Real-Time Systems

Lecture 19: Quasi-Equal Clocks

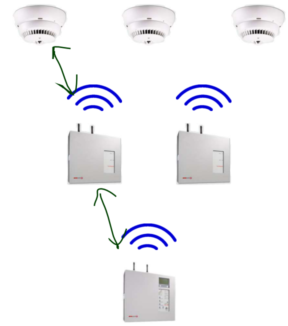
2018-01-25

Dr. Bernd Westphal

Albert-Ludwigs-Universität Freiburg, Germany

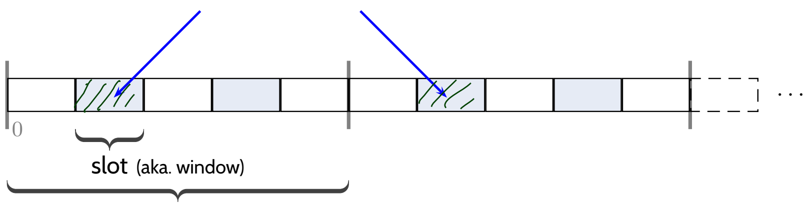
Motivation

WFAS Self-Monitoring

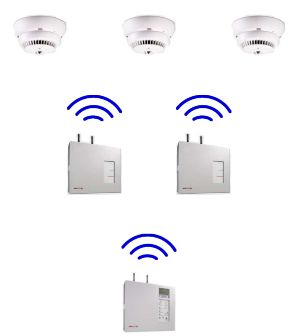


- Periodically, **each sensor** sends a “**hi master, I'm still here**” message to its master.
- If a master misses that message from one of its sensors: report incidence.
- To avoid message collision, employ a TDMA (time division multiple access) scheme.

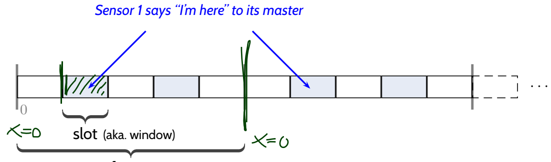
(Arenis et al., 2016)

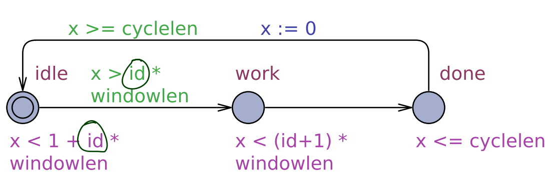


WFAS Self-Monitoring

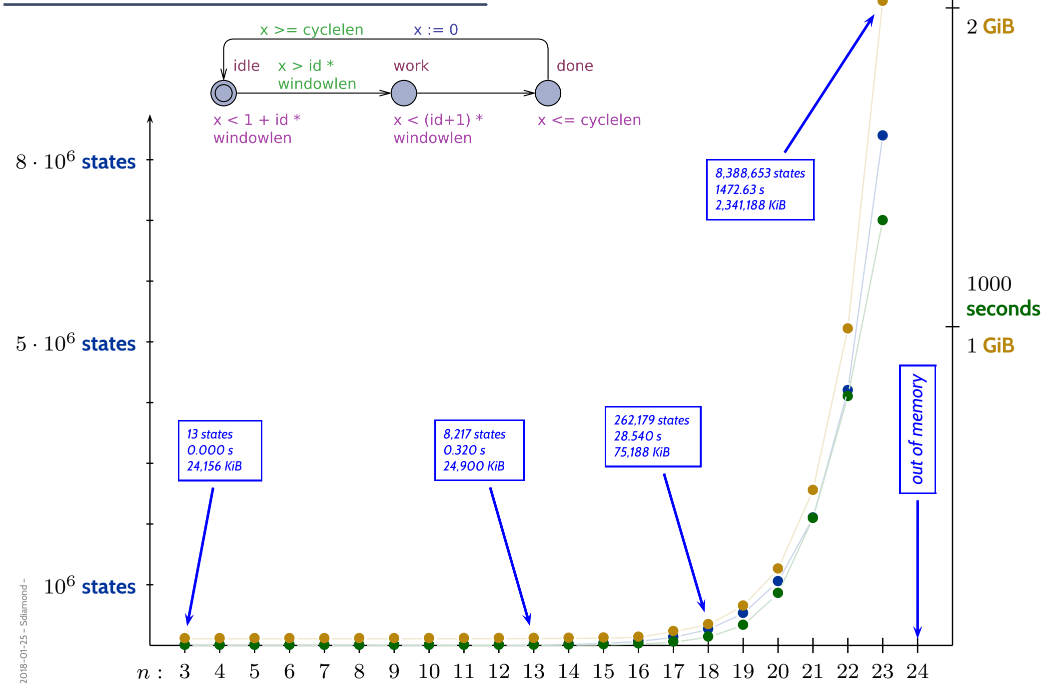


(Arenis et al., 2016)



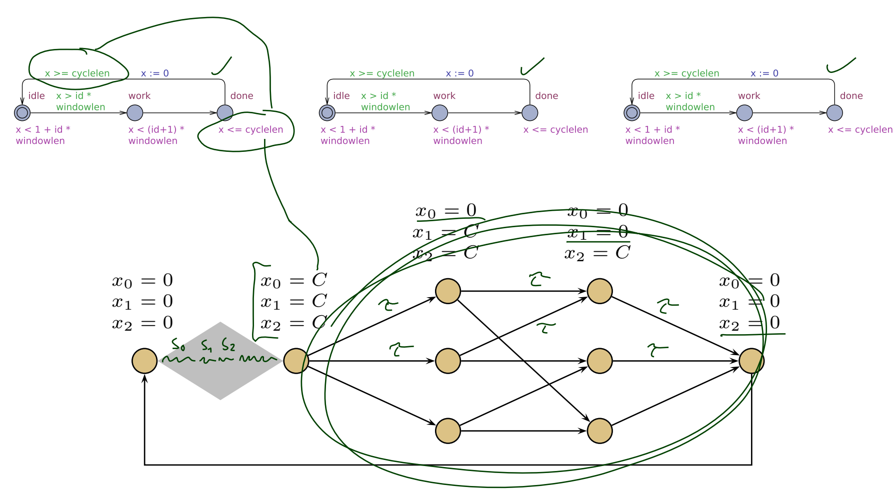


Experiments (`A[] not deadlock`)



A Closer Look

- Option 1: well, that's exponential space complexity, we need to accept that.
- Option 2: take a closer look.



Content

- **Quasi-Equal Clocks**
  - **Definition**, **Properties**
- **QE Clock Reduction**
  - **The simple, and wrong approach**
  - **Transformation example**
  - **Experiments**
  - **Simple** and **Complex Edges**
  - **Transformation schemes**
- **Correctness of the Transformation**
- **Excursion: Bisimulation Proofs**
- **Proof of QE-Correctness**
  - a particular **weak bisimulation relation**
- **More Experiments**

Quasi-Equal Clocks

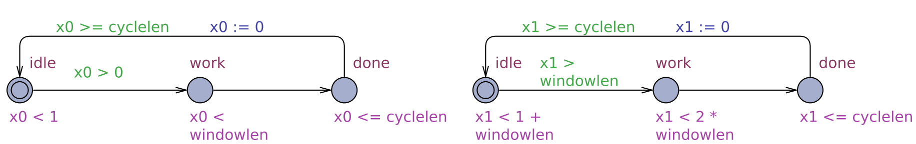
Quasi-Equal Clocks

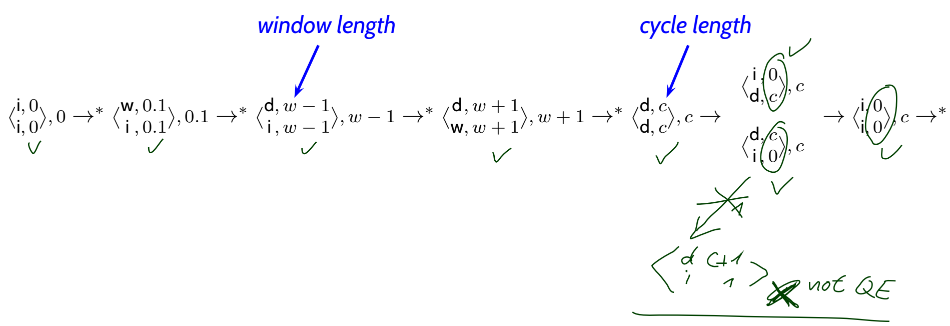
Definition. Let $\mathcal{N}$ be a network of timed automata with clocks X. Two clocks $x, y \in X$ are called quasi equal, denoted by $x \simeq y$, if and only if, for all reachable configurations of $\mathcal{N}$, x and y are equal or at least one has value 0, i.e.

