Robust Controller Synthesis in Timed Büchi Automata: A Symbolic Approach

Damien Busatto-Gaston

Aix-Marseille Université, LIS

April 4, 2019

joint work with Benjamin Monmege, Pierre-Alain Reynier and Ocan Sankur

Motivation: real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller??}} \quad \models \quad \text{Spec}$$

Motivation: real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller??}} \quad \models \quad \text{Spec}$$

Motivation: real-time synthesis

$$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller??}} \quad \models \quad \text{Spec}$$

Real-time requirements/environment $\Longrightarrow$ real-time controller

Motivation: real-time synthesis

$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller??}} \quad \models \quad \text{Spec}$

$\boxed{\text{Arena}} \| \boxed{\text{Environment's strategy}} \| \boxed{\text{Controller's strategy??}} \quad \models \quad \text{Büchi}$

Two-player game

Real-time requirements/environment $\Longrightarrow$ real-time controller

Motivation: real-time synthesis

$\boxed{\text{Environment}} \quad \| \quad \boxed{\text{Controller??}} \quad \models \quad \text{Spec}$

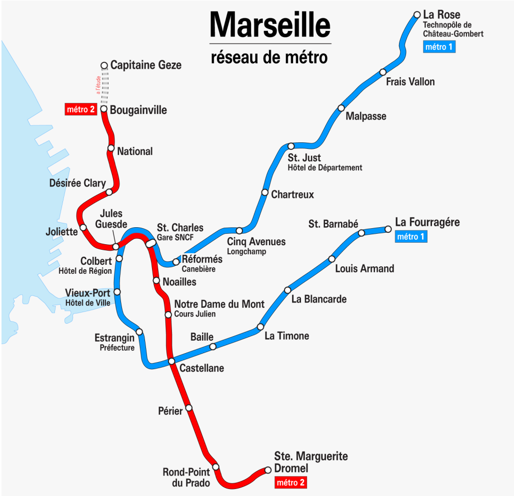
$\boxed{\text{Arena}} \| \boxed{\text{Environment's strategy}} \| \boxed{\text{Controller's strategy??}} \quad \models \quad \text{Büchi}$

Two-player game

Real-time requirements/environment $\Longrightarrow$ real-time controller

Two-player **timed** game

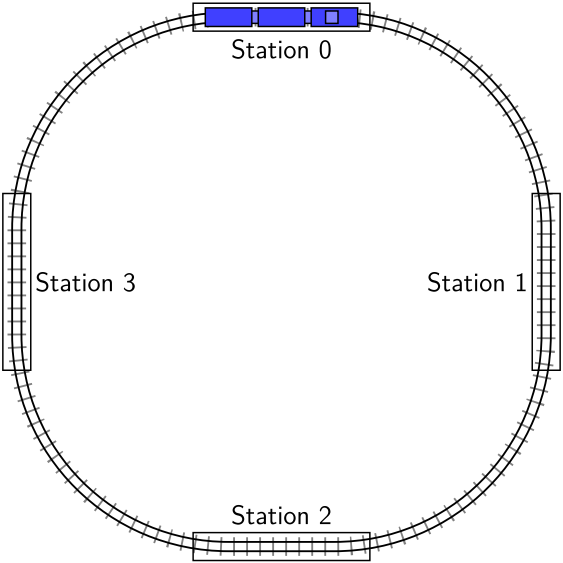
Example: Train Control



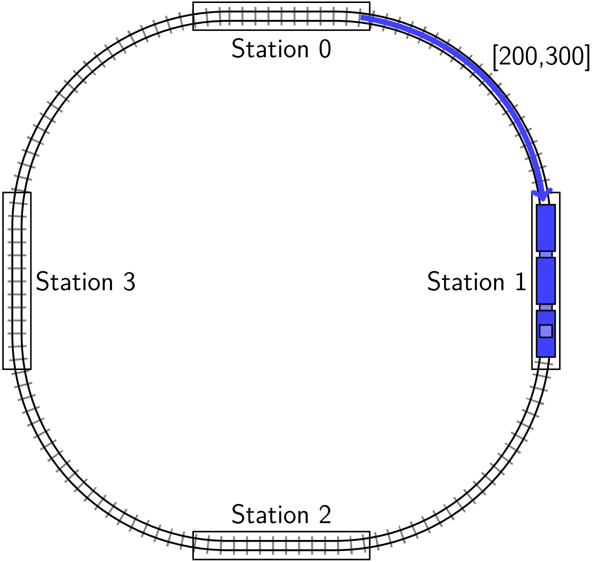
[1]

