Control of nonlinear switched systems based on validated simulation

Symbolic and Numerical Methods for Reachability Analysis

CPSWeek 2016

Adrien Le Coënt [1], Julien Alexandre dit Sandretto [2], Alexandre Chapoutot [2], Laurent Fribourg [3]

April 11, 2016

[1]CMLA - ENS Cachan
[2]U2IS - ENSTA ParisTech
[3]LSV - ENS Cachan

Introduction

Framework

- Framework of the switched control systems: one selects the working modes of the system over time, every mode is described by ordinary differential equations (ODEs)
- The tools we have:
  - Tiling (space discretization)
  - Zonotopes (symbolic representation of sets)
  - Validated simulation

Implementation in C++ with the library DynIbex:
http://perso.ensta-paristech.fr/∼chapoutot/dynibex/
Non linear extension of the tool MINIMATOR written in Octave:
https://bitbucket.org/ukuehne/minimator/

Outline

Outline

Switched Systems

A continuous-time switched system

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

is a family of continuous-time dynamical systems with a rule σ that determines at each time which one is active

Switched Systems

A continuous-time switched system

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

is a family of continuous-time dynamical systems with a rule σ that determines at each time which one is active

- state $x \in \mathbb{R}^n$

Switched Systems

A continuous-time switched system

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

is a family of continuous-time dynamical systems with a rule σ that determines at each time which one is active

- state $x \in \mathbb{R}^n$
- bounded perturbation $d \in \mathbb{R}^m$

Switched Systems

A continuous-time switched system

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

is a family of continuous-time dynamical systems with a rule σ that determines at each time which one is active

- state $x \in \mathbb{R}^n$
- bounded perturbation $d \in \mathbb{R}^m$
- switching signal $\sigma(\cdot) : \mathbb{R}^+ \longrightarrow U$ (piecewise constant)

Switched Systems

A continuous-time switched system

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

is a family of continuous-time dynamical systems with a rule σ that determines at each time which one is active

- state $x \in \mathbb{R}^n$
- bounded perturbation $d \in \mathbb{R}^m$
- switching signal $\sigma(\cdot) : \mathbb{R}^+ \longrightarrow U$ (piecewise constant)
- $U = \{1, \dots, N\}$ finite set of modes, associated with the dynamics

$$\dot{x}(t) = f_u(x(t), d(t)),\ u \in U$$

Switched Systems

A continuous-time switched system

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

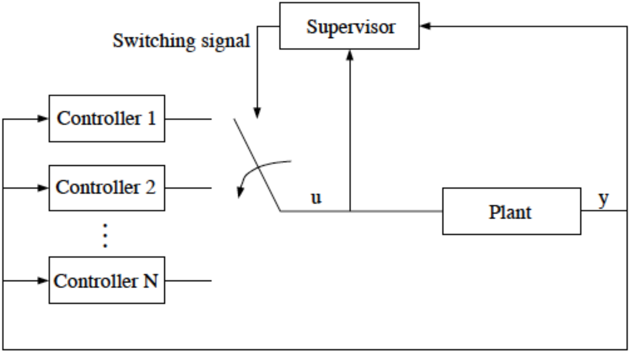
is a family of continuous-time dynamical systems with a rule σ that determines at each time which one is active

- state $x \in \mathbb{R}^n$
- bounded perturbation $d \in \mathbb{R}^m$
- switching signal $\sigma(\cdot) : \mathbb{R}^+ \longrightarrow U$ (piecewise constant)
- $U = \{1, \ldots, N\}$ finite set of modes, associated with the dynamics

$$\dot{x}(t) = f_u(x(t), d(t)), \ u \in U$$

We focus on sampled switched systems: switching instants occur periodically every τ, i.e. σ is constant on $[i\tau, (i+1)\tau)$

Controlled Switched Systems: Schematic View



Control Synthesis Problem

We consider the state-dependent control problem of synthesizing σ:

Control Synthesis Problem

We consider the state-dependent control problem of synthesizing σ:

At each sampling time $k\tau$, find the appropriate switched mode $u \in U$ according to the current value of x, in order to achieve some objectives:

Control Synthesis Problem

We consider the state-dependent control problem of synthesizing σ:

At each sampling time $k\tau$, find the appropriate switched mode $u \in U$ according to the current value of x, in order to achieve some objectives:

Given three sets R, S and B:

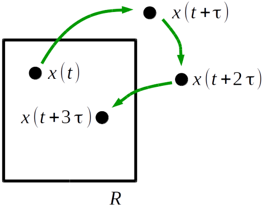
Control Synthesis Problem

We consider the state-dependent control problem of synthesizing σ:

At each sampling time $k\tau$, find the appropriate switched mode $u \in U$ according to the current value of x, in order to achieve some objectives:

Given three sets R, S and B:

- τ-stability: $x(t)$ returns in R infinitely often, at some multiples of sampling period τ



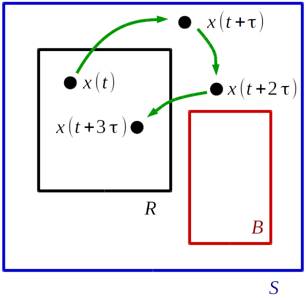
Control Synthesis Problem

We consider the state-dependent control problem of synthesizing σ:

At each sampling time $k\tau$, find the appropriate switched mode $u \in U$ according to the current value of x, in order to achieve some objectives:

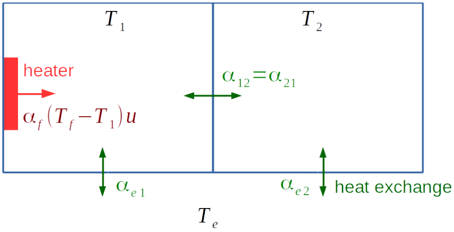
Given three sets R, S and B:

- τ-stability: $x(t)$ returns in R infinitely often, at some multiples of sampling period τ
- safety: $x(t)$ always stays in $S \backslash B$



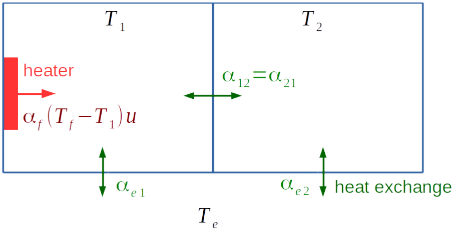
NB: classic stabilization impossible here (no common equilibrium pt)
$\rightsquigarrow$ practical stability

Example: Two-room apartment



$$\begin{pmatrix} \dot{T}_1 \\ T_2 \end{pmatrix} = \begin{pmatrix} -\alpha_{21} - \alpha_{e1} - \alpha_f u & \alpha_{21} \\ \alpha_{12} & -\alpha_{12} - \alpha_{e2} \end{pmatrix} \begin{pmatrix} T_1 \\ T_2 \end{pmatrix} + \begin{pmatrix} \alpha_{e1} T_e + \alpha_f T_f u \\ \alpha_{e2} T_e \end{pmatrix}.$$

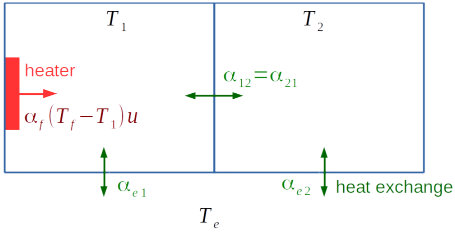
Example: Two-room apartment



$$\begin{pmatrix} \dot{T}_1 \\ \dot{T}_2 \end{pmatrix} = \begin{pmatrix} -\alpha_{21} - \alpha_{e1} - \alpha_f u & \alpha_{21} \\ \alpha_{12} & -\alpha_{12} - \alpha_{e2} \end{pmatrix} \begin{pmatrix} T_1 \\ T_2 \end{pmatrix} + \begin{pmatrix} \alpha_{e1} T_e + \alpha_f T_f u \\ \alpha_{e2} T_e \end{pmatrix}.$$

- Modes: $u = 0, 1$; sampling period τ

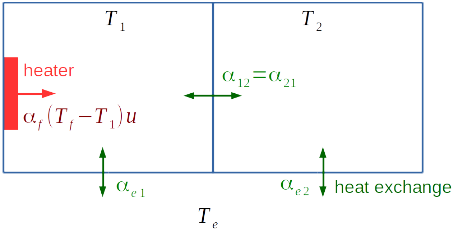
Example: Two-room apartment



$$\begin{pmatrix}\dot{T}_1\\ T_2\end{pmatrix} = \begin{pmatrix}-\alpha_{21}-\alpha_{e1}-\alpha_f u & \alpha_{21}\\ \alpha_{12} & -\alpha_{12}-\alpha_{e2}\end{pmatrix}\begin{pmatrix}T_1\\ T_2\end{pmatrix} + \begin{pmatrix}\alpha_{e1}T_e+\alpha_f T_f u\\ \alpha_{e2}T_e\end{pmatrix}.$$