$$\forall \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \xrightarrow{\lambda_1} \langle \vec{\ell}_1, \nu_1 \rangle, t_1 \ldots \in \mathit{Paths}(\mathcal{N}) \forall i \in \mathbb{N}_0 \bullet$$
$$\nu_i \models (x = y \lor x = 0 \lor y = 0).$$

Example

$$\forall \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \ldots \in \mathit{Paths}(\mathcal{N}) \forall i \in \mathbb{N}_0 \bullet \nu_i \models (x = y \lor x = 0 \lor y = 0).$$





Properties of Quasi-Equality

$$\forall \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \ldots \in \textit{Paths}(\mathcal{N}) \forall i \in \mathbb{N}_0 \bullet \nu_i \models (x = y \vee x = 0 \vee y = 0).$$

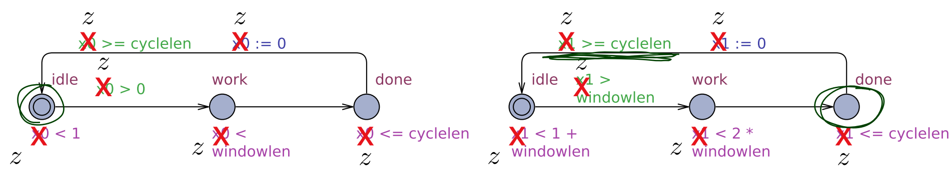
Lemma. Quasi-Equality is an equivalence relation.

Proof:

- **reflexive**: obvious.
- **symmetric**: obvious.
- **transitive**: a bit tricky
  (induction over a stronger property).

Quasi-Equal Clock Reduction

Idea: Use Just One Clock



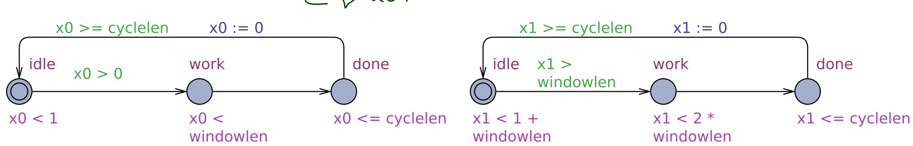
- **Behaviour**:

$$\left\langle {\mathsf{i}, 0 \atop \mathsf{i},} \right\rangle, 0 \to^* \left\langle {\mathsf{w}, 0.1 \atop \mathsf{i}\,,} \right\rangle, 0.1 \to^* \left\langle {\mathsf{d}, c \atop \mathsf{d},} \right\rangle, c \quad \begin{array}{l} \nearrow \left\langle {\mathsf{i}, 0 \atop \mathsf{d},} \right\rangle, c \\ \searrow \left\langle {\mathsf{d}, 0 \atop \mathsf{i},} \right\rangle, c \end{array}$$

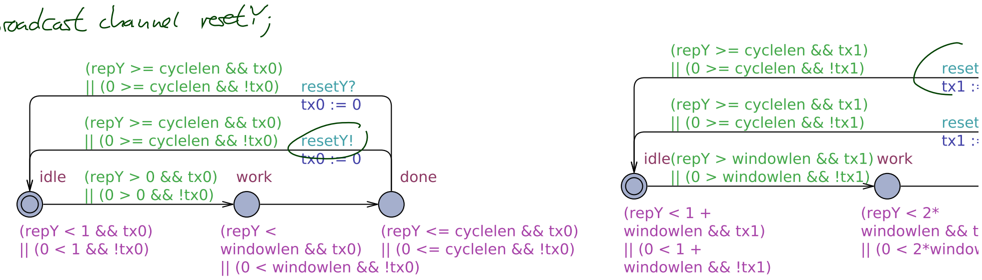
A More Elaborate Transformation

Y={x,y}, x≃y

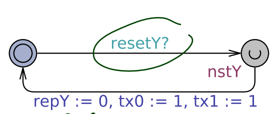
E◇ x0≠x1



broadcast channel resetY;

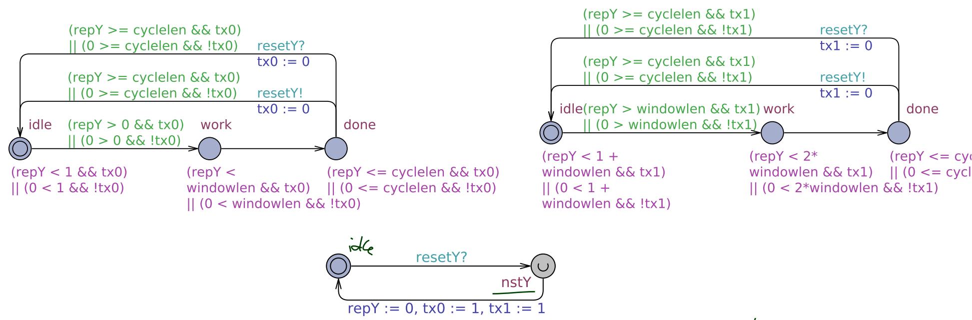


clock repY;
bool tx=1;
bool ty=1;



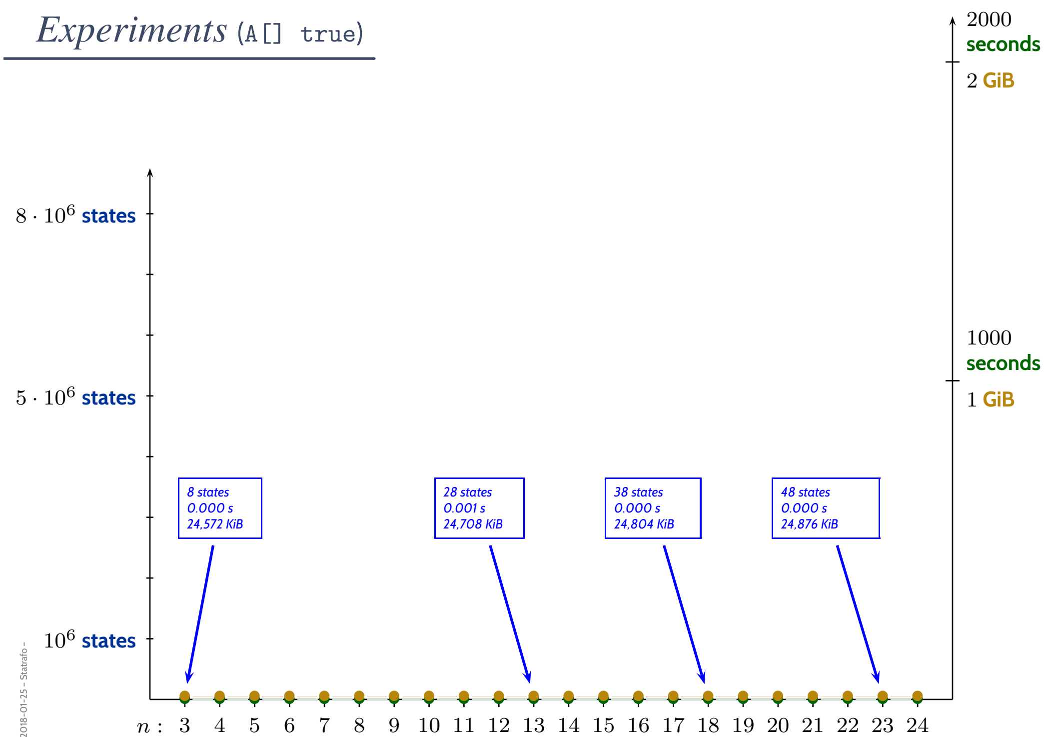
How Does It Work?

$$\left\langle {\mathsf{i},0 \atop \mathsf{i},0} \right\rangle, 0 \to^* \left\langle {\mathsf{w},0.1 \atop \mathsf{i},0.1} \right\rangle, 0.1 \to^* \left\langle {\mathsf{d},w-1 \atop \mathsf{i},w-1} \right\rangle, w-1 \to^* \left\langle {\mathsf{d},w+1 \atop \mathsf{w},w+1} \right\rangle, w+1 \to^* \left\langle {\mathsf{d},c \atop \mathsf{d},c} \right\rangle, c \to \begin{matrix} \left\langle {\mathsf{i},0 \atop \mathsf{d},c} \right\rangle, c \\ \left\langle {\mathsf{d},c \atop \mathsf{i},0} \right\rangle, c \end{matrix} \to \left\langle {\mathsf{i},0 \atop \mathsf{i},0} \right\rangle, c \to^*$$



$$\left\langle \begin{matrix} \mathsf{i},1 \\ \mathsf{i},1 \\ \mathsf{i},0 \end{matrix} \right\rangle, 0 \to^* \left\langle \begin{matrix} \mathsf{w}, & 1 \\ \mathsf{i}, & 1 \\ \mathsf{i}, & 0.1 \end{matrix} \right\rangle, 0.1 \to^* \left\langle \begin{matrix} \mathsf{d},1 \\ \mathsf{d},1 \\ \mathsf{i},c \end{matrix} \right\rangle, c \to \left\langle \begin{matrix} \mathsf{i},0 \\ \mathsf{i},0 \\ \mathsf{n},c \end{matrix} \right\rangle, c \to \left\langle \begin{matrix} \mathsf{i},1 \\ \mathsf{i},1 \\ \mathsf{i},0 \end{matrix} \right\rangle, c \to^*$$