[1]By Maximilian Dörrbecker (Chumwa), CC BY-SA 2.5

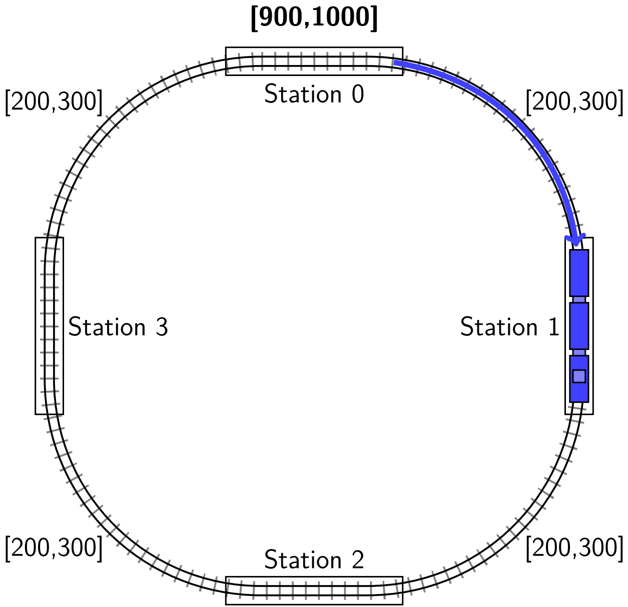
Example: Train Control



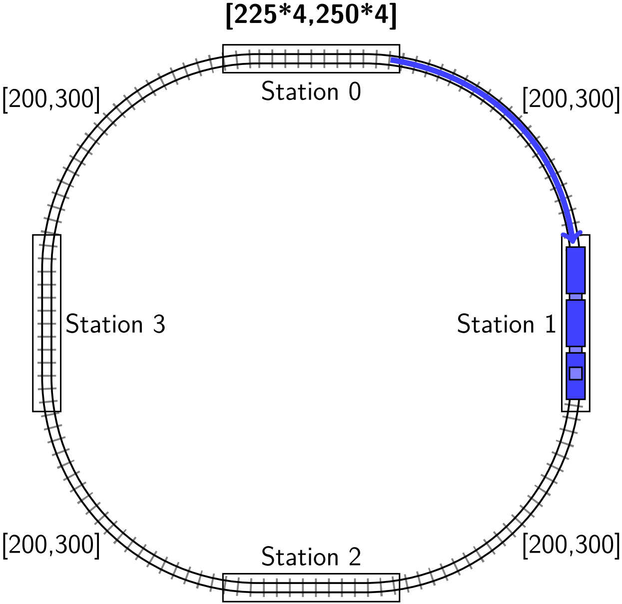
Example: Train Control



Example: Train Control



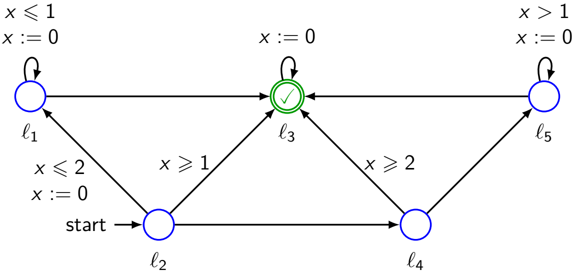
Example: Train Control



Timed Automata

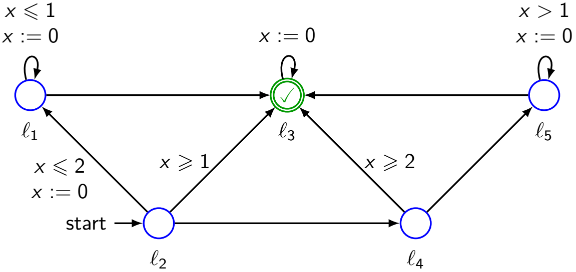
Timed Automaton

Finite set of clocks : $\{x\}$ Global invariant $x \in [0, 3]$



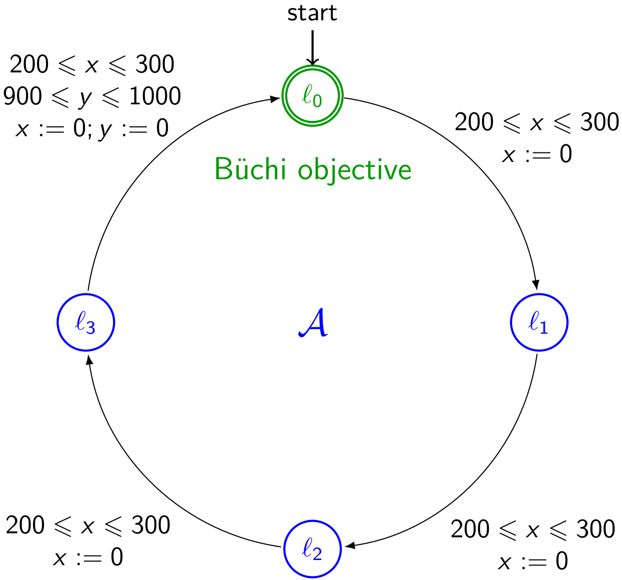
Timed Automaton

Finite set of clocks : $\{x\}$ Global invariant $x \in [0, 3]$

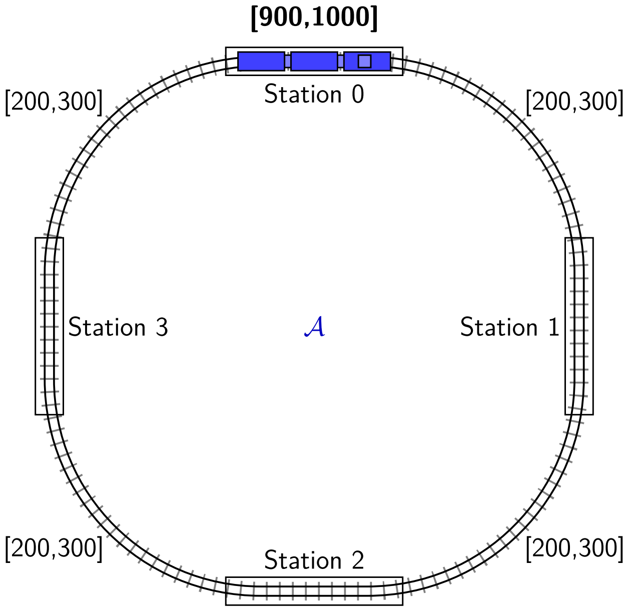


$$(\ell_2, x = 0) \xrightarrow[\text{time elapse}]{1.4} (\ell_2, x = 1.4) \xrightarrow[\text{transition}]{\ell_2 \to \ell_3} (\checkmark, x = 1.4) \to \cdots$$

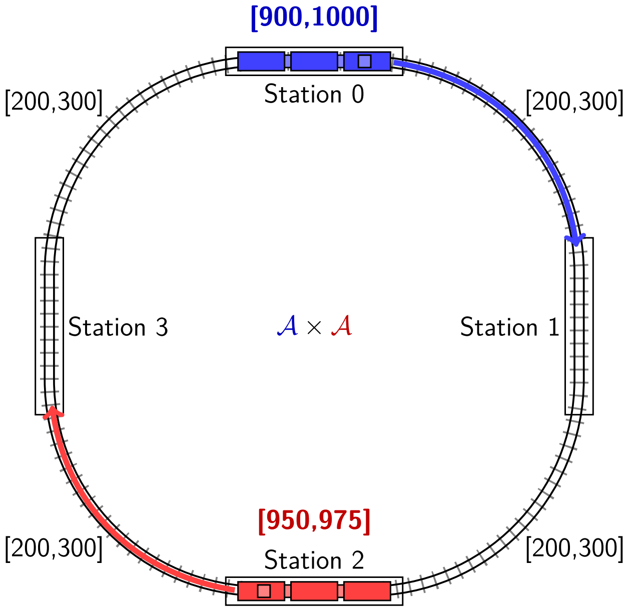
Train Control as a Timed Automaton



