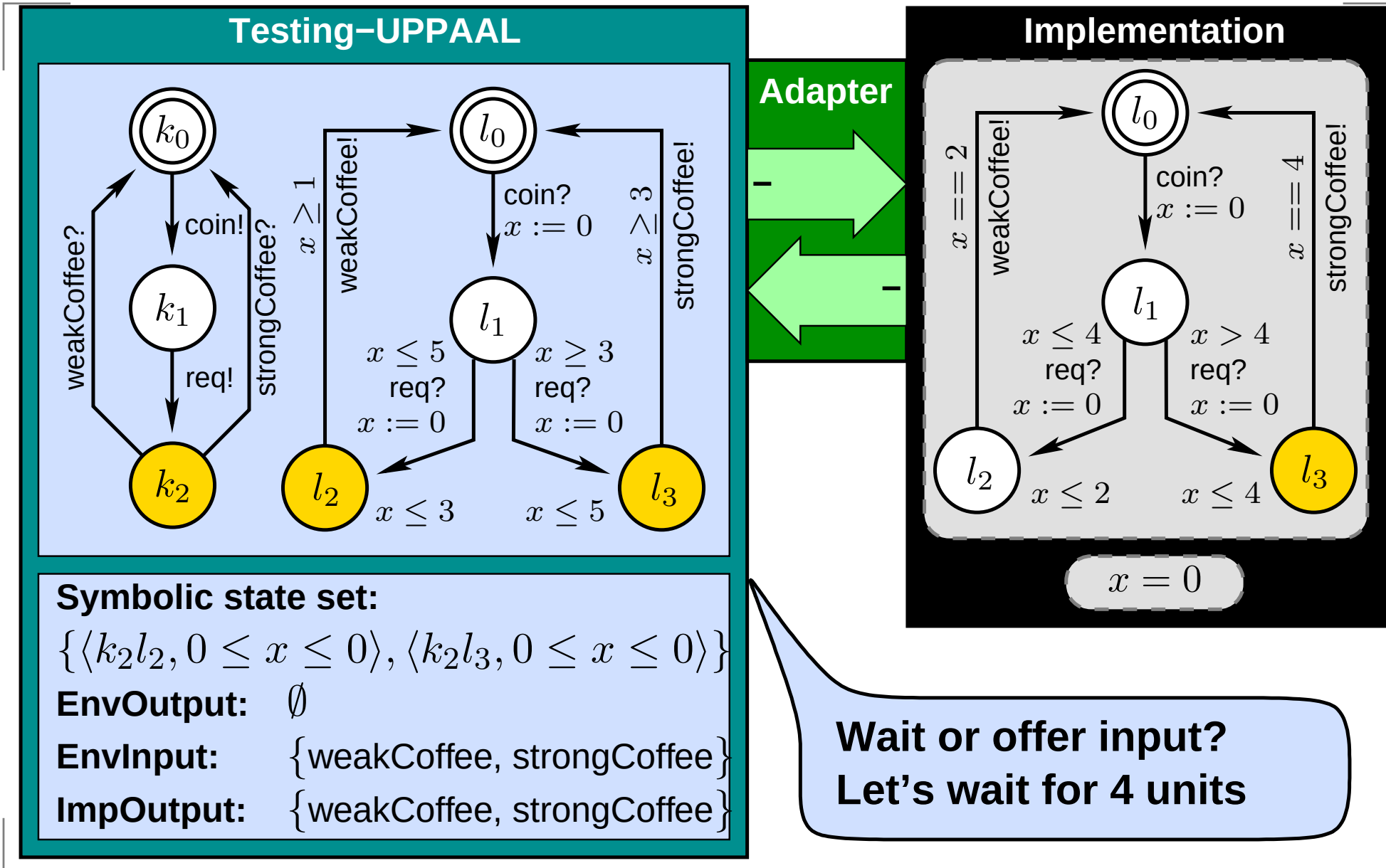
Testing Online



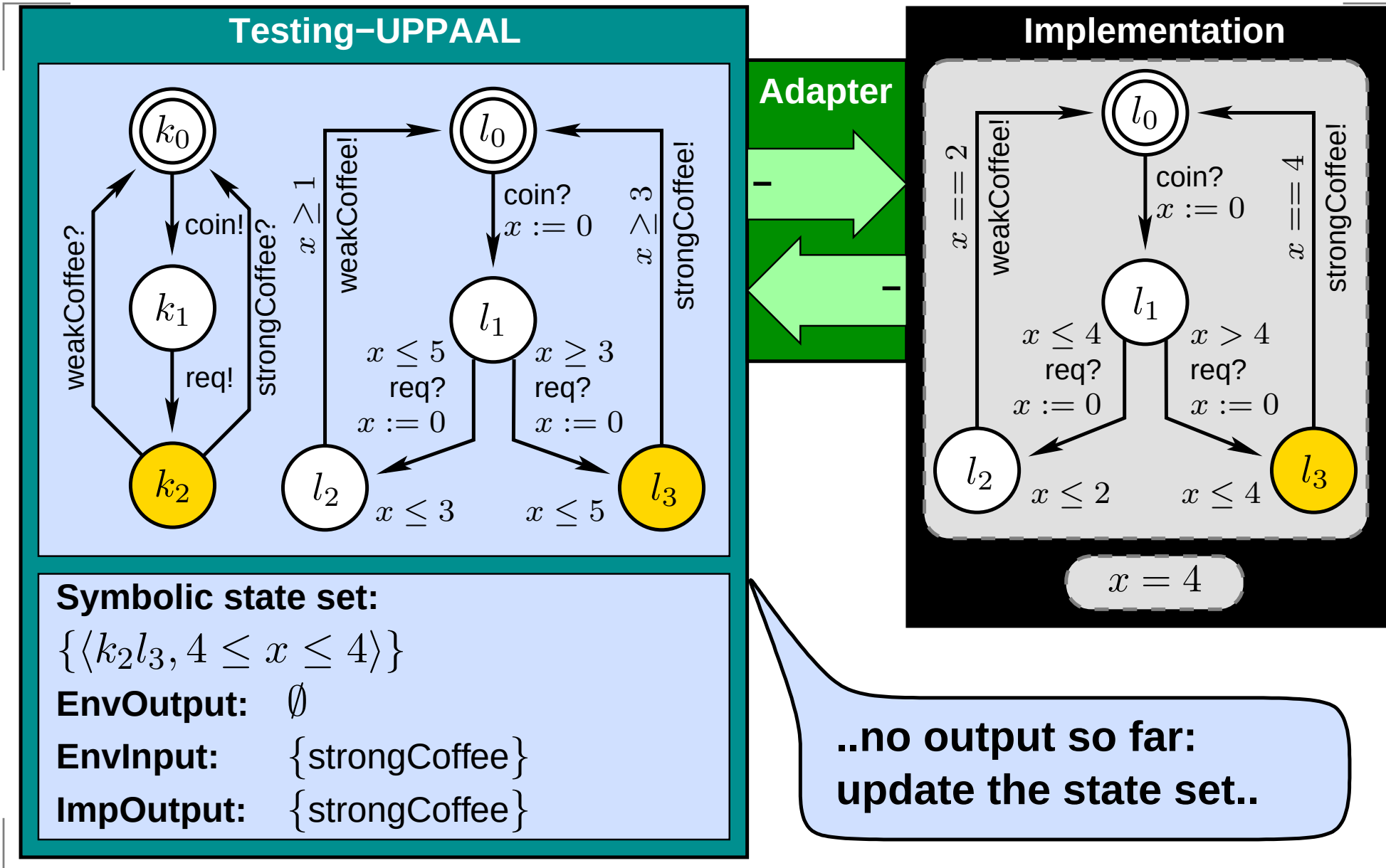
Testing Online



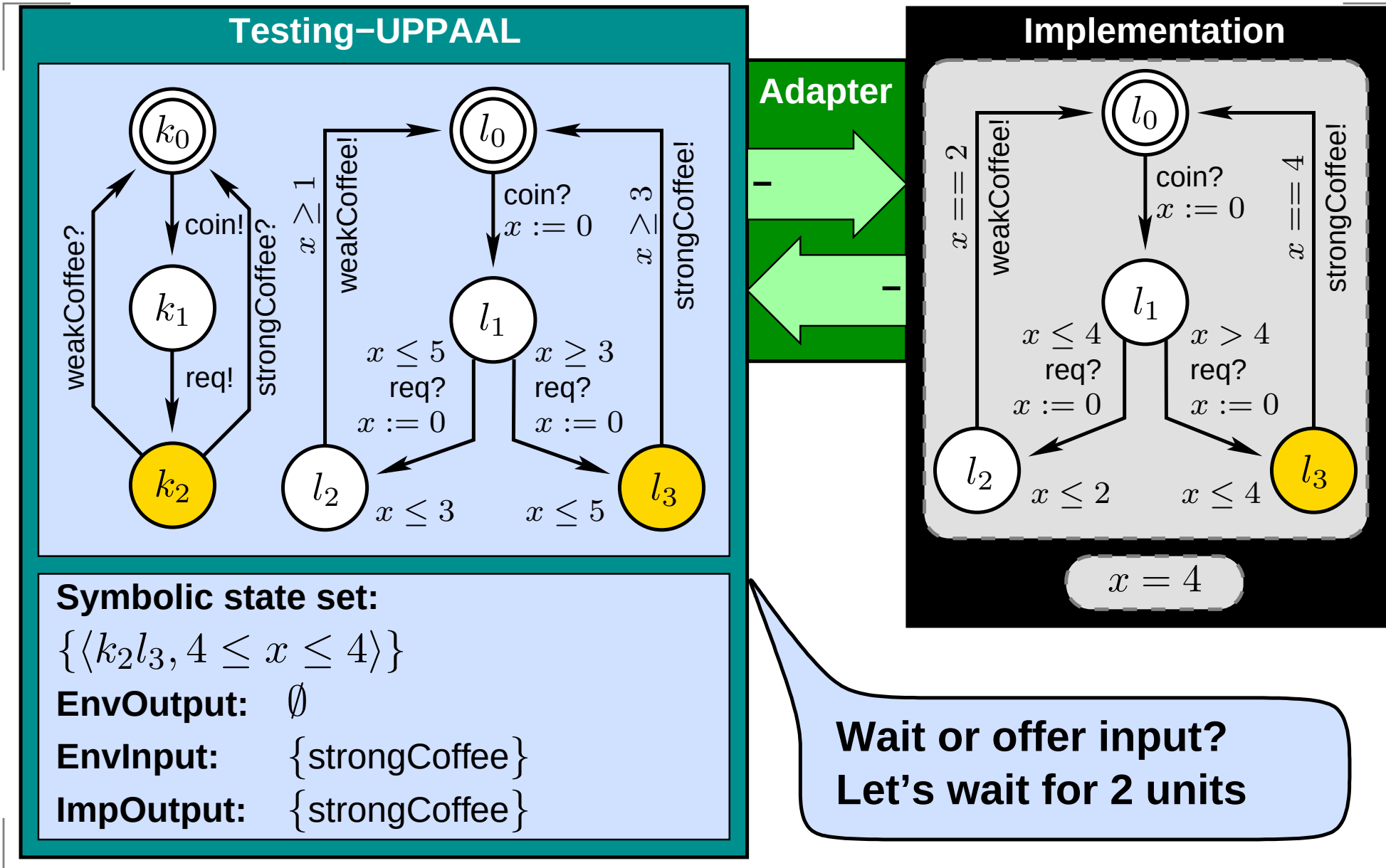
Testing Online



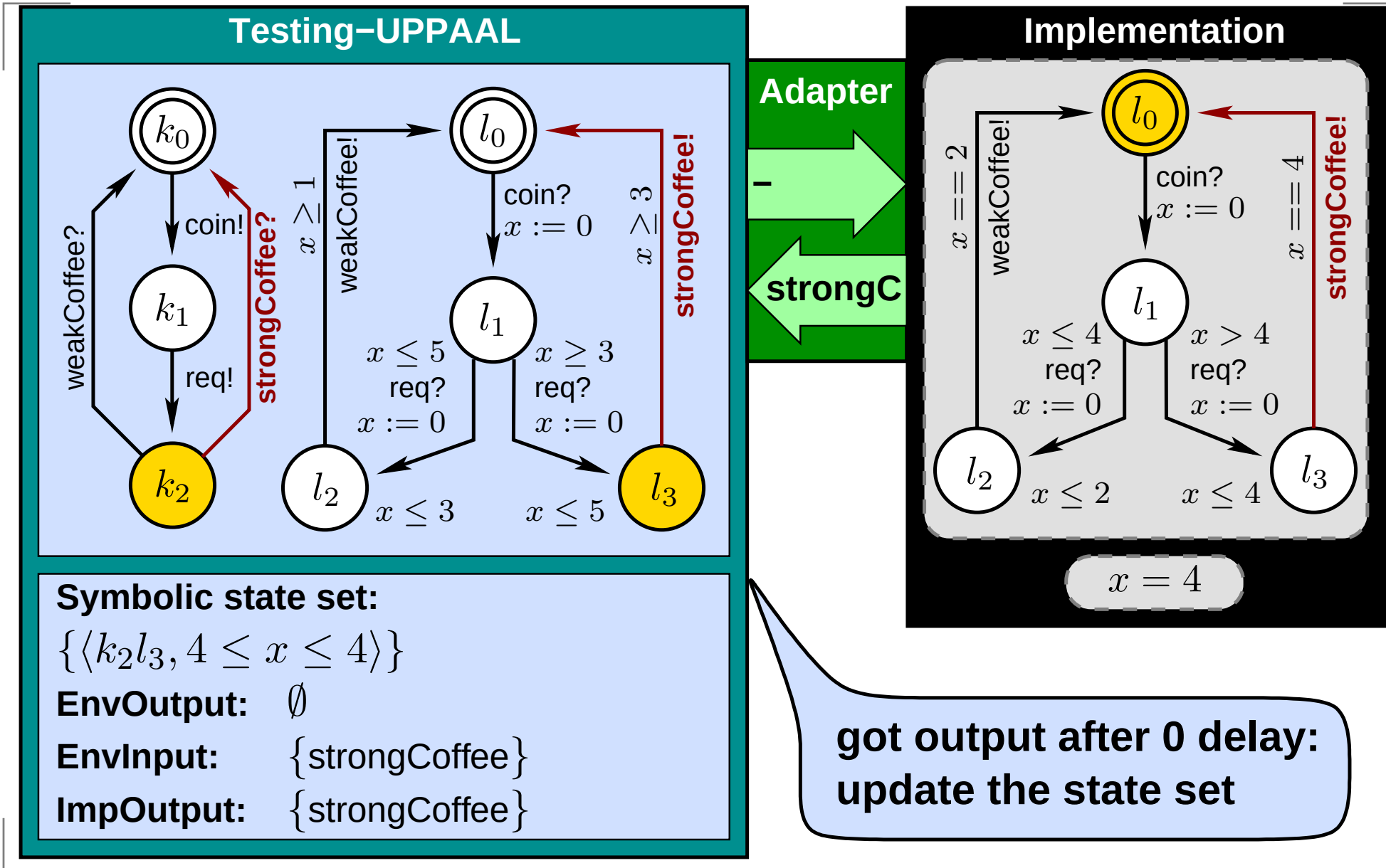
Testing Online



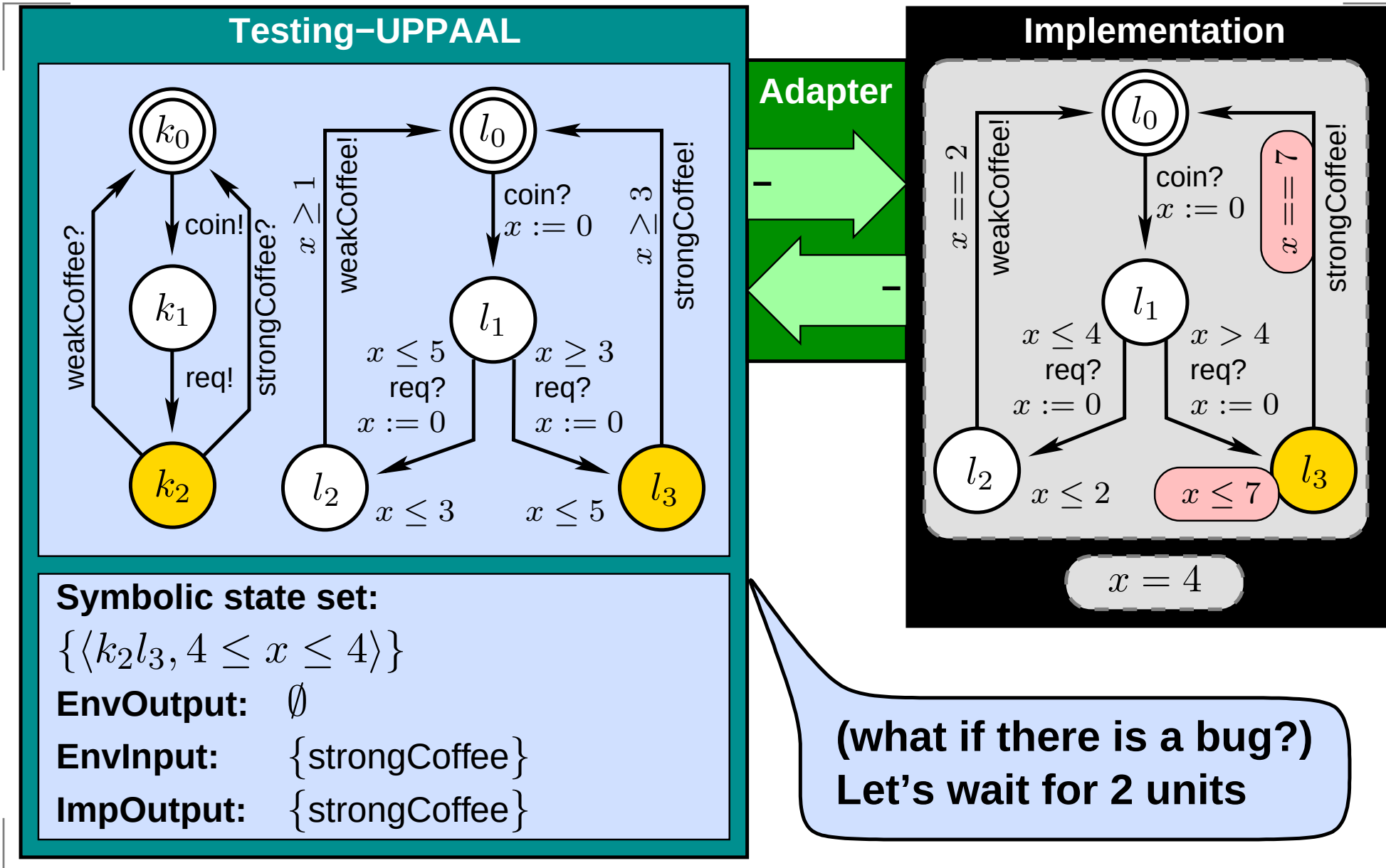
Testing Online



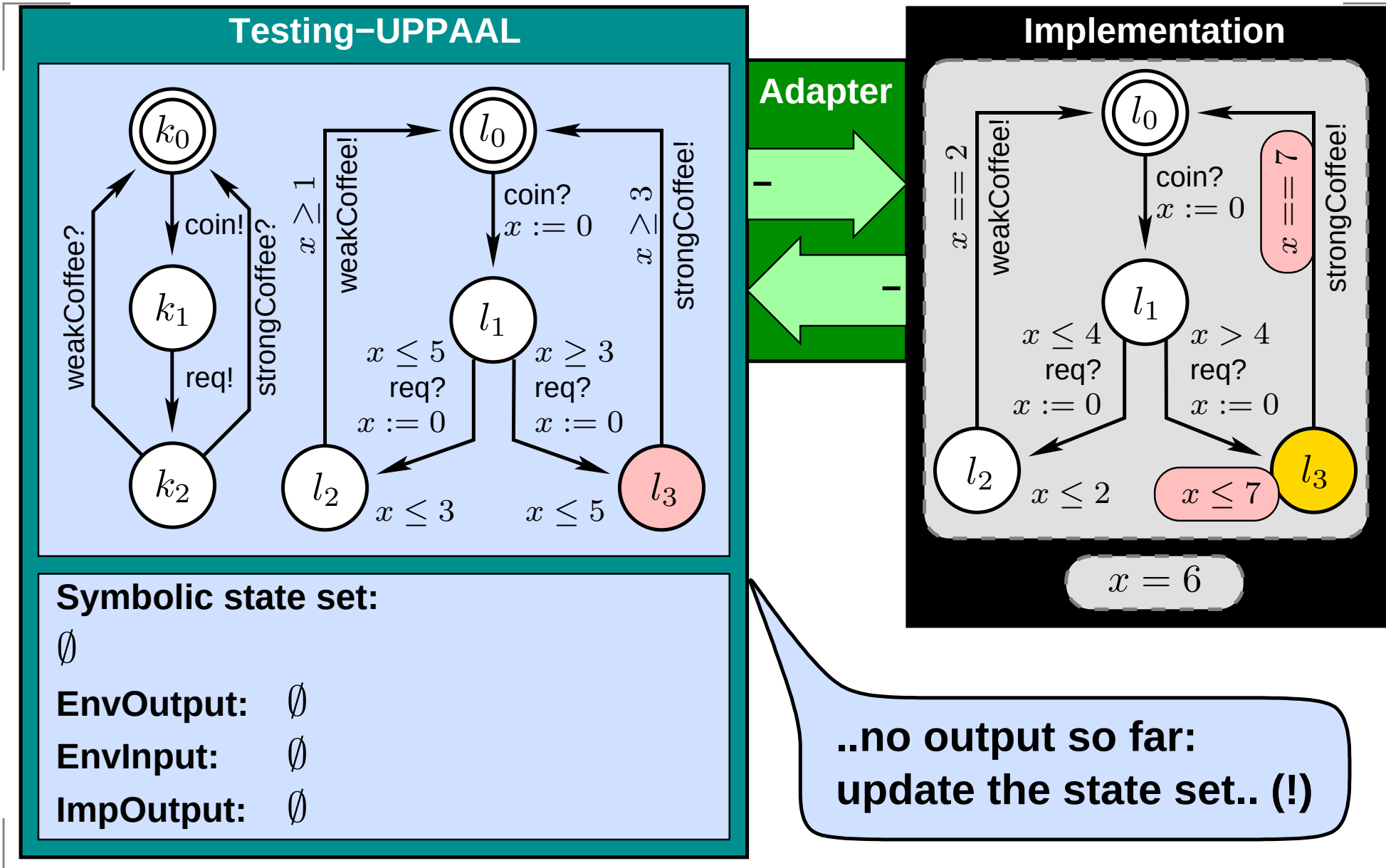
Testing Online



Testing Online



Testing Online



Test Generation and Execution

while not $timeout$ **do** choose randomly:

action: `// offer an input`

...

wait: `// wait for an output`

...

return *fail*