Experiments (A[] true)



Simple Edges

Definition. An edge $e = (\ell, \alpha, \varphi, \vec{r}, \ell')$ resetting at least one quasi-equal clock is called simple edge if and only if the following conditions are satisfied:

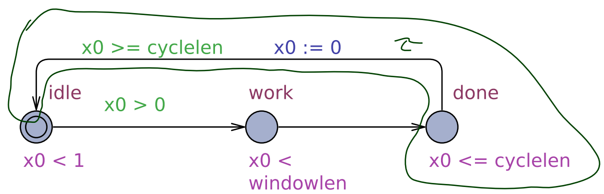
(i) $\alpha = \tau, \quad \varphi \equiv x \geq c, \quad \vec{r} = \langle x := 0 \rangle$,
for some constant c and **local clock** x,

(ii) $I(\ell) = x \leq c$,

(iii) e is **pre-** and **post-delayed**, and

(iv) e is the only edge with source ℓ.

Otherwise e is called complex edge.

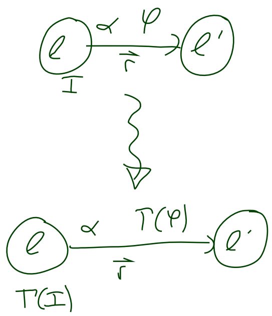


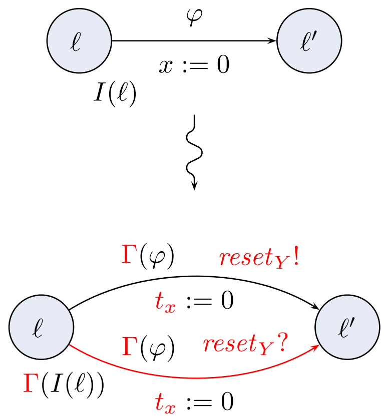
Transformation Scheme: Variables and Channels

Given a network $\mathcal{N}$ of timed automata, the **variables and channels** of **QE-transformation** of $\mathcal{N}'$ are obtained by the following procedure:

- **remove** all quasi-equal clocks from $\mathcal{N}$,
- for each **equivalence class** of quasi-equal clocks Y,
  add a fresh clock x_Y to $\mathcal{N}'$
- **add** a fresh boolean variable t_x to $\mathcal{N}'$
  for each quasi-equal clock x in $\mathcal{N}$,
  initial value: $t_x := 1$,
- **add** a fresh channel $reset_Y$ to $\mathcal{N}'$.

Transformation Scheme (for Simple Edges)





Constraint Transformation Γ

Definition. Let $\mathcal{N}$ be a network. Let $Y, W \in \mathcal{EC}_\mathcal{N}$ be sets of quasi-equal clocks of $\mathcal{N}$, $x \in Y$ and $y \in W$ clocks.

Given a clock constraint φ_{clk}, we define:

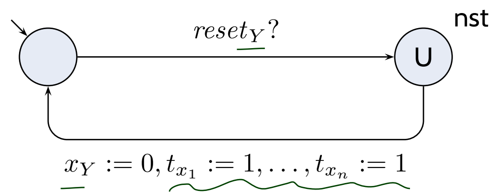
$$\Gamma_0(\varphi_{clk}) := \begin{cases} ((x_Y \sim c \wedge t_x) \vee (0 \sim c \wedge \neg t_x)) & \text{, if } \varphi_{clk} = x \sim c, \\ \big((x_Y - x_W \sim c \wedge t_x \wedge t_y) & \text{, if } \varphi_{clk} = x - y \sim c, \\ \quad \vee (0 - x_W \sim c \wedge \neg t_x \wedge t_y) & \\ \quad \vee (x_Y - 0 \sim c \wedge t_x \wedge \neg t_y) & \\ \quad \vee (0 \sim c \wedge \neg t_x \wedge \neg t_y)\big) & \\ \Gamma_0(\varphi_1) \wedge \Gamma_0(\varphi_2) & \text{, if } \varphi_{clk} = \varphi_1 \wedge \varphi_2. \end{cases}$$

Then $\Gamma(\varphi_{clk} \wedge \psi_{int}) := \Gamma_0(\varphi_{clk}) \wedge \psi_{int}$.

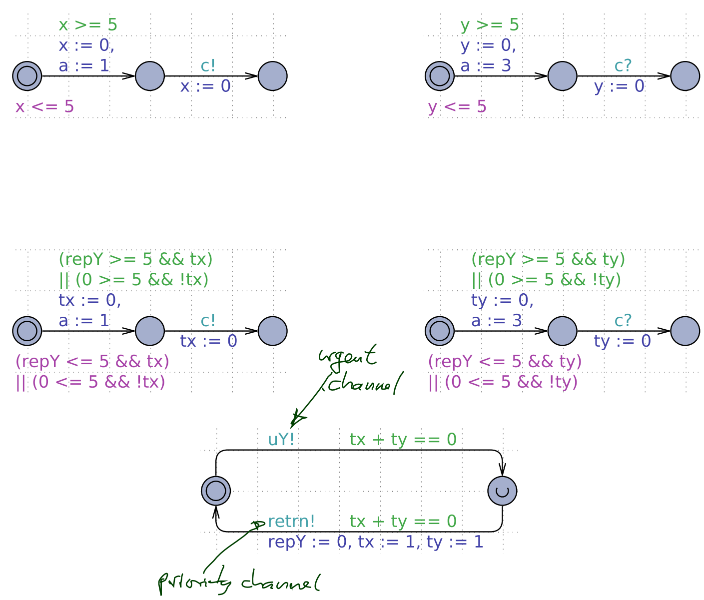
Here, $\mathcal{EC}_\mathcal{N}$ is the set of **equivalence classes** of **quasi-equal clocks** in $\mathcal{N}$.

Resetter Construction (for Simple Edges)

- For each equivalence class $Y = \{x_1, \dots, x_n\} \in \mathcal{EC}_\mathcal{N}$ **add** a **resetter** $\mathcal{R}_Y$ to $\mathcal{N}'$:



Transformation Example (for Complex Edges)



Correctness of the Transformation

QE-Transformation Correctness

Theorem. Let $\mathcal{N}$ be a network of timed automata and CF a configuration formula over $\mathcal{N}$. Then

$$\mathcal{N} \models \exists\Diamond\, CF \iff \mathcal{N}' \models \exists\Diamond\, \Omega(CF).$$

Query Transformation

Definition. Let $\mathcal{N} = \{\mathcal{A}_1, \dots, \mathcal{A}_n\}$ be a network with equivalence classes of quasi-equal clocks $\mathcal{EC}_\mathcal{N} = \{Y_1, \dots, Y_m\}$ and β a basic formula over $\mathcal{N}$.