Train Control as a Timed Automaton

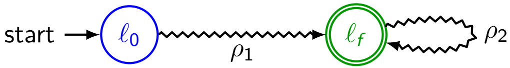


Train Control as a Timed Automaton



Büchi on a finite graph

Finding a path that goes through winning vertices infinitely often
$\Leftrightarrow$ Finding a winning lasso around one of the targets

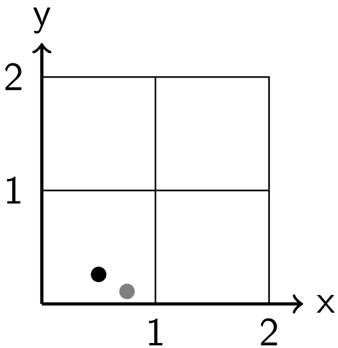


Double BFS algorithm

- First BFS from ℓ_0 $\rightsquigarrow$ find all reachable ℓ_f
- From each such ℓ_f, launch a second BFS $\rightsquigarrow$ look for loops around ℓ_f

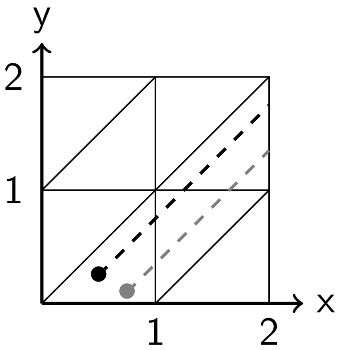
Regions

Valuations $(x = \cdots, y = \cdots)$ $\Leftrightarrow$ Points of $\mathbb{R}^2_{\geqslant 0}$

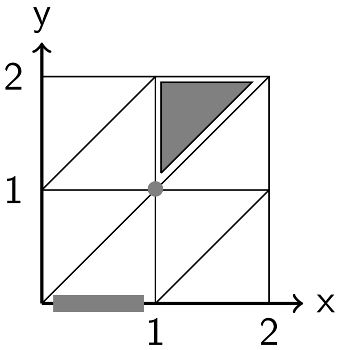


Regions

Valuations $(x = \cdots, y = \cdots) \quad \Leftrightarrow \quad$ Points of $\mathbb{R}^2_{\geqslant 0}$