...

return *inconclusive*

...

loop

return *pass*

- Terminate on $timeout$
- Two choices
- Pass on timeout if $Z \neq \emptyset$

Test Generation and Execution

action: `// offer an input`

$a := ChooseAction(EnvOutput(Z))$

send a to implementation

$Z := After(Z, a)$

- Terminate on $timeout$
- Two choices
- Pass on timeout if $Z \neq \emptyset$
- $EnvOutput$,
- $ChooseAction$,
- $After(action)$,

Test Generation and Execution

wait: `// wait for an output`

$\delta := ChooseDelay(Z)$

sleep for δ time units, wake up on output o

if o occurs at $\delta' \leq \delta$ **then**

$Z := After(Z, \delta')$

if $o \notin ImpOutput(Z)$ **then return** *fail*

else if $o \notin EnvInput(Z)$ **then return** *inc*

else $Z := After(Z, o)$

else `// no output within` δ `delay`

$Z := After(Z, \delta)$

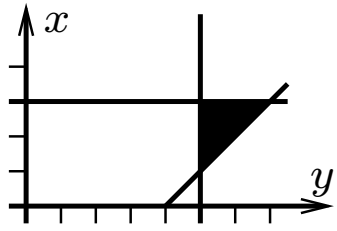
if $Z = \emptyset$ **then return** *fail*

- Terminate on $timeout$
- Two choices
- Pass on timeout if $Z \neq \emptyset$
- $EnvOutput$,
- $ChooseAction$,
- $After(action)$,
- $ChooseDelay$,
- $EnvInput$,
- $ImpOutput$,
- $After(delay)$

Symbolic Techniques in UPPAAL

- *Zone* is a conjunction of clock constraints of the form: $\{x_i - x_j \prec c_{ij}\} \cup \{a_i \prec x_i\} \cup \{x_j \prec b_j\}$ where $\prec \in \{\leq, \leq\}$

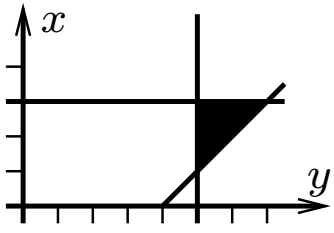
$$z = [(y - x \leq 4) \wedge (y \geq 5) \wedge (x \leq 3)]$$



Symbolic Techniques in UPPAAL

- *Zone* is a conjunction of clock constraints of the form: $\{x_i - x_j \prec c_{ij}\} \cup \{a_i \prec x_i\} \cup \{x_j \prec b_j\}$ where $\prec \in \{\leq, \leq\}$
- *Difference bound matrix* - compact representation.