- Modes: $u = 0, 1$; sampling period τ

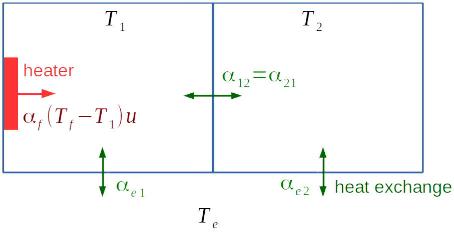
Example: Two-room apartment



$$\begin{pmatrix} \dot{T_1} \\ T_2 \end{pmatrix} = \begin{pmatrix} -\alpha_{21} - \alpha_{e1} - \alpha_f u & \alpha_{21} \\ \alpha_{12} & -\alpha_{12} - \alpha_{e2} \end{pmatrix} \begin{pmatrix} T_1 \\ T_2 \end{pmatrix} + \begin{pmatrix} \alpha_{e1} T_e + \alpha_f T_f u \\ \alpha_{e2} T_e \end{pmatrix}.$$

- Modes: $u = 0, 1$; sampling period τ
- A pattern π is a finite sequence of modes, e.g. $(0 \cdot 1 \cdot 0 \cdot 0)$

Example: Two-room apartment



$$\begin{pmatrix}\dot{T}_1\\ \dot{T}_2\end{pmatrix} = \begin{pmatrix}-\alpha_{21}-\alpha_{e1}-\alpha_f u & \alpha_{21}\\ \alpha_{12} & -\alpha_{12}-\alpha_{e2}\end{pmatrix}\begin{pmatrix}T_1\\ T_2\end{pmatrix} + \begin{pmatrix}\alpha_{e1}T_e + \alpha_f T_f u\\ \alpha_{e2}T_e\end{pmatrix}.$$

- Modes: $u = 0, 1$; sampling period τ
- A pattern π is a finite sequence of modes, e.g. $(0 \cdot 1 \cdot 0 \cdot 0)$
- A state dependent control consists in selecting at each τ a mode (or a pattern) according to the current value of the state.

Stability and safety properties for the two-room apartment

Input:

- R, B, S
- an integer K (maximal length of patterns)

Output: control tiling of R (each tile is coupled with a pattern)

Guaranteed properties: τ-stability in R, safety in $S \backslash B$

Stability and safety properties for the two-room apartment

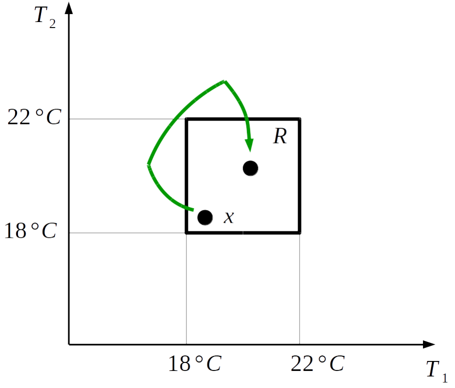
Input:

- R, B, S
- an integer K (maximal length of patterns)

Output: control tiling of R (each tile is coupled with a pattern)

Guaranteed properties: τ-stability in R, safety in $S \backslash B$

- τ-stability: after some ($\leq K$) steps of time, the temperature returns in R



Stability and safety properties for the two-room apartment

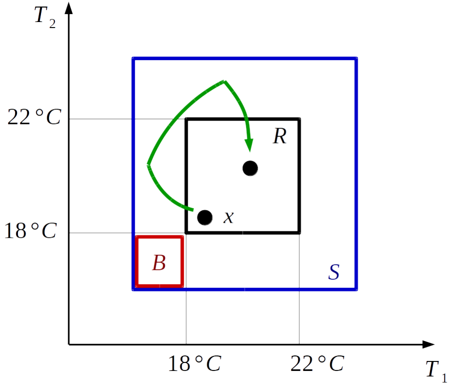
Input:

- R, B, S
- an integer K (maximal length of patterns)

Output: control tiling of R (each tile is coupled with a pattern)

Guaranteed properties: τ-stability in R, safety in $S \setminus B$

- τ-stability: after some ($\leq K$) steps of time, the temperature returns in R
- Safety: $x(t)$ always stays in $S \setminus B$.