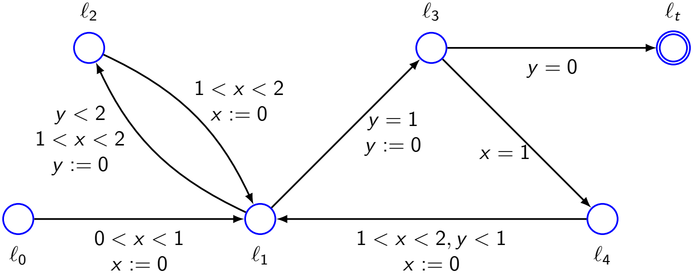
Regions



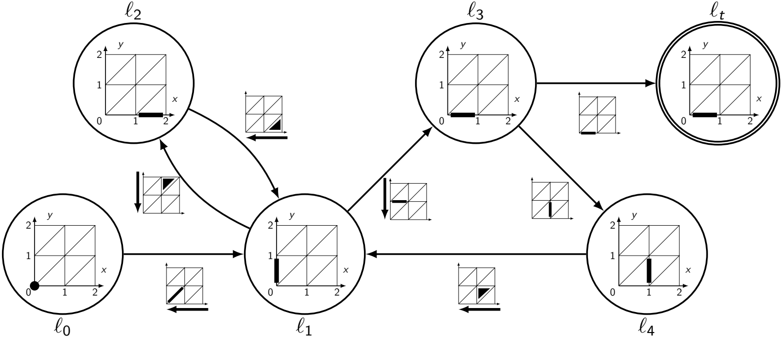
Region: set of time-abstract bissimilar points

- finite number of regions
- exponential in the number of clocks

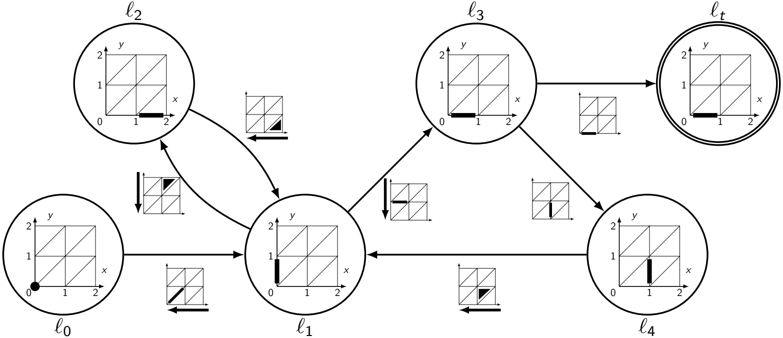
Region Automaton



Region Automaton



Region Automaton



Exponential blowup in the size of $\mathcal{A}$

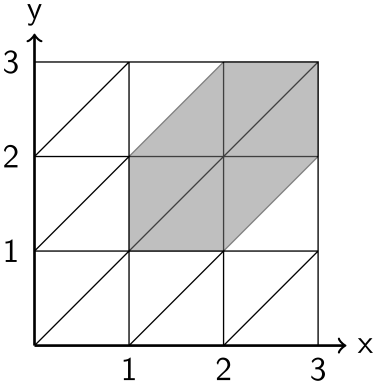
Complexity results

- Reachability in a **timed automaton**: **PSPACE**-complete[Alur and Dill, 1994]
- $\Rightarrow$ Büchi emptiness is also **PSPACE**-complete

Complexity results

- Reachability in a **timed automaton**: **PSPACE**-complete[Alur and Dill, 1994]
- $\Rightarrow$ Büchi emptiness is also **PSPACE**-complete
- algorithms based on regions are not amenable to implementation
  - train example: $\sim 10^6$ regions
  - after rescaling: $\sim 10^4$ regions

Zones

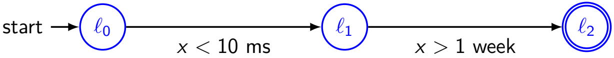


Difference Bound Matrix (DBM)

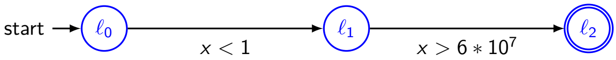
$$\begin{array}{cc} & \begin{array}{ccc} 0 & x & y \end{array} \\ \begin{array}{c} 0 \\ x \\ y \end{array} & \begin{pmatrix} \leqslant 0 & \leqslant -1 & \leqslant -1 \\ < 3 & \leqslant 0 & < 1 \\ < 3 & < 1 & \leqslant 0 \end{pmatrix} \end{array}$$

$$1 \leqslant x < 3 \quad \wedge \quad 1 \leqslant y < 3 \quad \wedge \quad x - 1 < y < x + 1$$

Zones vs regions

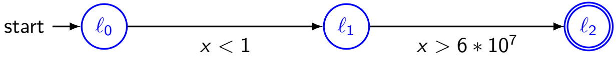


Zones vs regions



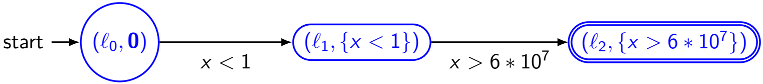
- 60 million regions
- 3 zones

Zones vs regions

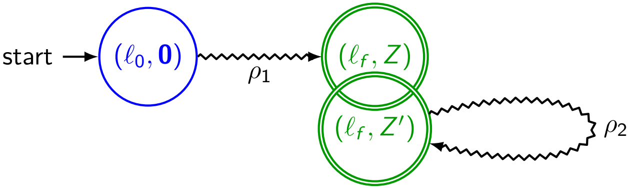


- 60 million regions
- 3 zones

Zone Graph:



Double BFS on the Zone Graph



Double BFS algorithm

- First BFS from $(\ell_0, \mathbf{0}) \rightsquigarrow$ find all reachable (ℓ_f, Z)
- From each such ℓ_f, launch a second BFS $\rightsquigarrow$ look for loops around ℓ_f
- For every such ρ_2, compute the largest infinitely iterable zone Z'
- check if Z and Z' have an intersection

Checking iterability of a loop

Fixed point reformulation

The loop ρ is infinitely iterable sarting from valuations in $\nu X.\ \mathsf{Pre}_\rho(X)$

Checking iterability of a loop

Fixed point reformulation

The loop ρ is infinitely iterable sarting from valuations in $\nu X.\ \mathsf{Pre}_\rho(X)$

Fixed Point computation

Let ρ be a path. We let $N = 2(n+1)^2$. If $\mathsf{Pre}_{\rho^{N+1}}(\top) \subsetneq \mathsf{Pre}_{\rho^N}(\top)$, then $\nu X\ \mathsf{CPre}^\delta_\rho(X) = \emptyset$.

Idea: pumping argument on a structure that represent the reachability relation for valuations

Reachability relation of a path

- For a sequence of transitions ρ, $R_\rho = \{(\nu, \nu') | (\ell, \nu) \xrightarrow{\rho} (\ell', \nu')\}$

Reachability relation of a path

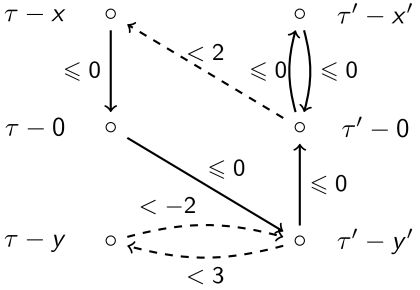
- For a sequence of transitions ρ, $R_\rho = \{(\nu, \nu') | (\ell, \nu) \xrightarrow{\rho} (\ell', \nu')\}$
- Example: $\rho = \ell_1 \xrightarrow{2<y<3, y:=0} \ell_2 \xrightarrow{x<2, x:=0} \ell_1$
- R_ρ can be expressed with linear inequalities:
  $(x, y) R_\rho (x', y') : \quad x' = 0,\ y' < 2 - x,\ y < 3,\ y' - y < -x$
- Not a zone : constraints can involve 3 or 4 clocks

Reachability relation of a path

- Example: $\rho = \ell_1 \xrightarrow{2<y<3, y:=0} \ell_2 \xrightarrow{x<2, x:=0} \ell_1$
- R_ρ can be expressed with linear inequalities:
  $(x, y) R_\rho (x', y')$: $\; x' = 0,\ y' < 2 - x,\ y < 3,\ y' - y < -x$
- Not a zone : constraints can involve 3 or 4 clocks
- Efficient representations as *Constraint Graphs*: express constraints on the last date of reset of clocks and not on their values
- Add a global clock τ, and rewrite constraints, st $(x, y) R_\rho (x', y')$:
  $\exists \tau \leqslant \tau', (\tau' - x') - (\tau' - 0) = 0,\ (\tau' - y) - (\tau' - 0) \leqslant 0,\ (\tau' - 0) - (\tau - x) < 2,\ (\tau - 0) - (\tau' - y') \leqslant 0,\ (\tau - y) - (\tau' - y') < -2,\ (\tau' - y') - (\tau - y) < 3,\ (\tau - x) - (\tau - 0) \leqslant 0$

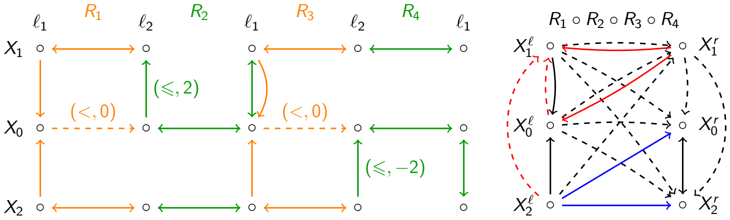
Reachability relation of a path

- Example: $\rho = \ell_1 \xrightarrow{2<y<3, y:=0} \ell_2 \xrightarrow{x<2, x:=0} \ell_1$
- Efficient representations as *Constraint Graphs*: express constraints on the last date of reset of clocks and not on their values



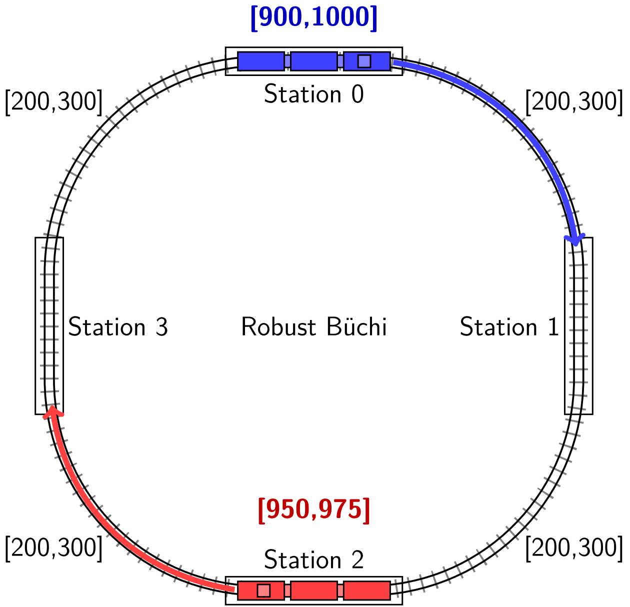
Reachability relation of a path

- Example: $\rho = \ell_1 \xrightarrow{2<y<3, y:=0} \ell_2 \xrightarrow{x<2, x:=0} \ell_1$
- Efficient representations as *Constraint Graphs*: express constraints on the last date of reset of clocks and not on their values
- Enables fast composition of relations: $R_{\rho_1} \circ R_{\rho_2} = R_{\rho_1 \cdot \rho_2}$