$$z = [(y - x \leq 4) \wedge (y \geq 5) \wedge (x \leq 3)]$$

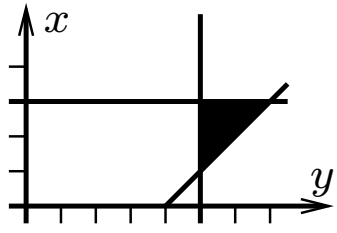


	0	y	x
0	–	-5	0
y	∞	–	4
x	3	∞	–

Symbolic Techniques in UPPAAL

- *Zone* is a conjunction of clock constraints of the form: $\{x_i - x_j \prec c_{ij}\} \cup \{a_i \prec x_i\} \cup \{x_j \prec b_j\}$ where $\prec \in \{\leq, \leq\}$
- *Difference bound matrix* - compact representation.
- Symbolic state set $Z = \{\langle \bar{l}_1, z_1 \rangle, \ldots, \langle \bar{l}_n, z_n \rangle\}$

$$z = [(y - x \leq 4) \wedge (y \geq 5) \wedge (x \leq 3)]$$

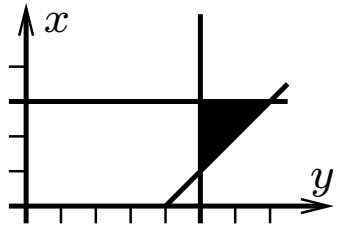


	0	y	x
0	–	-5	0
y	∞	–	4
x	3	∞	–

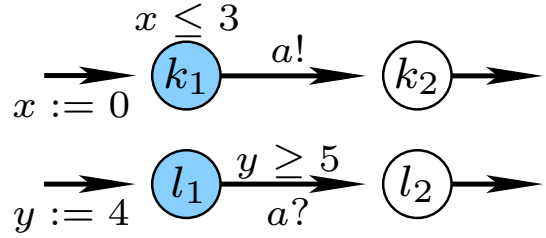
Symbolic Techniques in UPPAAL

- *Zone* is a conjunction of clock constraints of the form: $\{x_i - x_j \prec c_{ij}\} \cup \{a_i \prec x_i\} \cup \{x_j \prec b_j\}$ where $\prec \in \{\leq, \leq\}$
- *Difference bound matrix* - compact representation.
- Symbolic state set $Z = \{\langle \bar{l}_1, z_1 \rangle, \ldots, \langle \bar{l}_n, z_n \rangle\}$
- *Action* transition: $\langle \bar{l}, z \rangle \xrightarrow{a} \langle \bar{l}', (z \wedge g)_r \wedge I(\bar{l}') \rangle$: $l \xrightarrow{g,a,r} l'$ is a-action transition and $z \wedge g \neq \emptyset$,$(z \wedge g)_r \wedge I(\bar{l}') \neq \emptyset$.

$$z = [(y - x \leq 4) \wedge (y \geq 5) \wedge (x \leq 3)]$$



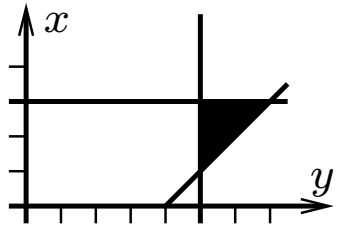
	0	y	x
0	–	-5	0
y	∞	–	4
x	3	∞	–



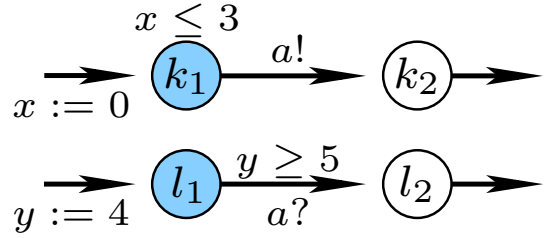
Symbolic Techniques in UPPAAL

- *Zone* is a conjunction of clock constraints of the form: $\{x_i - x_j \prec c_{ij}\} \cup \{a_i \prec x_i\} \cup \{x_j \prec b_j\}$ where $\prec \in \{\leq, \leq\}$
- *Difference bound matrix* - compact representation.
- Symbolic state set $Z = \{\langle \bar{l}_1, z_1 \rangle, \ldots, \langle \bar{l}_n, z_n \rangle\}$
- *Action* transition: $\langle \bar{l}, z \rangle \xrightarrow{a} \langle \bar{l}', (z \wedge g)_r \wedge I(\bar{l}') \rangle$: $l \xrightarrow{g,a,r} l'$ is a-action transition and $z \wedge g \neq \emptyset$,$(z \wedge g)_r \wedge I(\bar{l}') \neq \emptyset$.
- *Delay* transition: $\langle \bar{l}, z \rangle \xrightarrow{\delta} \langle \bar{l}, z^{+\delta} \wedge I(\bar{l}) \rangle$ iff $z^{+\delta} \wedge I(\bar{l}) \neq \emptyset$.

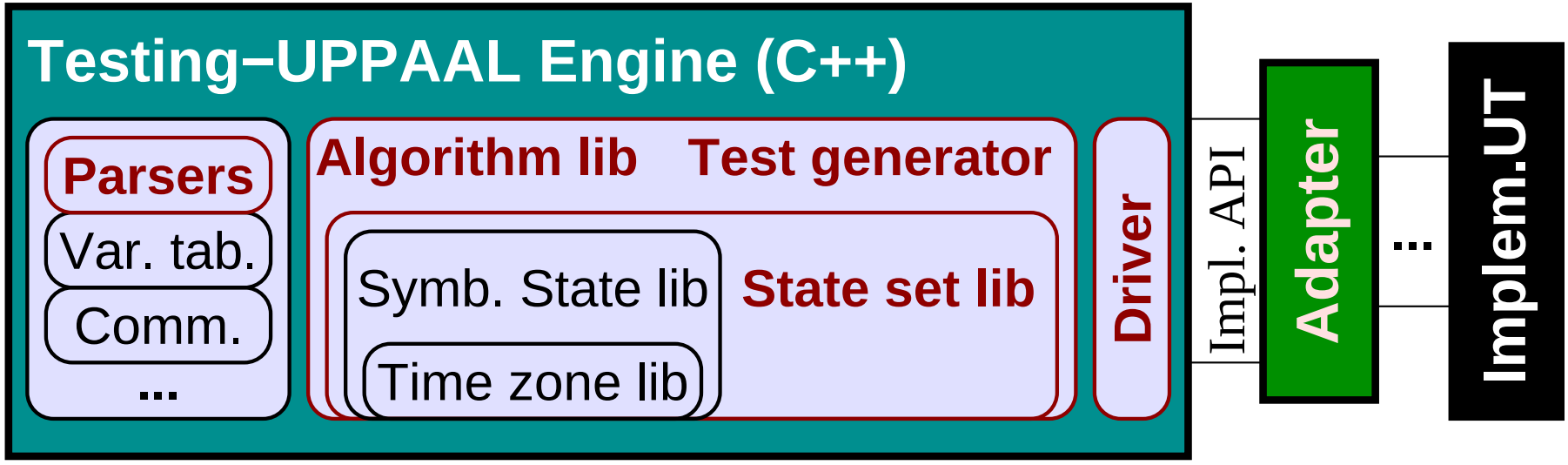
$$z = [(y - x \leq 4) \wedge (y \geq 5) \wedge (x \leq 3)]$$