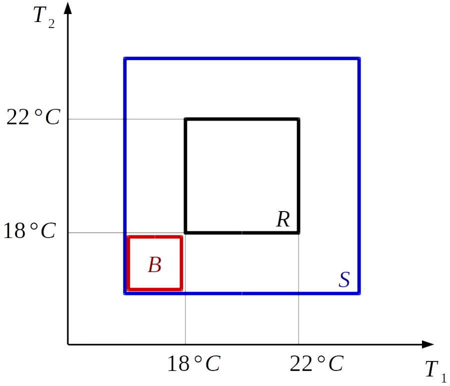


Outline

Control tiling procedure

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

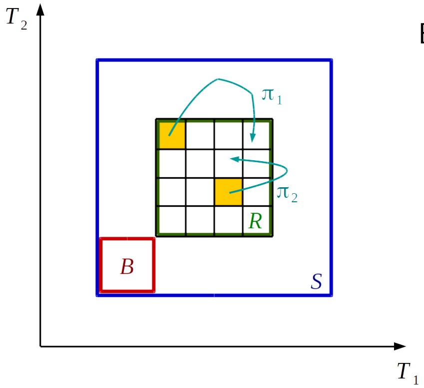
Goal: from any $x \in R$, return in R while always staying in $S \backslash B$.



Control tiling procedure

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

Goal: from any $x \in R$, return in R while always staying in $S \backslash B$.



Basic idea:

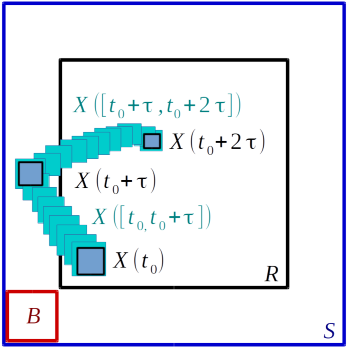
- Generate a tiling of R
- Look for patterns (input sequences) mapping the tiles into R while always staying in $S \backslash B$
- If it fails, generate another tiling.

Controlling a tile

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

Example of a validated pattern of length 2 mapping the tile X into R with safety in $S \backslash B$:

- mode u is applied during $[t_0, t_0 + \tau]$
- mode v is applied during $[t_0 + \tau, t_0 + 2\tau]$



τ-stability:

- $X(t_0) \subset R$
- $X(t_0 + 2\tau) \subset R$

safety:

- $\forall t \in [t_0, t_0 + 2\tau]$, $X(t) \subset S \backslash B$

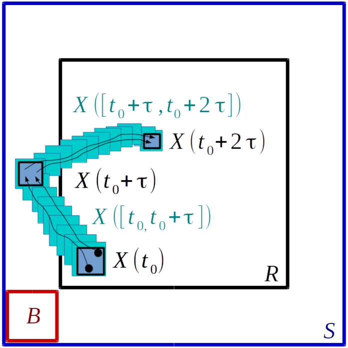
- Pattern $u \cdot v$ depends only on X

Controlling a tile

$$\dot{x}(t) = f_{\sigma(t)}(x(t), d(t))$$

Example of a validated pattern of length 2 mapping the tile X into R with safety in $S \backslash B$:

- mode u is applied during $[t_0, t_0 + \tau]$
- mode v is applied during $[t_0 + \tau, t_0 + 2\tau]$



τ-stability:

- $X(t_0) \subset R$
- $X(t_0 + 2\tau) \subset R$

safety:

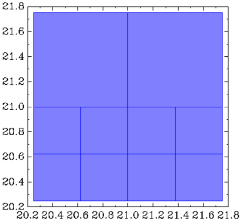
- $\forall t \in [t_0, t_0 + 2\tau]$, $X(t) \subset S \backslash B$

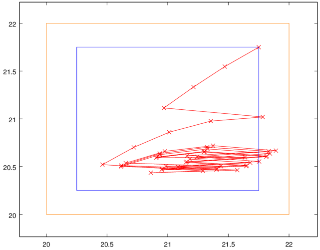
- Pattern $u \cdot v$ depends only on X

Control tiling for the two-room apartment

Inputs: $R = [20.25, 21.75] \times [20.25, 21.75]$, $S = [20, 22] \times [20, 22]$, $K = 4$

Outputs:

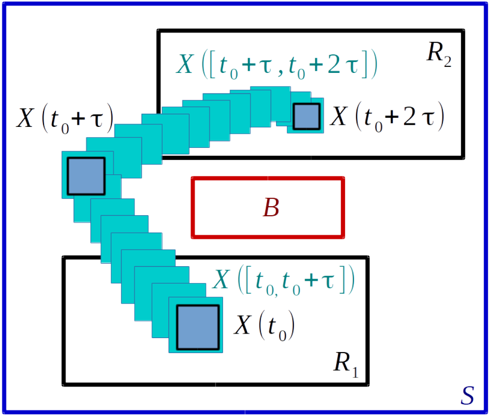




Tiling of R (left) and simulation for the initial condition $(21.75, 21.75)$ (right).