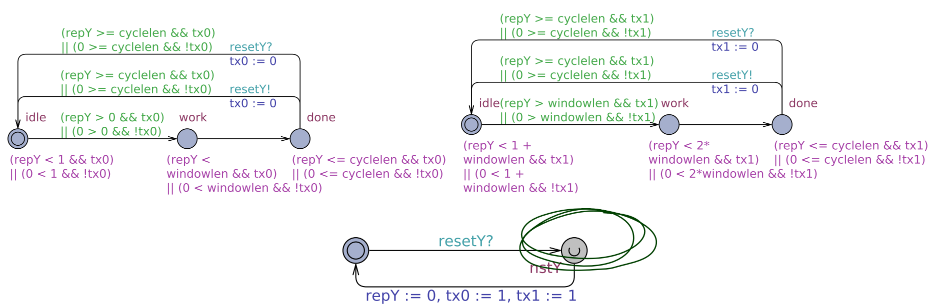
$$\Omega_0(\beta) = \begin{cases} \ell \vee (\ell' \wedge \tilde{x}) & \text{, if } \beta = \mathcal{A}_i.\ell, \quad (\ell, \alpha, \varphi, \langle x := 0 \rangle, \ell') \text{ simple.} \\ (\ell' \wedge \neg\tilde{x}) & \text{, if } \beta = \mathcal{A}_i.\ell', \quad (\ell, \alpha, \varphi, \langle x := 0 \rangle, \ell') \text{ simple.} \\ \beta & \text{, if } \beta \in \{\mathcal{A}_i.\ell, \mathcal{A}_i.\ell'\}, \quad (\ell, \alpha, \varphi, \vec{r}, \ell') \text{ not simple.} \\ \Gamma(\beta)[t_x/(t_x \vee \tilde{x}) \mid x \in Y, Y \in \mathcal{EC}_\mathcal{N}] & \text{, if } \beta = \varphi_{clk} \wedge \varphi_{int}. \end{cases}$$

$\Omega(CF) = \exists\, \tilde{x}_1, \dots, \tilde{x}_{|X(\mathcal{N})|} \bullet \Omega_0(CF) \wedge \kappa_\mathcal{N}$, where

$$\kappa_\mathcal{N} := \bigwedge_{\substack{1 \le i \le n, \\ 1 \le j \le m, \\ (\ell,\alpha,\varphi,\langle x:=0\rangle,\ell') \in SimpEdges_{Y_j}(\mathcal{A}_i)}} \kappa(x), \qquad \kappa(x) : (\tilde{x} \implies \bigvee_{(\ell,\alpha,\varphi,\langle x:=0\rangle,\ell') \in SimpEdges_{Y_j}(\mathcal{A}_i)} \ell' \wedge \ell_{nst\mathcal{R}_{Y_j}}).$$

By structural induction Ω_0 transforms configuration formulas CF.

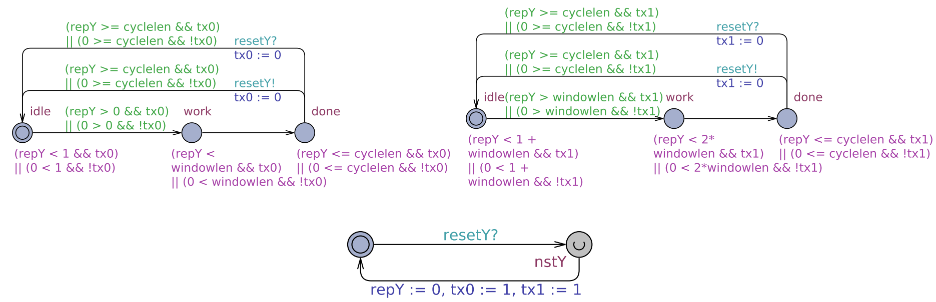
Example



$$\left\langle \begin{matrix} \mathsf{i}, 1 \\ \mathsf{i}, 1 \\ 0, 0 \end{matrix} \right\rangle, \to^* \left\langle \begin{matrix} \mathsf{w}, 1 \\ \mathsf{i}, 1 \\ 0.1, 0.1 \end{matrix} \right\rangle, \to^* \left\langle \begin{matrix} \mathsf{d}, 1 \\ \mathsf{d}, 1 \\ c, c \end{matrix} \right\rangle, \to \left\langle \begin{matrix} \mathsf{i}, 0 \\ \mathsf{i}, 0 \\ c, c \end{matrix} \right\rangle, \to \left\langle \begin{matrix} \mathsf{i}, 1 \\ \mathsf{i}, 1 \\ 0, c \end{matrix} \right\rangle, \to^*$$

- $\mathcal{N} \models \exists\Diamond$ S0.idle $\wedge$ S1.done
- $\mathcal{N}' \models \exists\Diamond(\exists\, \tilde{x}_0, \tilde{x}_1 \bullet$ (S0.idle $\wedge \neg\tilde{x}_0) \wedge$ (S1.done $\vee$ (S1.idle $\wedge\, \tilde{x}_1$))
  $\wedge(\tilde{x}_0 \implies$ (S0.idle $\wedge$ nst)) $\wedge\, (\tilde{x}_1 \implies$ (S1.idle $\wedge$ nst)))

Example



$$\left\langle \begin{matrix} \mathsf{i}, 1 \\ \mathsf{i}, 1 \\ \mathsf{O}, 0 \end{matrix} \right\rangle, \to^* \left\langle \begin{matrix} \mathsf{w}, 1 \\ \mathsf{i}, 1 \\ \mathsf{O.1}, 0.1 \end{matrix} \right\rangle, \to^* \left\langle \begin{matrix} \mathsf{d}, 1 \\ \mathsf{d}, 1 \\ \mathsf{c}, c \end{matrix} \right\rangle, \to \left\langle \begin{matrix} \mathsf{i}, 0 \\ \mathsf{i}, 0 \\ \mathsf{c}, c \end{matrix} \right\rangle, \to \left\langle \begin{matrix} \mathsf{i}, 1 \\ \mathsf{i}, 1 \\ \mathsf{O}, c \end{matrix} \right\rangle, \to^*$$

- $\mathcal{N} \models \exists\Diamond(x_0 = 0 \wedge x_1 > 0)$
- $\mathcal{N}' \models \exists\Diamond(\exists\, \tilde{x}_0, \tilde{x}_1 \bullet ((x_0 = 0 \wedge (t_{x_0} \vee \tilde{x}_0)) \vee (0 = 0 \wedge \neg(t_{x_0} \vee \tilde{x}_0)))$
  $\wedge((x_1 > 0 \wedge (t_{x_1} \vee \tilde{x}_1)) \vee (0 > 0 \wedge \neg(t_{x_1} \vee \tilde{x}_1)))$
  $\wedge(\tilde{x}_0 \implies (\mathsf{S0.idle} \wedge \mathsf{nst})) \wedge (\tilde{x}_1 \implies (\mathsf{S1.idle} \wedge \mathsf{nst})))$

Bisimulation Proofs

Proof Sketch

- Use a **weak bisimulation relation** – the basic idea:
  - Let $\mathcal{T}_i = (Conf_i, \Lambda_i, \{\xrightarrow{\lambda} \mid \lambda \in \Lambda_i\}, C_{\mathsf{ini},i})$, $i = 1, 2$, be labelled transition systems with (for simplicity) $C_{\mathsf{ini},i} = \{c_{\mathsf{ini},i}\}$.
  - A relation $R \subseteq Conf_1 \times Conf_2$ is called **weak bisimulation** if and only if
    - (i) the **initial configurations** are related, i.e. $(c_{\mathsf{ini},1}, c_{\mathsf{ini},2}) \in R$,
    - (ii) two related configurations **satisfy the same terms**, i.e.

$$\forall\, c_1, c_2, term \bullet (c_1, c_2) \in R \implies (c_1 \models term \iff c_2 \models term)$$

    - (iii) given two related configurations $(c_1, c_2) \in R$,
      - a) if $\mathcal{T}_1$ has a λ-transition from c_1 to some c_1', then $\mathcal{T}_2$ has τ- and λ-transitions from c_2 to a related c_2', i.e.

$$\forall\, c_1' \bullet c_1 \xrightarrow{\lambda} c_1' \implies \exists\, c_2' \bullet c_2 \xrightarrow{\lambda}{}^{*} c_2' \land (c_1', c_2') \in R$$

      - b) similarly for $\mathcal{T}_2$ to $\mathcal{T}_1$, i.e.

$$\forall\, c_2' \bullet c_2 \xrightarrow{\lambda} c_2' \implies \exists\, c_1' \bullet c_1 \xrightarrow{\lambda}{}^{*} c_1' \land (c_1', c_2') \in R$$

- $\mathcal{T}_1$ and $\mathcal{T}_2$ are called **weakly bisimilar** iff there exists a weak bisimulation for $\mathcal{T}_1, \mathcal{T}_2$.