	0	y	x
0	–	-5	0
y	∞	–	4
x	3	∞	–



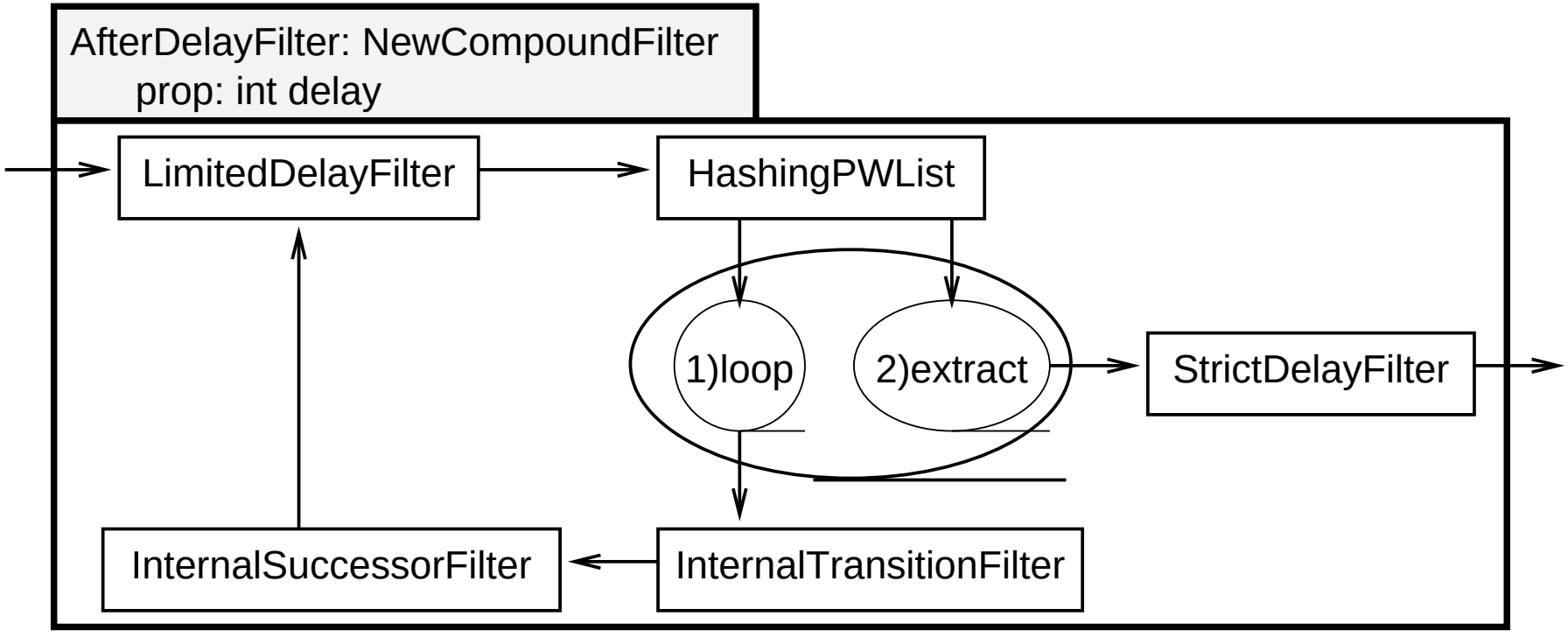
T-UPPAAL Architecture



- Adopted parsers for test specification extensions.
- Extended and reused symb. state operation algorithms.
- Test generation and execution algorithm added.
- Driver for event time-stamping and extracting.
- Simple method call and call-back interface for adapter.