Control tiling for reachability

Extension for reachability: compute patterns such that tile X belonging to R_1 is sent into an objective set R_2.



τ-reachability:

- $X(t_0) \subset R_1$
- $X(t_0 + 2\tau) \subset R_2$

safety:

- $\forall t \in [t_0, t_0 + 2\tau]$, $X(t) \subset S \backslash B$

- Pattern $u \cdot v$ depends only on X

Outline

Validated simulation

Definition (Initial Value Problem (IVP))

Consider an ODE with a given initial condition

$$\dot{x}(t) = f(t, x(t), d(t)) \quad \text{with} \quad x(0) \in X_0,\ d(t) \in [d], \tag{1}$$

with $f : \mathbb{R}^+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ assumed to be continuous in t and d and globally Lipschitz in x. We assume that parameters d are bounded (used to represent a perturbation, a modeling error, an uncertainty on measurement, ...). An *IVP* consists in finding a function $x(t)$ described by the ODE (1) for all $d(t)$ lying in $[d]$ and for all the initial conditions in X_0.

Goal: compute an enclosure of all the solutions $x(t; x_0)$ of (1) $\forall t \in [0, T]$.

Validated simulation

Definition (Initial Value Problem (IVP))

Consider an ODE with a given initial condition

$$\dot{x}(t) = f(t, x(t), d(t)) \quad \text{with} \quad x(0) \in X_0,\ d(t) \in [d], \tag{1}$$

with $f : \mathbb{R}^+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ assumed to be continuous in t and d and globally Lipschitz in x. We assume that parameters d are bounded (used to represent a perturbation, a modeling error, an uncertainty on measurement, . . .). An *IVP* consists in finding a function $x(t)$ described by the ODE (1) for all $d(t)$ lying in $[d]$ and for all the initial conditions in X_0.

Goal: compute an enclosure of all the solutions $x(t; x_0)$ of (1) $\forall t \in [0, T]$.
$\Rightarrow$ Solution approximated with a Runge-Kutta numerical scheme

Validated simulation

Definition (Initial Value Problem (IVP))

Consider an ODE with a given initial condition

$$\dot{x}(t) = f(t, x(t), d(t)) \quad \text{with} \quad x(0) \in X_0,\ d(t) \in [d], \tag{1}$$

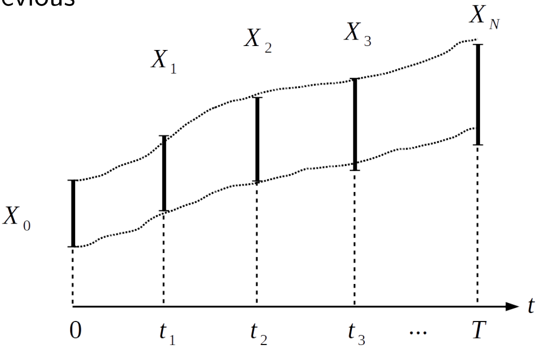
with $f : \mathbb{R}^+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ assumed to be continuous in t and d and globally Lipschitz in x. We assume that parameters d are bounded (used to represent a perturbation, a modeling error, an uncertainty on measurement, ...). An *IVP* consists in finding a function $x(t)$ described by the ODE (1) for all $d(t)$ lying in $[d]$ and for all the initial conditions in X_0.