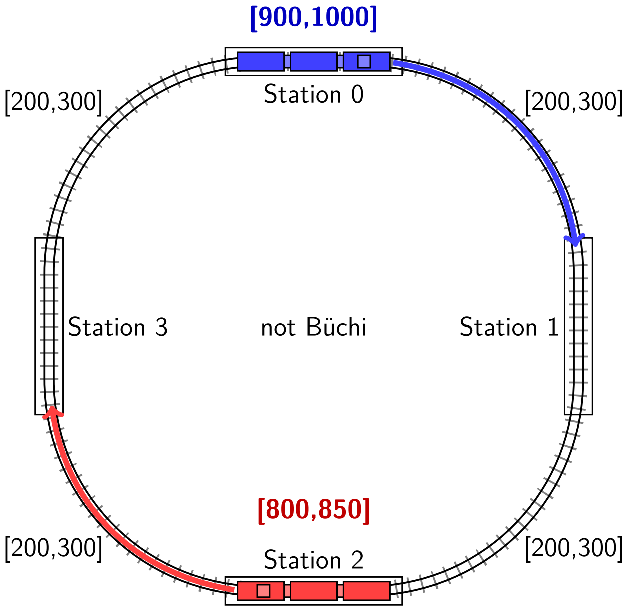


Robustness

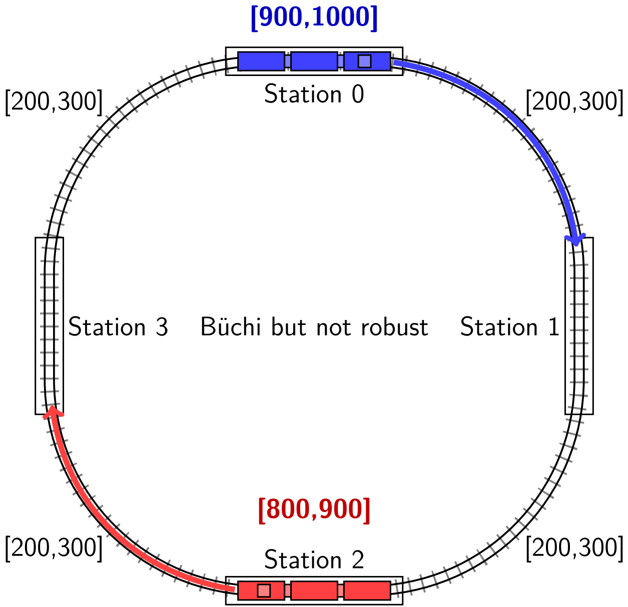
Robustness


[900,1000]
Station 0
[200,300]
[200,300]
Station 3
Robust Büchi
Station 1
[950,975]
[200,300]
Station 2
[200,300]

Robustness


[900,1000]
Station 0
[200,300]
[200,300]
Station 3
not Büchi
Station 1
[800,850]
[200,300]
Station 2
[200,300]

Robustness



[900,1000]
Station 0
[200,300]
[200,300]
Station 3
Büchi but not robust
Station 1
[800,900]
[200,300]
Station 2
[200,300]

Robust Büchi

Perturbation Game $\mathcal{G}(\mathcal{A})^\delta$

- 2 Players: Controller and Environment
- Arena: a Timed Automaton $\mathcal{A}$
- Controller chooses delays and transitions to be taken
- Environment can nudge the delays by adding $\varepsilon \in [-\delta, +\delta]$
- Objective for Controller: Büchi condition
- Objective for Environment: Controller loses

Robust Büchi

Perturbation Game $\mathcal{G}(\mathcal{A})^\delta$

- 2 Players: Controller and Environment
- Arena: a Timed Automaton $\mathcal{A}$
- Controller chooses delays and transitions to be taken
- Environment can nudge the delays by adding $\varepsilon \in [-\delta, +\delta]$
- Objective for Controller: Büchi condition
- Objective for Environment: Controller loses

Robust Büchi decision problem

Given $\mathcal{A}$, does there exists $\delta > 0$ such that Controller has a winning strategy in $\mathcal{G}(\mathcal{A})^\delta$?

Robust Büchi

Perturbation Game $\mathcal{G}(\mathcal{A})^\delta$

- 2 Players: Controller and Environment
- Arena: a Timed Automaton $\mathcal{A}$
- Controller chooses delays and transitions to be taken
- Environment can nudge the delays by adding $\varepsilon \in [-\delta, +\delta]$
- Objective for Controller: Büchi condition
- Objective for Environment: Controller loses

Robust Büchi decision problem

Given $\mathcal{A}$, does there exists $\delta > 0$ such that Controller has a winning strategy in $\mathcal{G}(\mathcal{A})^\delta$?