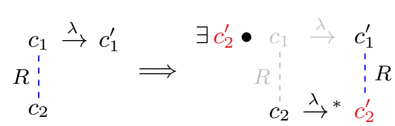
Once Again

(i) $(c_{\mathsf{ini},1}, c_{\mathsf{ini},2}) \in R$,

(ii) $\forall\, c_1, c_2, term \bullet (c_1, c_2) \in R \implies (c_1 \models term \iff c_2 \models term)$

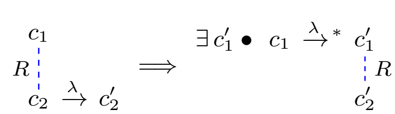
(iii) for all $(c_1, c_2) \in R$,

a) **"$\mathcal{T}_2$ can simulate transitions of $\mathcal{T}_1$"**:



(using any finite number of τ-transitions in between)

b) **"$\mathcal{T}_1$ can simulate transitions of $\mathcal{T}_2$"**:



Example

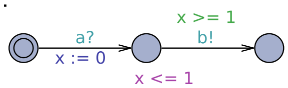
(i) $(c_{\text{ini},1}, c_{\text{ini},2}) \in R$, (ii) $\forall\, c_1, c_2, term \bullet (c_1, c_2) \in R \implies (c_1 \models term \iff c_2 \models term)$

(iii) for all $(c_1, c_2) \in R$,

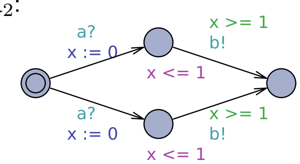
a) $c_1 \xrightarrow{\lambda} c_1'$, $c_1 \; R \; c_2$ $\implies$ $\exists\, c_2' \bullet c_2 \xrightarrow{\lambda}{}^* c_2'$, $c_1' \; R \; c_2'$

b) $c_2 \xrightarrow{\lambda} c_2'$, $c_1 \; R \; c_2$ $\implies$ $\exists\, c_1' \bullet c_1 \xrightarrow{\lambda}{}^* c_1'$, $c_1' \; R \; c_2'$

$\mathcal{A}_1$:



$\mathcal{A}_2$:



Example

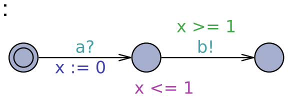
(i) $(c_{\text{ini},1}, c_{\text{ini},2}) \in R$, (ii) $\forall\, c_1, c_2, term \bullet (c_1, c_2) \in R \implies (c_1 \models term \iff c_2 \models term)$

(iii) for all $(c_1, c_2) \in R$,

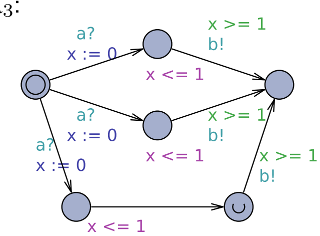
a) $c_1 \xrightarrow{\lambda} c_1'$, $c_1 \; R \; c_2$ $\implies$ $\exists\, c_2' \bullet c_2 \xrightarrow{\lambda}{}^* c_2'$, $c_1' \; R \; c_2'$

b) $c_2 \xrightarrow{\lambda} c_2'$, $c_1 \; R \; c_2$ $\implies$ $\exists\, c_1' \bullet c_1 \xrightarrow{\lambda}{}^* c_1'$, $c_1' \; R \; c_2'$

$\mathcal{A}_1$:

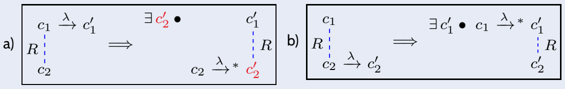


$\mathcal{A}_3$:



What is It Good For?

(i) $(c_{\text{ini},1}, c_{\text{ini},2}) \in R$, (ii) $\forall\, c_1, c_2, term \bullet (c_1, c_2) \in R \implies (c_1 \models term \iff c_2 \models term)$
(iii) for all $(c_1, c_2) \in R$,



- Let $term$ be a term over two weakly bisimilar networks $\mathcal{N}$ and $\mathcal{N}'$.
- **Claim**: $\mathcal{N} \models \exists\Diamond\, term \iff \mathcal{N}' \models \exists\Diamond\, term$.
- **Proof**:
  - Because $\mathcal{N}$ and $\mathcal{N}'$ are weakly bisimilar, there is a **simulation relation** R.
  - Direction "$\implies$": Let $\mathcal{N} \models \exists\Diamond\, term$.
    - Thus there is a **computation path** $c_{1,0} \xrightarrow{\lambda_1} c_{1,1} \xrightarrow{\lambda_2} \dots \xrightarrow{\lambda_n} c_{1,n}$ with $c_{1,n} \models term$.
    - **Induction over length of path**:
      - **Case** $n = 0$:
        Then $c_{1,0} \models term$ and $c_{0,1}$ is an initial configuration,
        thus $c_{2,0}$ is R-related (by (i)) and thus $c_{2,0} \models term$ (by (ii)).
      - **Case** $n \to n+1$:
        For the path $c_{1,0} \xrightarrow{\lambda_1} \dots \xrightarrow{\lambda_n} c_{1,n} \xrightarrow{\lambda_{n+1}} c_{1,n+1}$, there is (by **induction hypothesis**) an R-related configuration $c_{2,m}$, $m \geq n$, **reachable** in $\mathcal{N}'$.

        By (iii).a), there is a configuration $c'_{2,m}$, which is R-related to $c_{1,n+1}$, and **reachable** from $c_{2,m}$, thus, by (ii), $c_{1,n+1} \models term$.
  - Direction "$\Longleftarrow$": similar. $\square$

Proof of QE-Correctness

Another Weak Bisimulation Relation Notion

Definition. [*Weak Bisimulation*]

Networks $\mathcal{N}, \mathcal{N}'$ are called weakly bisimilar if and only if there is a weak bisimulation relation $QE \subseteq Conf(\mathcal{N}) \times Conf(\mathcal{N}')$ such that:

(i) $\forall\, s \in C_{ini}(\mathcal{N})\ \exists\, r \in C_{ini}(\mathcal{N}') \bullet (s, r) \in QE$,
$\forall\, r \in C_{ini}(\mathcal{N}')\ \exists\, s \in C_{ini}(\mathcal{N}) \bullet (s, r) \in QE$.

(ii) $\forall\, CF \in \mathcal{CF}_\mathcal{N} \forall\, (s, r) \in QE \bullet s \models_\delta CF \implies r \models_\delta \Omega(CF)$.

(iii) $\forall\, CF \in \mathcal{CF}_\mathcal{N} \forall\, (s, r) \in QE \bullet$
$r \models_\delta \Omega(CF) \implies \exists\, \dot{s} \in Conf(\mathcal{N}) \bullet (\dot{s}, r) \in QE \wedge \dot{s} \models_\delta CF$.

(iv) $\forall\, (s, r) \in QE\ \forall\, \lambda, s' \bullet s \xrightarrow{\lambda} s' \implies \exists\, r' \bullet r \xrightarrow{\lambda}{}^* r' \wedge (s', r') \in QE$

(v) $\forall\, CF \in \mathcal{CF}_\mathcal{N} \forall\, (s, r) \in QE\ \forall\, \lambda, r' \bullet$
$r \xrightarrow{\lambda} r' \wedge r' \models_{\delta'} \Omega_0(CF) \implies \exists\, s' \bullet s \xrightarrow{\lambda}{}^* s' \wedge (s', r') \in QE$.

Here, $r \xrightarrow{\tau}{}^* r'$ denotes zero or more successive τ-transitions from r to r'.

- 19 - 2018-01-25 - Sproof -

A Weak Bisimulation Relation for QE-Transformation

- Let $\mathcal{N}$ be a network of timed automata and $\mathcal{N}'$ the network obtained by QE-transformation of $\mathcal{N}$. Then $QE : Conf(\mathcal{N}) \to 2^{Conf\,\mathcal{N}'}$ defined as follows is a **weak bisimulation relation**.

$$QE(\langle \vec{\ell}_{\dot{s}}, \nu_{\dot{s}} \rangle) = \Big\{ r = \langle (\ell_{r,1}, \dots, \ell_{r,n}, \ell_{\mathcal{R}_{Y_1}}, \dots, \ell_{\mathcal{R}_{Y_m}}), \nu_r \rangle \mid$$

$$(\forall\, x \in V(\mathcal{N}) \bullet \nu_r(x) = \nu_{\dot{s}}(x)) \tag{6.2.1}$$

$$\wedge\ \forall\, 1 \leq i \leq n\ \bullet \tag{6.2.2}$$

$$\Big(\big(\ell_{r,i} = \ell_{\dot{s},i} \wedge \forall\, x \in X(\mathcal{A}_i) \bullet \nu_{\dot{s}}(x) = \nu_r(x_x) \cdot \nu_r(t_x)\big) \tag{6.2.2a}$$

$$\vee \big(\exists\, (\ell, \alpha, \varphi, \langle x := 0 \rangle, \ell') \in SimpEdges_Y(\mathcal{A}_i) \bullet \ell_{\mathcal{R}_Y} \neq \ell_{ini\mathcal{R}_Y} \wedge \ell_{\dot{s},i} = \ell \wedge \ell_{r,i} = \ell' \wedge \nu_{\dot{s}}(x) = \nu_r(x_x) \wedge \nu_r(t_x) = 0 \wedge \forall\, y \in X(\mathcal{A}_i) \setminus \{x\} \bullet \nu_{\dot{s}}(y) = \nu_r(x_y) \cdot \nu_r(t_y)\big)\Big) \tag{6.2.2b}$$