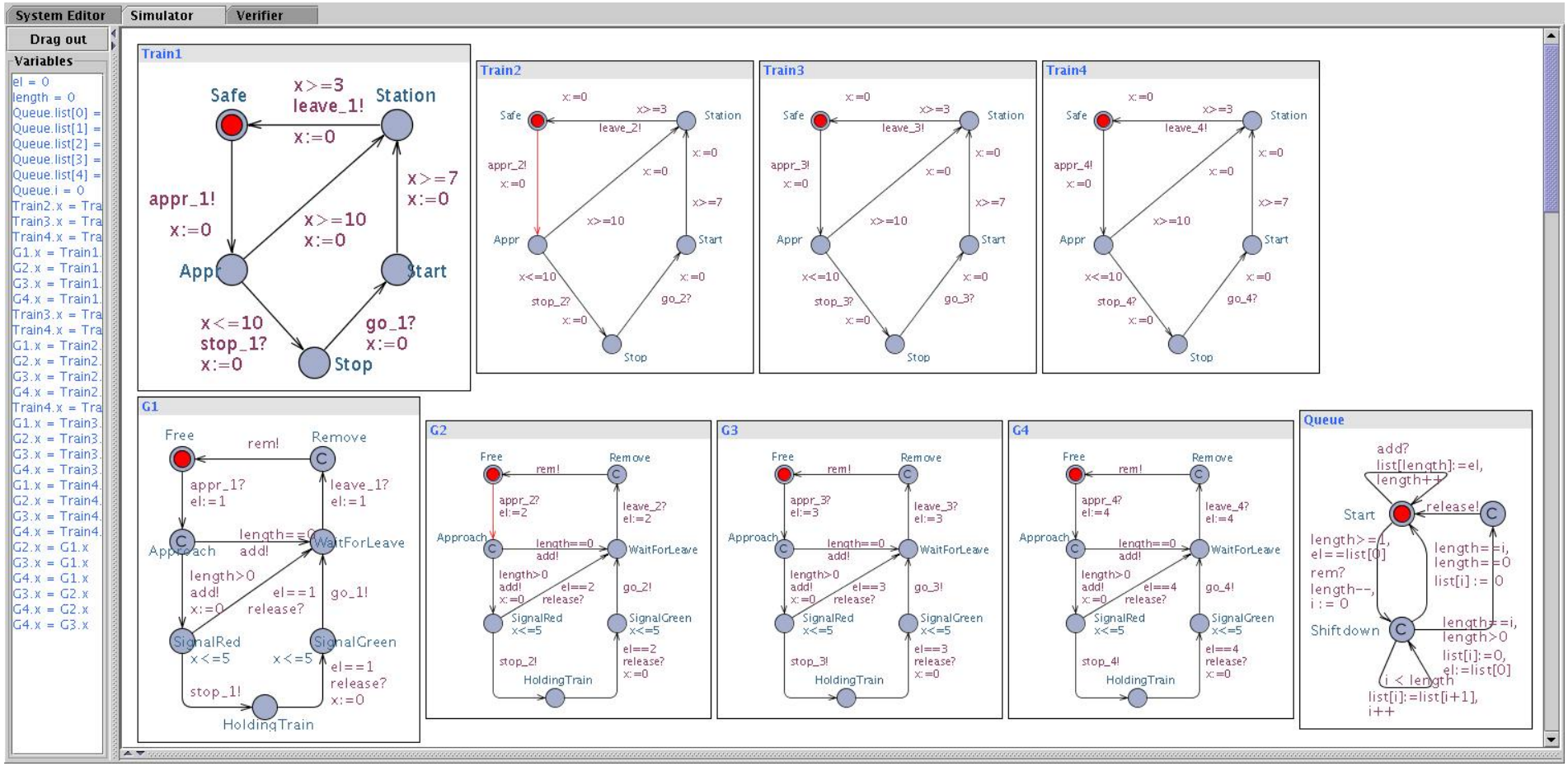
T-UPPAAL Implementation: Pipelines

- Reachability algorithms for *afterDelay* and *afterAction*.



T-UPPAAL Applications

- Mouse buttons (non-determinism $\Rightarrow$ explosion).
- Train gate - more complicated, benchmarks.

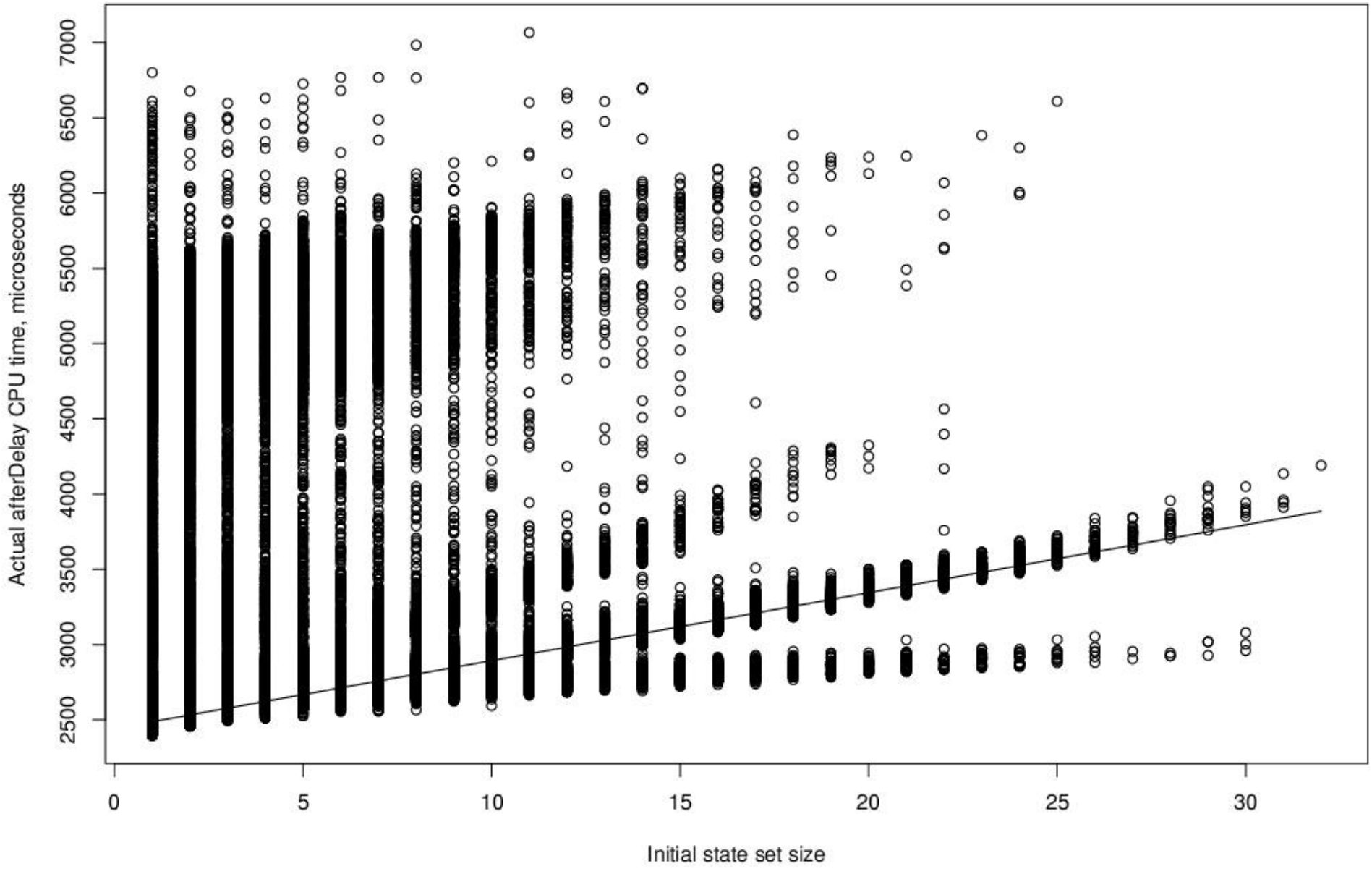


- Light controller - interactive demo.