Robust controller synthesis

Given $\mathcal{A}$, construct a strategy for Controller that will win in $\mathcal{G}(\mathcal{A})^\delta$ for an arbitrarily small $\delta > 0$

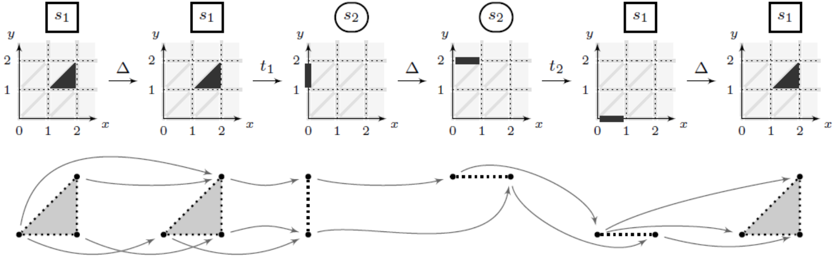
Existing work

- Robust Büchi is **PSPACE**-complete [Bouyer, Markey, Reynier, and Sankur, 2013]

Existing work

- Robust Büchi is **PSPACE**-complete [Bouyer, Markey, Reynier, and Sankur, 2013]

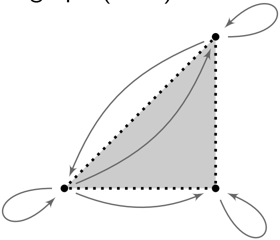
- proof heavilly relies on regions
- Search for robust aperiodic cycles in the region abstraction
- notion of folded orbit graph (FOG)



Existing work

- Robust Büchi is **PSPACE**-complete [Bouyer, Markey, Reynier, and Sankur, 2013]

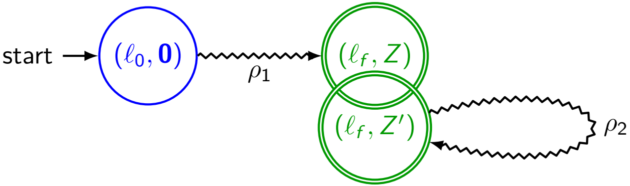
- proof heavilly relies on regions
- Search for robust aperiodic cycles in the region abstraction
- notion of folded orbit graph (FOG)



Existing work

- Robust Büchi is **PSPACE**-complete [Bouyer, Markey, Reynier, and Sankur, 2013]
- proof heavilly relies on regions
- We want to solve this with zone-based techniques

Finding robust lassos



Double BFS algorithm

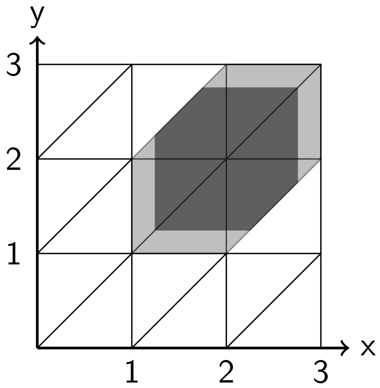
- First BFS from $(\ell_0, \mathbf{0})$ $\rightsquigarrow$ find all reachable (ℓ_f, Z)
- From each such ℓ_f, launch a second BFS $\rightsquigarrow$ look for loops around ℓ_f
- when a lasso $\rho_1 \rho_2^\omega$ is found, check if it is robust
- If not, keep going

Checking robustness of a path

- $\mathsf{Pre}_\rho(Z)$:the largest zone that reaches Z when one follows ρ (no perturbation)
- $\mathsf{Shrink}^\delta(Z)$: set of valuation that can ensure being in Z after perturbation in $[-\delta, +\delta]$

Checking robustness of a path

- $\mathrm{Pre}_\rho(Z)$:the largest zone that reaches Z when one follows ρ (no perturbation)
- $\mathrm{Shrink}^\delta(Z)$: set of valuation that can ensure being in Z after perturbation in $[-\delta, +\delta]$



- $\mathrm{CPre}^\delta_\rho(Z)$:the largest zone where Controller can ensure reaching Z by following ρ in $\mathcal{G}(\mathcal{A})^\delta$

Checking robustness of a path

- $\mathsf{Pre}_\rho(Z)$:the largest zone that reaches Z when one follows ρ (no perturbation)
- $\mathsf{Shrink}^\delta(Z)$: set of valuation that can ensure being in Z after perturbation in $[-\delta, +\delta]$
- $\mathsf{CPre}^\delta_\rho(Z)$:the largest zone where Controller can ensure reaching Z by following ρ in $\mathcal{G}(\mathcal{A})^\delta$
- δ is not fixed, $\mathsf{CPre}^\delta_\rho(Z)$ is represented as a **Shrunk DBM**

$$\begin{array}{cc} & \begin{array}{ccc} \quad 0 \quad & \quad x \quad & \quad y \quad \end{array} \\ \begin{array}{c} 0 \\ x \\ y \end{array} & \begin{pmatrix} \leqslant 0 & \leqslant -1-\delta & \leqslant -1-\delta \\ < 3-\delta & \leqslant 0 & < 1 \\ < 3-\delta & < 1 & \leqslant 0 \end{pmatrix} \end{array}$$

Checking robustness of a lasso

Fixed point reformulation

The lasso $\rho_1\rho_2^\omega$ is robustly iterable iff $\mathbf{0} \in \mathsf{CPre}^{\delta}_{\rho_1}(\nu X\ \mathsf{CPre}^{\delta}_{\rho_2}(X))$?