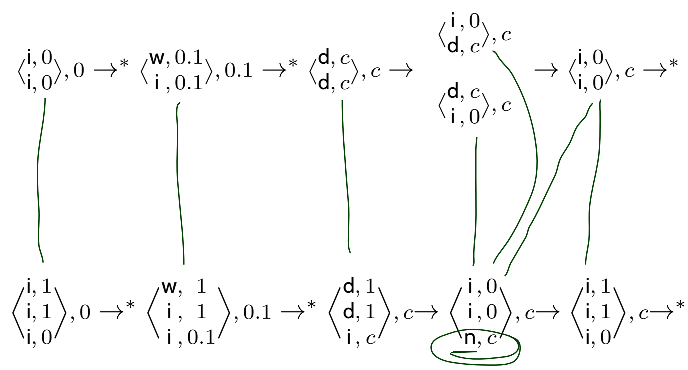
$$\wedge\ \forall\, Y \in \mathcal{EC}_\mathcal{N}\ \bullet$$

$$\Big((\nu_r(s_Y^{\mathcal{A}_i}) = 1 \iff \exists\, (\ell, \alpha, \varphi, \vec{r}, \ell') \in SimpEdges_Y(\mathcal{A}_i) \bullet \ell_{r,i} = \ell) \tag{6.2.3}$$

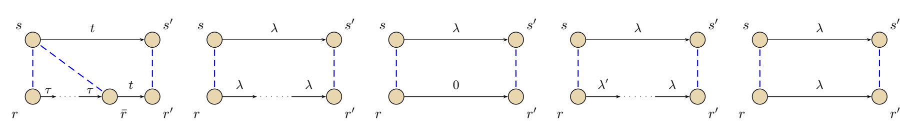
$$\wedge\ \nu_r(prio_Y) = 1 \iff (\ell_{r,\mathcal{R}_Y} = \ell_{nst\mathcal{R}_Y})\Big)\Big\} \tag{6.2.4}$$

- 19 - 2018-01-25 - Sproof -

Example



Proof of Having Indeed a Bisimulation

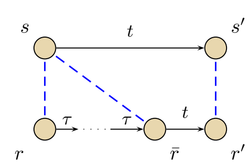


- $s \xrightarrow{\lambda} s'$ to $r \xrightarrow{\lambda}^* r'$:

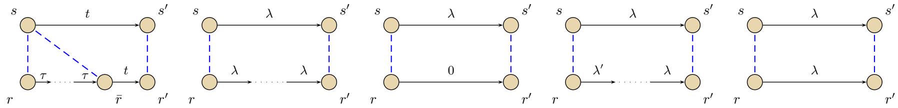
  Cases:

  - delay $d > 0$:

    resetter may need to go back to idle, then do same delay.



Proof of Having Indeed a Bisimulation

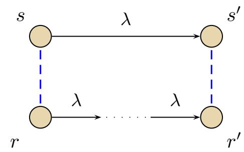


- $s \xrightarrow{\lambda} s'$ to $r \xrightarrow{\lambda}^* r'$:

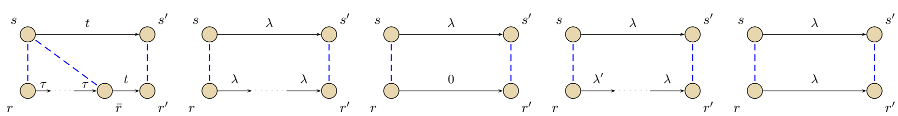
  Cases:

  - delay $d > 0$
  - first simple edge:

    first simple edges pushes resetter and all other simples.



Proof of Having Indeed a Bisimulation

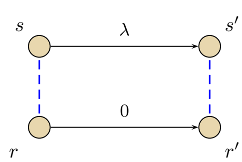


- $s \xrightarrow{\lambda} s'$ to $r \xrightarrow{\lambda}^* r'$:

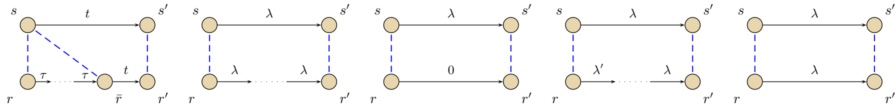
  Cases:

  - delay $d > 0$
  - first simple edge
  - other simple edge:

    resetter is in nst, do nothing in $\mathcal{N}'$.



Proof of Having Indeed a Bisimulation

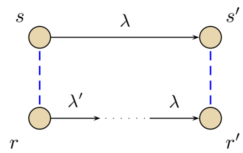


- $s \xrightarrow{\lambda} s'$ to $r \xrightarrow{\lambda}{}^{*} r'$:

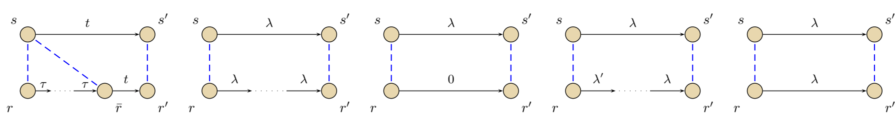
  Cases:

  - delay $d > 0$
  - first simple edge
  - other simple edge
  - non-reset, or at least one complex:

    resetter may need to push simples first, then take same edge in $\mathcal{N}'$.



Proof of Having Indeed a Bisimulation

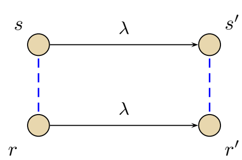


- $s \xrightarrow{\lambda} s'$ to $r \xrightarrow{\lambda}{}^{*} r'$:

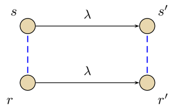
  Cases:

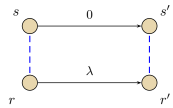
  - delay $d > 0$
  - first simple edge
  - other simple edge
  - non-reset, or at least one complex
  - delay $d = 0$:

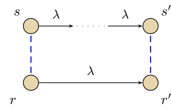
    do same delay in $\mathcal{N}'$.



Proof of Having Indeed a Bisimulation





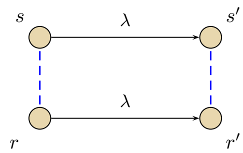


- $r \xrightarrow{\lambda} r'$ to $s \xrightarrow{\lambda}^{*} s'$:

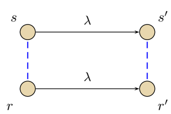
  Cases:

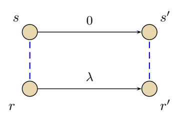
  - delay $d > 0$:

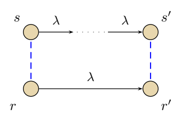
    do same delay in $\mathcal{N}$.



Proof of Having Indeed a Bisimulation





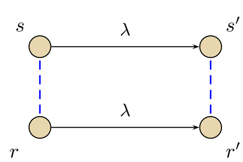


- $r \xrightarrow{\lambda} r'$ to $s \xrightarrow{\lambda}^{*} s'$:

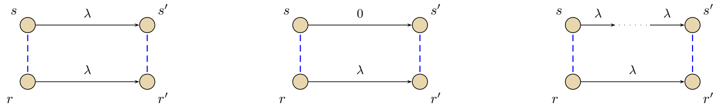
  Cases:

  - delay $d > 0$
  - complex, or non-resetting:

    take same edge in $\mathcal{N}$.



Proof of Having Indeed a Bisimulation

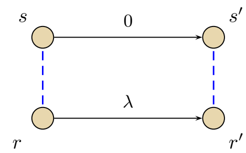


- $r \xrightarrow{\lambda} r'$ to $s \xrightarrow{\lambda}^{*} s'$:

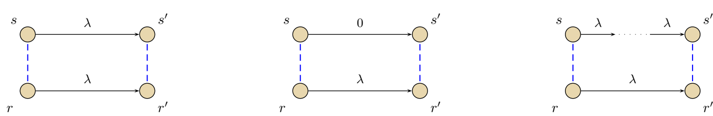
  Cases:

  - delay $d > 0$
  - complex, or non-resetting
  - resetter to nst,
    or returns (no simples enab. in $\mathcal{N}$):

    do nothing
    in $\mathcal{N}$.



Proof of Having Indeed a Bisimulation

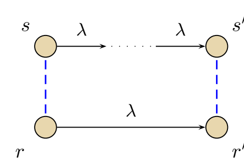


- $r \xrightarrow{\lambda} r'$ to $s \xrightarrow{\lambda}^{*} s'$:

  Cases:

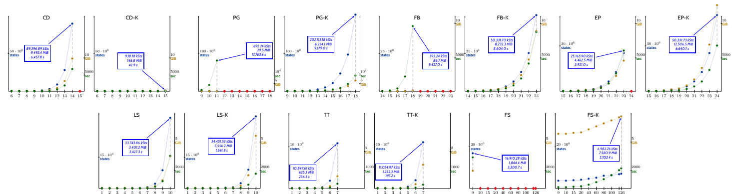
  - delay $d > 0$
  - complex, or non-resetting
  - resetter to nst,
    or returns (no simples enab. in $\mathcal{N}$)
  - resetter returns (some simples enab. in $\mathcal{N}$):

    take all enabled
    simple edges in $\mathcal{N}$.