Train-Gate: Error Detection Capability

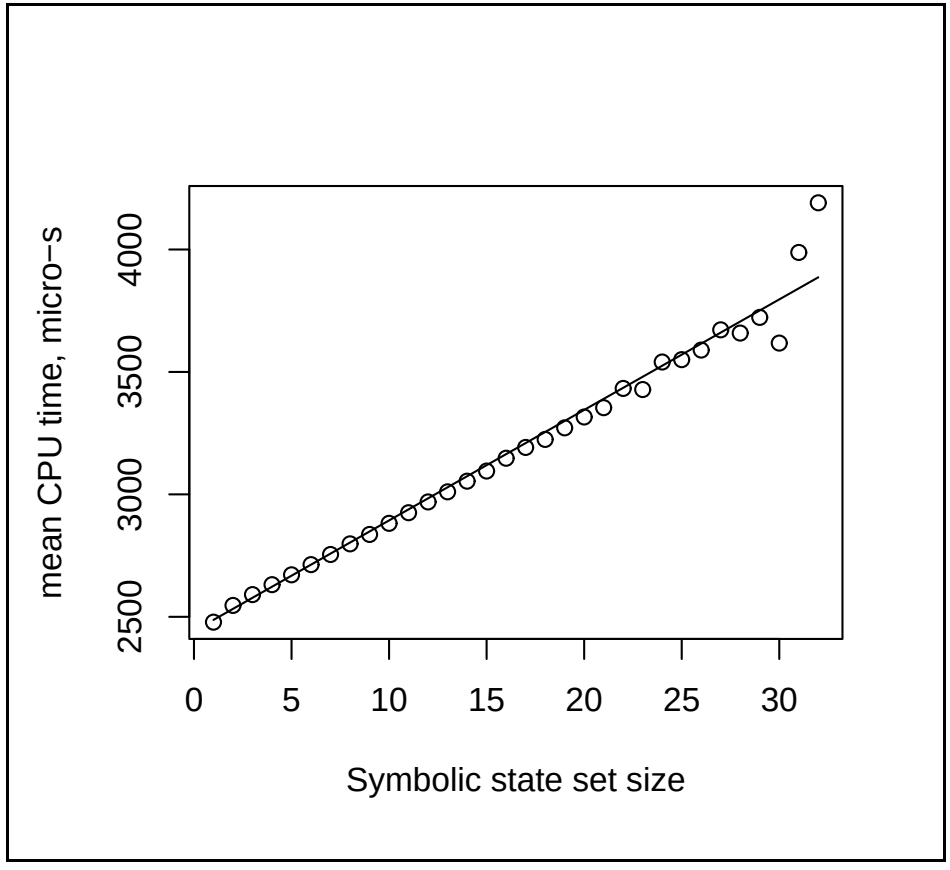
Mu-	Number of verdicts				Input actions			Duration (time units)		
tant	Pass	Fail	Inc	xxx	Min	Avg	Max	Min	Avg	Max
M1	0	1100	0	0	2	5.0	18	6	72.7	359
M2	0	1099	0	1	2	4.6	12	3	66.7	370
M3	0	1100	0	0	2	4.8	12	6	80.2	389
M4	0	1100	0	0	6	8.6	22	37	163.4	641
M5	0	1099	0	1	4	5.7	14	17	92.0	435
M6	0	1100	0	0	2	3.9	14	6	62.8	349
M0	1077	3	10	10	99	376.0	442	2408	9951.1	10000

Computing Performance (instances)

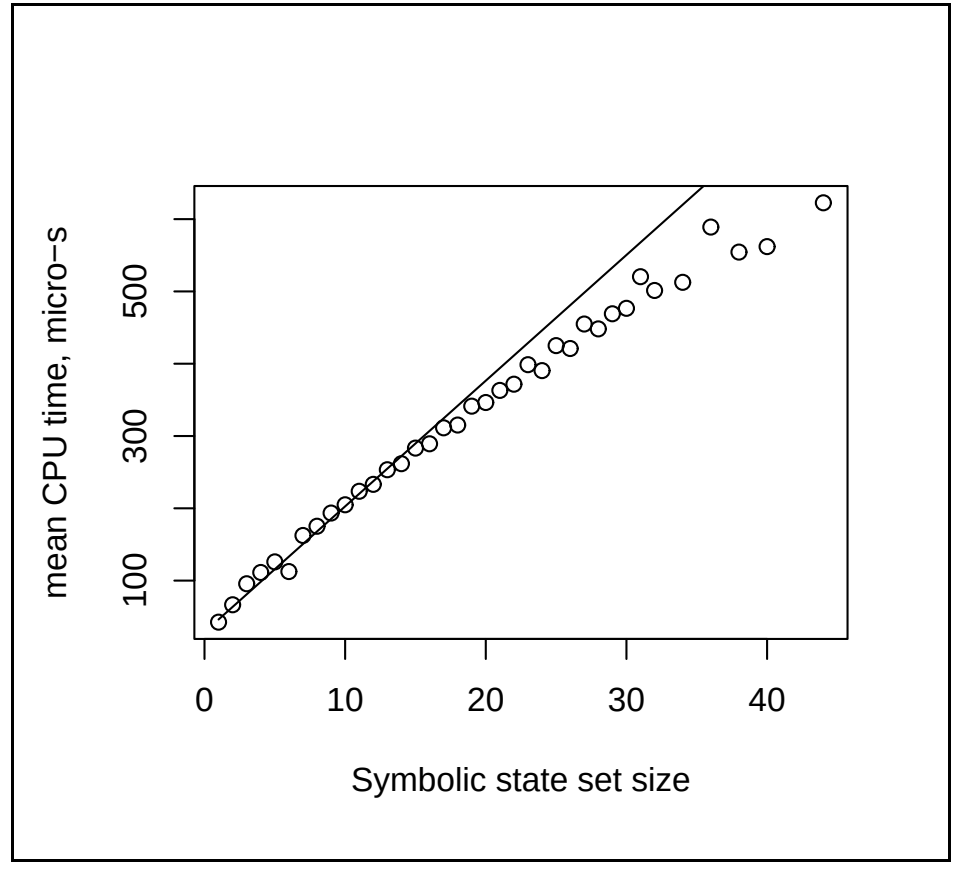


Computing Performance (means)

after delay



after action



Computing Performance (summary)

Mu-tant	Execution time, μs				Symb. state set size, #			
	after(delay)		*after*(action)		*after*(delay)		*after*(action)	
	avg	max	avg	max	avg	max	avg	max
M1	2720.8	6739.3	123.3	762.2	2.31	17	2.65	34
M2	2783.9	6744.6	131.0	759.9	2.38	19	2.76	30
M3	2770.9	6640.2	125.9	755.5	2.38	20	2.68	30
M4	2696.1	6666.1	106.5	750.2	2.91	31	3.04	36
M5	2771.0	6830.4	129.6	731.2	2.94	31	3.26	32
M6	2814.6	6660.7	130.2	810.3	2.07	16	2.50	32
M0	2573.8	7066.6	78.0	722.4	2.91	32	2.83	44

Future Work

- New UPPAAL features (broadcast, U-Code, ...)
- Termination of testing.
- Time synchronization: uncertainty in clock values.
- Advanced choices in online testing algorithm.
- IUT connectivity: adapters, value passing.
- Test traces (offline):
  - Symbolic state set display for debugging.
  - Conversion to test purpose for repeating test run.
  - Model coverage analysis.
- Obtain coverage and guide testing on-the-fly.
- Non-determinism.