Goal: compute an enclosure of all the solutions $x(t; x_0)$ of (1) $\forall t \in [0, T]$.
$\Rightarrow$ Solution approximated with a Runge-Kutta numerical scheme
$\Rightarrow$ Use of zonotope data structure to represent sets, permits to take parameters as intervals (i.e. $X_0 \subset \mathbb{R}^n$ and $[d] \subset \mathbb{R}^m$)

Validated simulation

Runge-Kutta numerical scheme:

- Computation of a sequence of approximations (t_n, X_n) of the solution $X(t; X_0)$
- X_i computed with the previous step: $X_i = h(t_{i-1}, X_{i-1})$



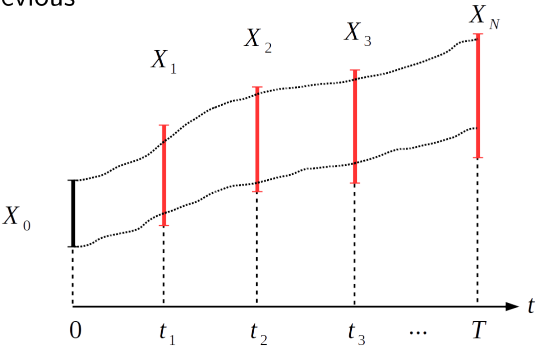
Validated simulation

Runge-Kutta numerical scheme:

- Computation of a sequence of approximations (t_n, X_n) of the solution $X(t; X_0)$
- X_i computed with the previous step: $X_i = h(t_{i-1}, X_{i-1})$

Making it guaranteed:

- Bound the error $\|x_n - x(t_n; x_{n-1})\|$



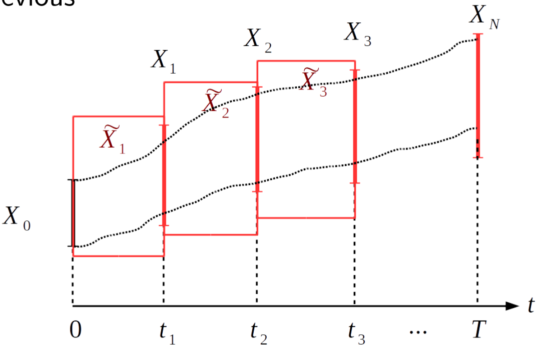
Validated simulation

Runge-Kutta numerical scheme:

- Computation of a sequence of approximations (t_n, X_n) of the solution $X(t; X_0)$
- X_i computed with the previous step: $X_i = h(t_{i-1}, X_{i-1})$

Making it guaranteed:

- Bound the error $\|x_n - x(t_n; x_{n-1})\|$
- Enclose solutions on $[t_{n-1}, t_n]$



Outline

A polynomial example

[Liu, Ozay, Topcu, Murray, 2013]

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -x_2 - 1.5x_1 - 0.5x_1^3 + u_1 + d_1 \\ x_1 + u_2 + d_2 \end{bmatrix}.$$

Control inputs given by 4 different state-feedback controllers:
$u = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = K_{\sigma(t)} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, $\sigma(t) \in U = \{1, 2, 3, 4\}$

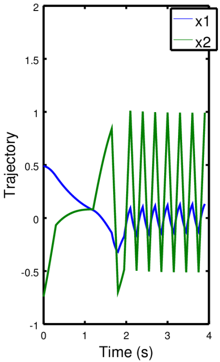
Perturbations: $d = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}$ lies in $[-0.005, 0.005] \times [-0.005, 0.005]$

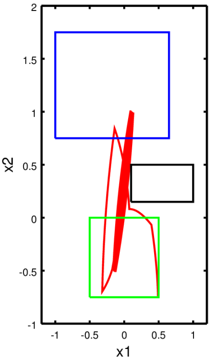
A polynomial example

[Liu, Ozay, Topcu, Murray, 2013]

Inputs: $R_1 = [-0.5, 0.5] \times [-0.75, 0.0]$, $R_2 = [-1.0, 0.65] \times [0.75, 1.75]$, $S = [-2.0, 2.0] \times [-1.5, 3.0]$, $B = [0.1, 1.0] \times [0.15, 0.5]$
$K = 12$

Outputs: 128 tiles for R_1, 128 tiles for R_2, cpu time: 2min 30s





Building ventilation

[Meyer, Nazarpour, Girard, Witrant, 2014]

Dynamics of a four-room apartment:

$$\frac{dT_i}{dt} = \sum_{j \in \mathcal{N}^*} a_{ij}(T_j - T_i) + \delta_{s_i} b_i (T_{s_i}^4 - T_i^4) + c_i \max\left(0, \frac{V_i - V_i^*}{\bar{V}_i - V_i^*}\right)(T_u - T_i).$$

$\mathcal{N}^* = \{1, 2, 3, 4, u, o, c\}$

Control inputs: V_1 and V_4 can take the values 0V or 3.5V, and V_2 and V_3 can take the values 0V or 3V

$\Rightarrow$ 16 switching modes

Perturbations:

- external environment: $T_o \in [27, 30]$
- heat transfer through the ceiling: $T_c \in [27, 30]$
- heat transfer through the underfloor: $T_u = 17$
- presence of humans: $\delta_{s_i} \in [0, 1]$

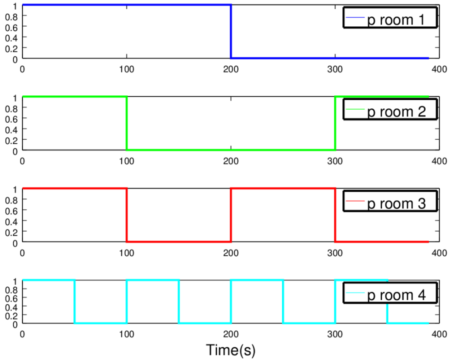
Building ventilation

[Meyer, Nazarpour, Girard, Witrant, 2014]

Inputs: $R = [20, 22]^2 \times [22, 24]^2$, $S = [19, 23]^2 \times [21, 25]^2$, $K = 1$

Outputs: 704 tiles, cpu time: 29min

Presence of humans:

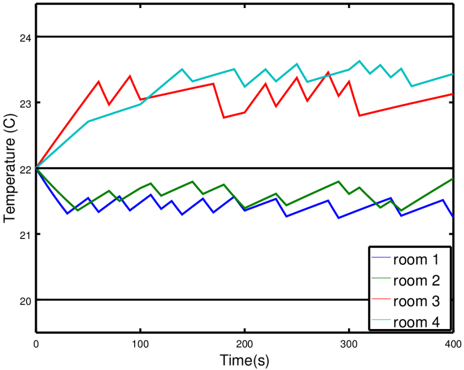


Building ventilation

[Meyer, Nazarpour, Girard, Witrant, 2014]

Inputs: $R = [20, 22]^2 \times [22, 24]^2$, $S = [19, 23]^2 \times [21, 25]^2$, $K = 1$

Outputs: 704 tiles, cpu time: 29min



Conclusion

Recap:

- Guaranteed control
- Tiling of the state-space
- Validated simulation
- High computational cost

Conclusion

Recap:

- Guaranteed control
- Tiling of the state-space
- Validated simulation
- High computational cost

Further work:

- Improvement of the research of patterns
- Optimal patterns
- Distributed control then smart grids

Conclusion

Recap:

- Guaranteed control
- Tiling of the state-space
- Validated simulation
- High computational cost

Further work:

- Improvement of the research of patterns
- Optimal patterns
- Distributed control then smart grids

Already some results for distributed control:

Adrien Le Coënt, Laurent Fribourg, Nicolas Markey, Florian De Vuyst, Ludovic Chamoin. *Distributed synthesis of state-dependent switching control*. HAL 2016.

Some References

Laurent Fribourg, Ulrich Kühne, and Romain Soulat.
Finite controlled invariants for sampled switched systems.
Formal Methods in System Design, 45(3):303–329, December 2014.

Adrien Le Coënt, Florian De Vuyst, Christian Rey, Ludovic Chamoin, and Laurent Fribourg.
Guaranteed control of switched control systems using model order reduction and state-space bisection.
Open Acces Series in Informatics, 2015.

Adrien Le Coënt, Laurent Fribourg, Nicolas Markey, Florian De Vuyst, and Ludovic Chamoin.
Distributed synthesis of state-dependent switching control.
HAL, 2016.

Some References

Laurent Fribourg, Ulrich Kühne, and Romain Soulat.
Finite controlled invariants for sampled switched systems.
Formal Methods in System Design, 45(3):303–329, December 2014.

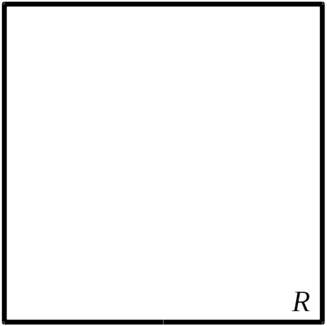
Adrien Le Coënt, Florian De Vuyst, Christian Rey, Ludovic Chamoin, and Laurent Fribourg.
Guaranteed control of switched control systems using model order reduction and state-space bisection.
Open Acces Series in Informatics, 2015.

Adrien Le Coënt, Laurent Fribourg, Nicolas Markey, Florian De Vuyst, and Ludovic Chamoin.
Distributed synthesis of state-dependent switching control.
HAL, 2016.