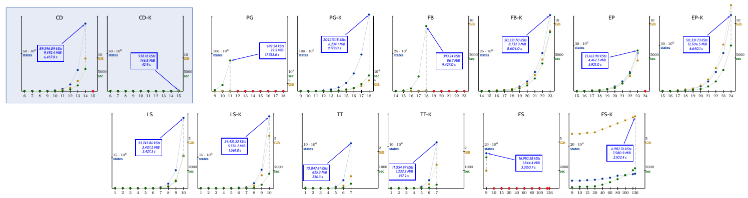


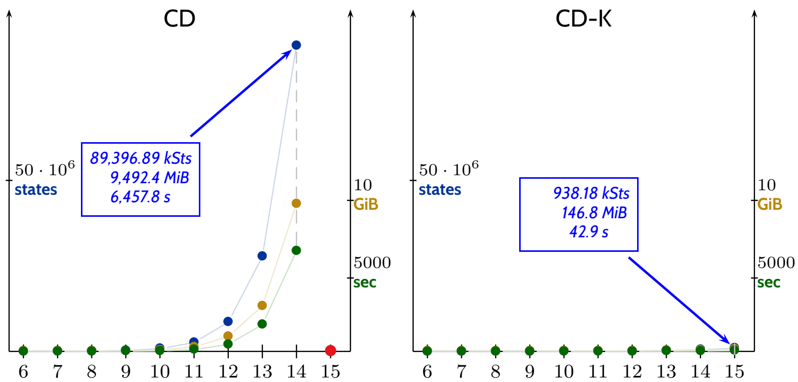
More Experiments

Case Studies



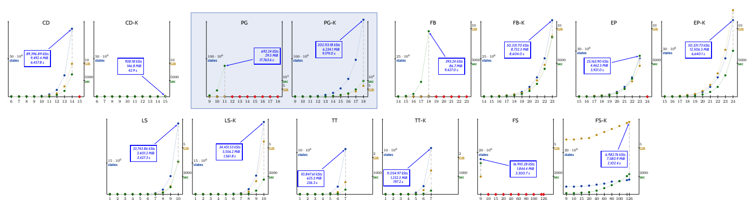
Case Studies

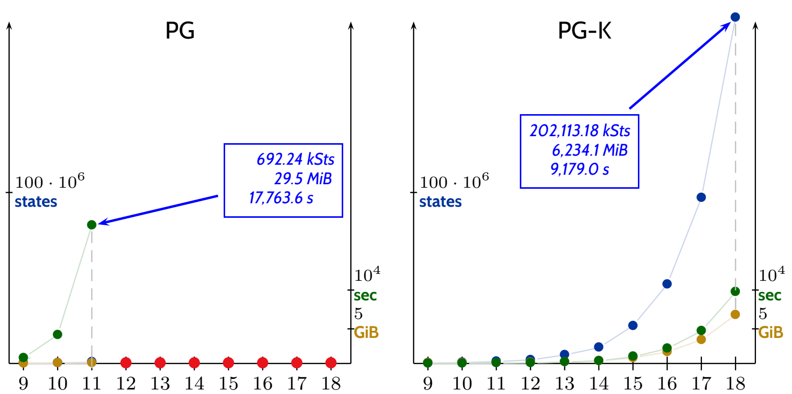




H. E. Jensen, K. G. Larsen, et al. *"Modelling and Analysis of a Collision Avoidance Protocol using SPIN and Uppaal"*. In: DIMACS. Vol. 32. DIMACS. 1996, pp. 33–50.

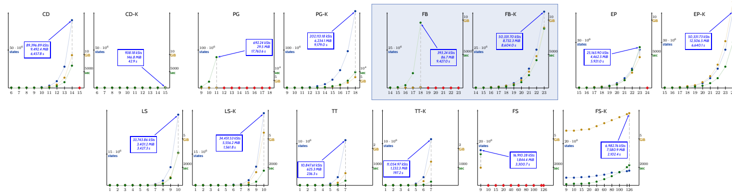
Case Studies

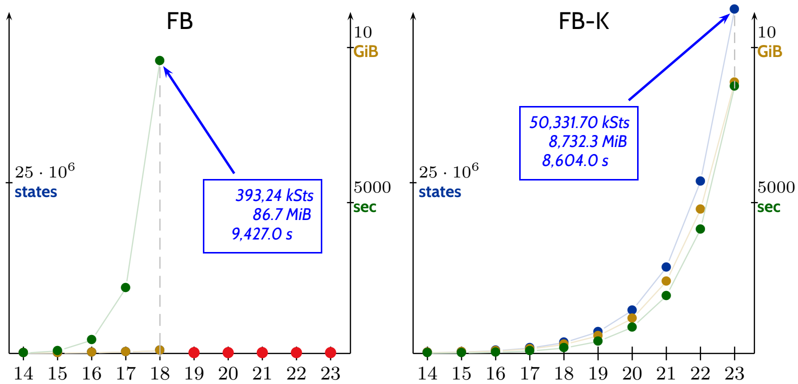




B. Bérard, P. Bouyer, et al. *"Analysing the PGM Protocol with UPPAAL"*. In: IJPR 42.14 (2004), pp. 2773–2791.

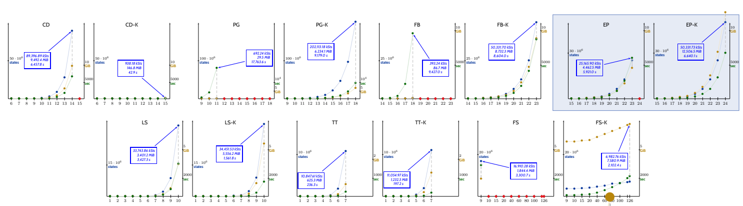
Case Studies

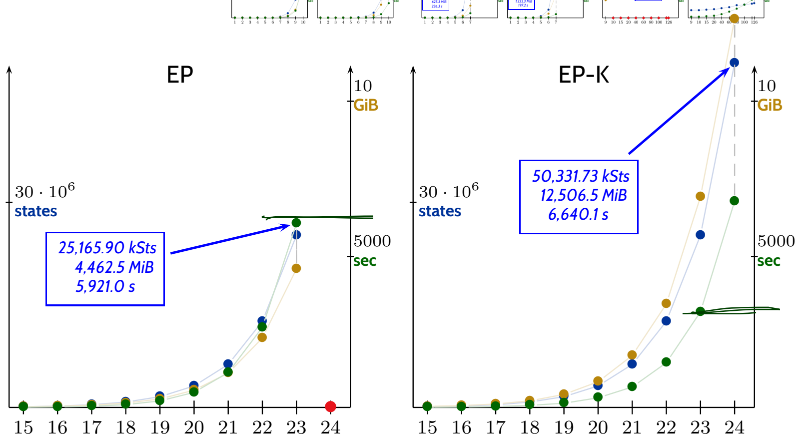




N. Petalidis. *"Verification of a Fieldbus Scheduling Protocol Using Timed Automata"*. In: CI 28.5 (2009), pp. 655-672.

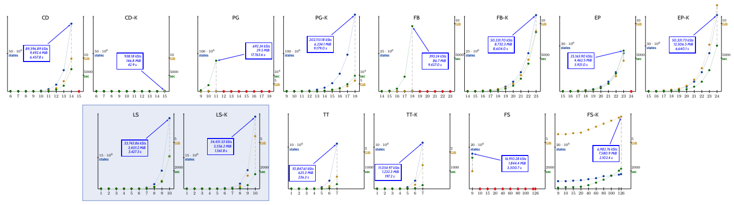
Case Studies

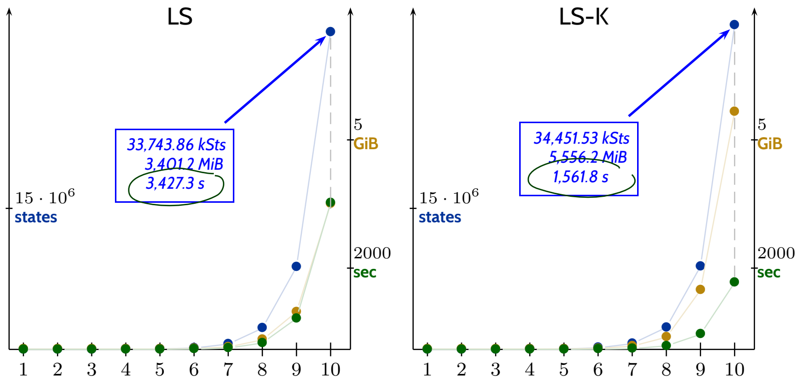




S. Limal, S. Potier, et al. *"Formal Verification of Redundant Media Extension of Ethernet PowerLink"*. In: ETFA. IEEE, 2007, pp. 1045-1052.