Checking robustness of a lasso

Fixed point reformulation

The lasso $\rho_1\rho_2^\omega$ is robustly iterable iff $\mathbf{0} \in \mathsf{CPre}^\delta_{\rho_1}(\nu X\ \mathsf{CPre}^\delta_{\rho_2}(X))$?

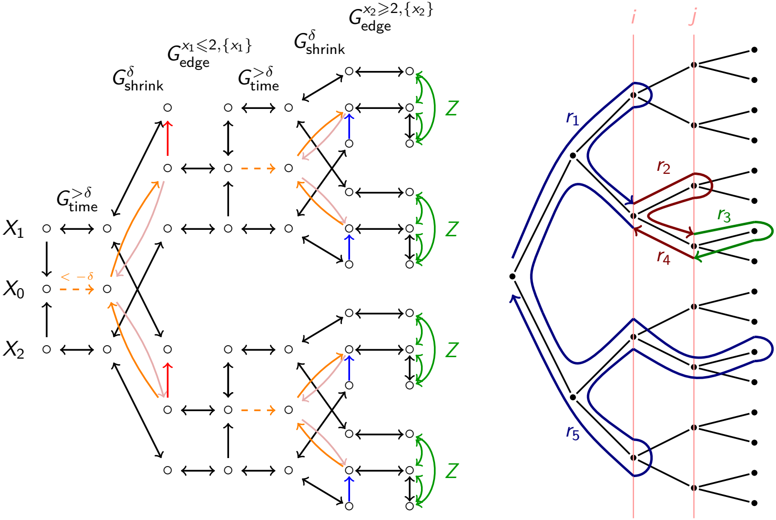
Fixed Point computation

Let ρ be a path and δ be a non-negative rational number. We let $N = 2(n+1)^2$. If $\mathsf{CPre}^\delta_{\rho^{N+1}}(\top) \subsetneq \mathsf{CPre}^\delta_{\rho^N}(\top)$, then $\nu X\ \mathsf{CPre}^\delta_\rho(X) = \emptyset$.

Idea: pumping argument on a structure that represent the reachability relation for valuations in the perturbation game

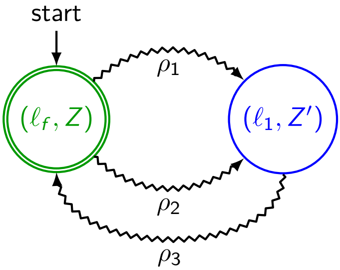
Branching constraint graph

Idea: pumping argument on a structure that represent the reachability relation for valuations in the perturbation game



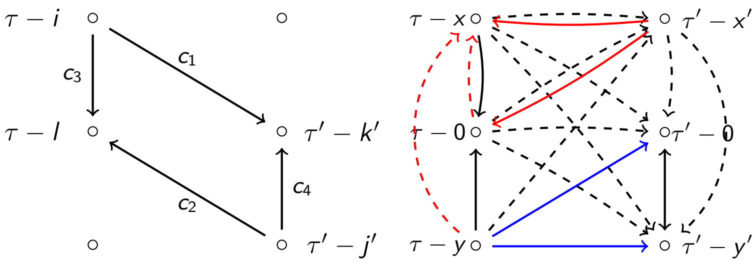
Stopping the BFS

- using zone inclusion is not complete



- solution: keep track of the whole reachability relation R along paths $(\ell_f, \nu) R_\rho (\ell_1, \nu')$
- check for inclusion of R_{ρ_1} into R_{ρ_2}
- This can be done with constraint graphs!

Checking equality/inclusion of relations



- from constraint graph to polyhedra in $\mathbb{R}^{2n}$?
- $\bigwedge_{i,j',k',l} -i + k' - j' + l \leqslant \min(c_1 + c_2, c_3 + c_4)$
- constraint graph $\Rightarrow$ canonical representation of R in $O(n^4)$

Our results

Robustness of a lasso

We can solve the robust controller synthesis problem for a given lasso in time complexity polynomial in the number of clocks and in the length of the lasso.

Maximal perturbation of a robust lasso

We can compute the largest admissible perturbation of a lasso.

Robustness of a lasso

We can solve the robust controller synthesis problem (for Büchi) using zone exploration techniques.

Conclusion

Implementation

- Prototype tool based on TChecker and UPPAAL's DBM Library
- Can handle small instances of the train example
  - 2 trains: up to 30 stations
  - 4 trains: up to 6 stations
  - $\rightarrow$ about 10^3 locations in the associated timed automaton

Possible improvements

- Smarter Büchi algorithm
- Run explorations in parallel
- Extrapolation techniques to remove the bounded clocks requirement