Thank you ! Questions?

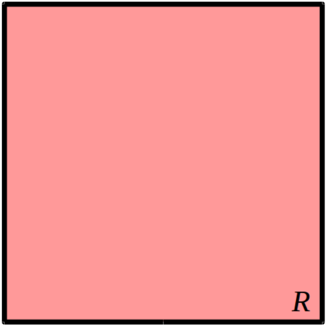
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of all the dimensions



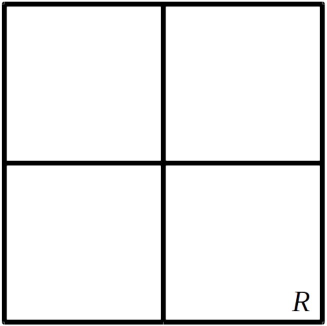
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of all the dimensions



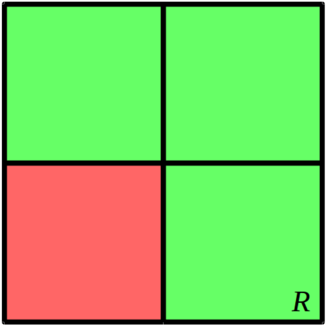
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of all the dimensions



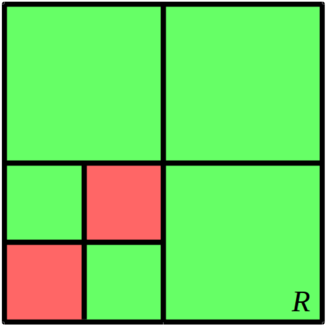
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of all the dimensions



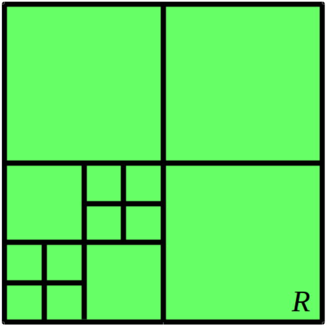
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of all the dimensions



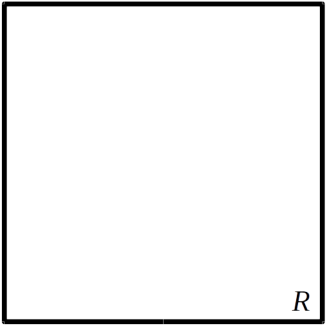
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of all the dimensions



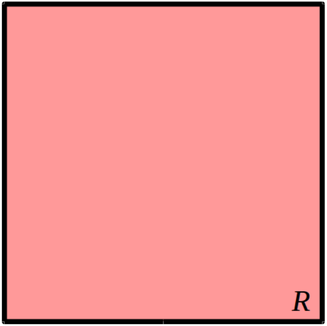
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of largest dimension



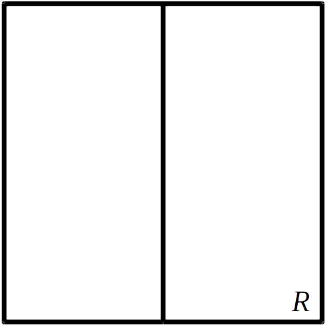
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of largest dimension



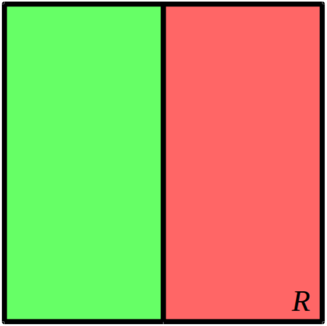
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of largest dimension



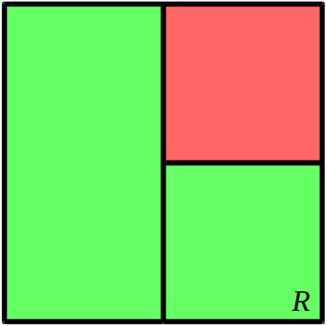
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of largest dimension



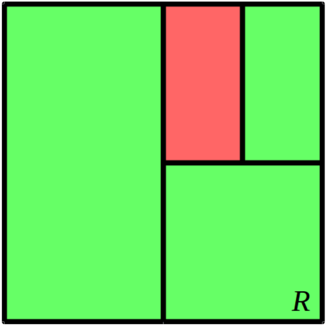
Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of largest dimension



Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of largest dimension



Sate-space bisection algorithm: several heuristics possible

Heuristics: bisection of largest dimension