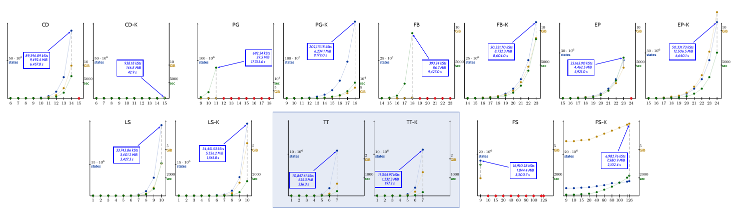
Case Studies

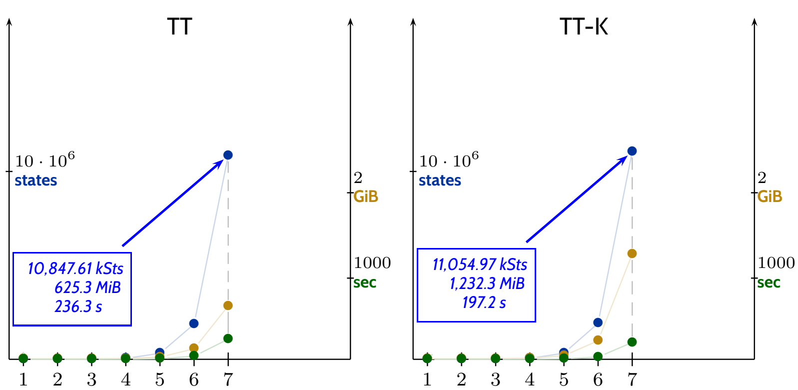




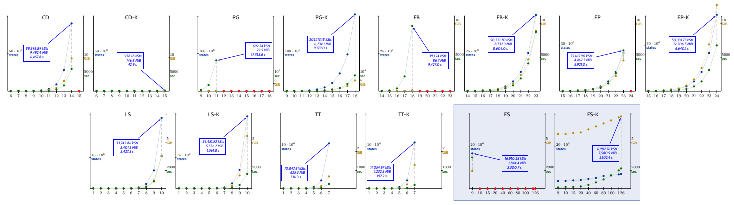
P. Kordy, R. Langerak, et al. *"Re-verification of a Lip Synchronization Protocol Using Robust Reachability"*. In: FMA. Vol. 20. EPTCS. 2009, pp. 49-62.

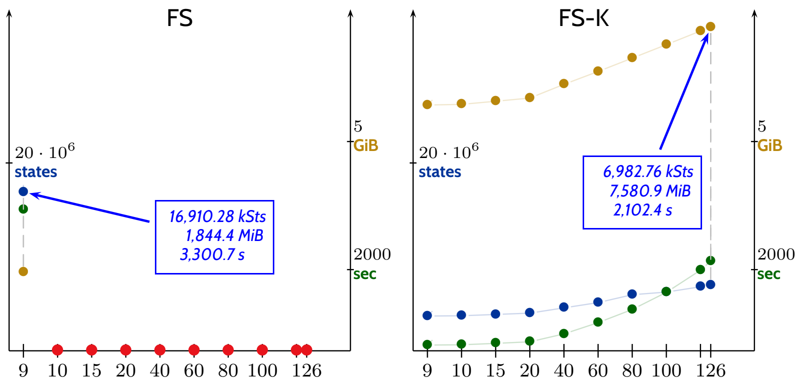
Case Studies





W. Steiner and W. Elmenreich. *"Automatic Recovery of the TTP/A Sensor/Actuator Network"*. In: WISES. Vienna University of Technology, 2003, pp. 25-37.





D. Dietsch, A. Podelski, et al. *"Disambiguation of Industrial Standards Through Formalization and Graphical Languages"*. In: RE. IEEE, 2011, pp. 265-270.

Savings

Upper Bound on Number of Configurations

Theorem. Let $\mathcal{N}$ be a network of timed automata with equivalence classes of quasi-equal clocks $\mathcal{EC}_\mathcal{N} = \{Y_1, \dots, Y_m\}$.

Then the number of configurations of $\mathcal{N}'$ is bounded above by:

$$|L(\mathcal{A}_1) \times \dots \times L(\mathcal{A}_n) \times L(\mathcal{R}_{Y_1}) \times \dots \times L(\mathcal{R}_{Y_m})|$$
$$\cdot (2c+2)^{|\mathcal{EC}_\mathcal{N}|} \cdot (4c+3)^{\frac{1}{2}|\mathcal{EC}_\mathcal{N}| \cdot (|\mathcal{EC}_\mathcal{N}|-1)}$$
$$\cdot 2^{|Y_1 \cup \dots \cup Y_m|},$$

where $c = max\{c_x \mid x \in \mathcal{X}(\mathcal{N})\}$.

– 19 – 2018-01-25 – Ssavings –

Only Simple Edges

Lemma. Let $\mathcal{N}$ be a network of timed automata with a set of equivalence classes of quasi-equal clocks $\mathcal{EC}_\mathcal{N}$, where

- $|Y| \geq 2, Y \in \mathcal{EC}_\mathcal{N}$, and
- each clock $x \in Y, Y \in \mathcal{EC}_\mathcal{N}$, is **exclusively reset by simple edges**.

Then $|Reach_{\mathcal{N}'}| < |Reach_\mathcal{N}|$.

(Here, $Reach_\mathcal{N}$ denotes the set of all reachable (zone graph-)configurations of $\mathcal{N}$.)

Proof: Use the following lemma.

Lemma. Let $\mathcal{N}$ be a network where all quasi-equal clocks are exclusively reset by simple edges. Then

$$|Reach_{\mathcal{N}'}| = |Reach_\mathcal{N}| - \big(\sum_{s \in RC} 2^{|clks(s)|}\ \big) + \sum_{s \in RC} \Big[|class(s)| + 2\Big].$$

– 19 – 2018-01-25 – Ssavings –

Complex Edges

- "it's (a bit more) complicated"

Content

Tell Them What You've Told Them...

- The **space complexity** of Pure-TA reachability-checking is

$$L_1 \times \cdots \times L_n \times \text{Regions}(X),$$

  i.e., exponential in number of clocks, and **of TA**.

- If a model is **expensive to check**,
  - it may necessarily be that expensive,
  - or artificially / non-necessarily.

  $\rightarrow$ take a closer look ($\rightarrow$ exercises).
- One example: **Quasi-equal clocks**
  - advantage: **can be good for validation**,
  - dis-advantage: **expensive to check**.
- The **QE transformation** (source-to-source)
  - **eliminates interleavins of simple edges**,
  - reduces DBM size to $($**number of equiv. classes**$)^2$,
  - **reflects** all queries.

References

References

Arenis, S. F., Westphal, B., Dietsch, D., Muñiz, M., Andisha, A. S., and Podelski, A. (2016). Ready for testing: ensuring conformance to industrial standards through formal verification. *Formal Asp. Comput.*, 28(3):499–527.

Herrera, C. (2011). Reducing quasi-equal clocks in networks of timed automata. Master's thesis, Albert-Ludwigs-Universität Freiburg.

Herrera, C. (2017). *The Class of Timed Automata with Quasi-Equal Clocks*. PhD thesis, Albert-Ludwigs-Universität Oldenburg.

Herrera, C. and Westphal, B. (2015). Quasi-equal clock reduction: Eliminating assumptions on networks. In Piterman, N., editor, *HVC*, volume 9434 of *LNCS*, pages 173–189. Springer.

Herrera, C. and Westphal, B. (2016). The model checking problem in networks with quasi-equal clocks. In Dyreson, C. E., Hansen, M. R., and Hunsberger, L., editors, *TIME*, pages 21–30. IEEE Computer Society.

Herrera, C., Westphal, B., and Podelski, A. (2014). Quasi-equal clock reduction: More networks, more queries. In Ábrahám, E. and Havelund, K., editors, *TACAS*, volume 8413 of *LNCS*, pages 295–309. Springer.

Olderog, E.-R. and Dierks, H. (2008). *Real-Time Systems - Formal Specification and Automatic Verification*. Cambridge University Press.